

Exam Questions AWS-Certified-Data-Engineer-Associate

AWS Certified Data Engineer - Associate (DEA-C01)

<https://www.2passeasy.com/dumps/AWS-Certified-Data-Engineer-Associate/>



NEW QUESTION 1

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

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

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

Which solution will meet these requirements?

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

Answer: A

Explanation:

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

? Sharing data across Amazon Redshift clusters

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

NEW QUESTION 2

A company uses AWS Step Functions to orchestrate a data pipeline. The pipeline consists of Amazon EMR jobs that ingest data from data sources and store the data in an Amazon S3 bucket. The pipeline also includes EMR jobs that load the data to Amazon Redshift.

The company's cloud infrastructure team manually built a Step Functions state machine. The cloud infrastructure team launched an EMR cluster into a VPC to support the EMR jobs. However, the deployed Step Functions state machine is not able to run the EMR jobs.

Which combination of steps should the company take to identify the reason the Step Functions state machine is not able to run the EMR jobs? (Choose two.)

- A. Use AWS CloudFormation to automate the Step Functions state machine deployment
- B. Create a step to pause the state machine during the EMR jobs that fail
- C. Configure the step to wait for a human user to send approval through an email message
- D. Include details of the EMR task in the email message for further analysis.
- E. Verify that the Step Functions state machine code has all IAM permissions that are necessary to create and run the EMR job
- F. Verify that the Step Functions state machine code also includes IAM permissions to access the Amazon S3 buckets that the EMR jobs use
- G. Use Access Analyzer for S3 to check the S3 access properties.
- H. Check for entries in Amazon CloudWatch for the newly created EMR cluster
- I. Change the AWS Step Functions state machine code to use Amazon EMR on EKS
- J. Change the IAM access policies and the security group configuration for the Step Functions state machine code to reflect inclusion of Amazon Elastic Kubernetes Service (Amazon EKS).
- K. Query the flow logs for the VPC
- L. Determine whether the traffic that originates from the EMR cluster can successfully reach the data provider
- M. Determine whether any security group that might be attached to the Amazon EMR cluster allows connections to the data source servers on the informed ports.
- N. Check the retry scenarios that the company configured for the EMR job
- O. Increase the number of seconds in the interval between each EMR task
- P. Validate that each fallback state has the appropriate catch for each decision state
- Q. Configure an Amazon Simple Notification Service (Amazon SNS) topic to store the error messages.

Answer: BD

Explanation:

To identify the reason why the Step Functions state machine is not able to run the EMR jobs, the company should take the following steps:

? Verify that the Step Functions state machine code has all IAM permissions that are necessary to create and run the EMR jobs. The state machine code should have an IAM role that allows it to invoke the EMR APIs, such as RunJobFlow, AddJobFlowSteps, and DescribeStep. The state machine code should also have IAM permissions to access the Amazon S3 buckets that the EMR jobs use as input and output locations. The company can use Access Analyzer for S3 to check the access policies and permissions of the S3 buckets. Therefore, option B is correct.

? Query the flow logs for the VPC. The flow logs can provide information about the network traffic to and from the EMR cluster that is launched in the VPC. The company can use the flow logs to determine whether the traffic that originates from the EMR cluster can successfully reach the data providers, such as Amazon RDS, Amazon Redshift, or other external sources. The company can also determine whether any security group that might be attached to the EMR cluster allows connections to the data source servers on the informed ports. The company can use Amazon VPC Flow Logs or Amazon CloudWatch Logs Insights to query the flow logs. Therefore, option D is correct.

Option A is incorrect because it suggests using AWS CloudFormation to automate the Step Functions state machine deployment. While this is a good practice to ensure consistency and repeatability of the deployment, it does not help to identify the reason why the state machine is not able to run the EMR jobs. Moreover, creating a step to pause the state machine during the EMR jobs that fail and wait for a human user to send approval through an email message is not a reliable way to troubleshoot the issue. The company should use the Step Functions console or API to monitor the execution history and status of the state machine, and use Amazon CloudWatch to view the logs and metrics of the EMR jobs. Option C is incorrect because it suggests changing the AWS Step Functions state machine code to use Amazon EMR on EKS. Amazon EMR on EKS is a service that allows you to run EMR jobs on Amazon Elastic Kubernetes Service (Amazon EKS) clusters. While this service has some benefits, such as lower cost and faster execution time, it does not support all the features and integrations that EMR on EC2 does, such as EMR Notebooks, EMR Studio, and EMRFS. Therefore, changing the state machine code to use EMR on EKS may not be compatible with the existing data pipeline and may introduce new issues. Option E is incorrect because it suggests checking the retry scenarios that the company configured for the EMR jobs. While this is a good practice to handle transient failures and errors, it does not help to identify the root cause of why the state machine is not able to run the EMR jobs. Moreover, increasing the number of seconds in the interval between each EMR task may not improve the success rate of the jobs, and may

increase the execution time and cost of the state machine. Configuring an Amazon SNS topic to store the error messages may help to notify the company of any failures, but it does not provide enough information to troubleshoot the issue.

References:

- ? 1: Manage an Amazon EMR Job - AWS Step Functions
- ? 2: Access Analyzer for S3 - Amazon Simple Storage Service
- ? 3: Working with Amazon EMR and VPC Flow Logs - Amazon EMR
- ? [4]: Analyzing VPC Flow Logs with Amazon CloudWatch Logs Insights - Amazon Virtual Private Cloud
- ? [5]: Monitor AWS Step Functions - AWS Step Functions
- ? [6]: Monitor Amazon EMR clusters - Amazon EMR
- ? [7]: Amazon EMR on Amazon EKS - Amazon EMR

NEW QUESTION 3

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 4

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 5

A data engineer needs to use an Amazon QuickSight dashboard that is based on Amazon Athena queries on data that is stored in an Amazon S3 bucket. When the data engineer connects to the QuickSight dashboard, the data engineer receives an error message that indicates insufficient permissions.

Which factors could cause to the permissions-related errors? (Choose two.)

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

Answer: CD

Explanation:

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

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

? Using Amazon Athena as a Data Source

? Granting Amazon QuickSight Access to AWS Resources

? Encrypting Data at Rest in Amazon S3

NEW QUESTION 6

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

Which solution will give the company the ability to use Spark to access Athena?

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

Answer: C

Explanation:

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

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

? Using Apache Spark in Amazon Athena

? Athena JDBC Driver

? Spark SQL

? Athena query settings

? [Athena workgroups]

? [Athena query editor]

NEW QUESTION 7

A security company stores IoT data that is in JSON format in an Amazon S3 bucket. The data structure can change when the company upgrades the IoT devices. The company wants to create a data catalog that includes the IoT data. The company's analytics department will use the data catalog to index the data.

Which solution will meet these requirements MOST cost-effectively?

- A. Create an AWS Glue Data Catalog
- B. Configure an AWS Glue Schema Registry
- C. Create a new AWS Glue workload to orchestrate the ingestion of the data that the analytics department will use into Amazon Redshift Serverless.
- D. Create an Amazon Redshift provisioned cluster
- E. Create an Amazon Redshift Spectrum database for the analytics department to explore the data that is in Amazon S3. Create Redshift stored procedures to load the data into Amazon Redshift.
- F. Create an Amazon Athena workgroup
- G. Explore the data that is in Amazon S3 by using Apache Spark through Athena
- H. Provide the Athena workgroup schema and tables to the analytics department.
- I. Create an AWS Glue Data Catalog
- J. Configure an AWS Glue Schema Registry
- K. Create AWS Lambda user defined functions (UDFs) by using the Amazon Redshift Data API
- L. Create an AWS Step Functions job to orchestrate the ingestion of the data that the analytics department will use into Amazon Redshift Serverless.

Answer: C

Explanation:

The best solution to meet the requirements of creating a data catalog that includes the IoT data, and allowing the analytics department to index the data, most cost-effectively, is to create an Amazon Athena workgroup, explore the data that is in Amazon S3 by using Apache Spark through Athena, and provide the Athena workgroup schema and tables to the analytics department.

Amazon Athena is a serverless, interactive query service that makes it easy to analyze data directly in Amazon S3 using standard SQL or Python¹. Amazon Athena also supports Apache Spark, an open-source distributed processing framework that can run large-scale data analytics applications across clusters of servers². You can use Athena to run Spark code on data in Amazon S3 without having to set up, manage, or scale any infrastructure. You can also use Athena to create and manage external tables that point to your data in Amazon S3, and store them in an external data catalog, such as AWS Glue Data Catalog, Amazon Athena Data Catalog, or your own Apache Hive metastore³. You can create Athena workgroups to separate query execution and resource allocation based on different criteria, such as users, teams, or applications⁴. You can share the schemas and tables in your Athena workgroup with other users or applications, such as Amazon QuickSight, for data visualization and analysis⁵.

Using Athena and Spark to create a data catalog and explore the IoT data in Amazon S3 is the most cost-effective solution, as you pay only for the queries you run

or the compute you use, and you pay nothing when the service is idle¹. You also save on the operational overhead and complexity of managing data warehouse infrastructure, as Athena and Spark are serverless and scalable. You can also benefit from the flexibility and performance of Athena and Spark, as they support various data formats, including JSON, and can handle schema changes and complex queries efficiently.

Option A is not the best solution, as creating an AWS Glue Data Catalog, configuring an AWS Glue Schema Registry, creating a new AWS Glue workload to orchestrate the ingestion of the data that the analytics department will use into Amazon Redshift Serverless, would incur more costs and complexity than using Athena and Spark. AWS Glue Data Catalog is a persistent metadata store that contains table definitions, job definitions, and other control information to help you manage your AWS Glue components⁶. AWS Glue Schema Registry is a service that allows you to centrally store and manage the schemas of your streaming data in AWS Glue Data Catalog⁷. AWS Glue is a serverless data integration service that makes it easy to prepare, clean, enrich, and move data between data stores⁸. Amazon Redshift Serverless is a feature of Amazon Redshift, a fully managed data warehouse service, that allows you to run and scale analytics without having to manage data warehouse infrastructure⁹. While these services are powerful and useful for many data engineering scenarios, they are not necessary or cost-effective for creating a data catalog and indexing the IoT data in Amazon S3. AWS Glue Data Catalog and Schema Registry charge you based on the number of objects stored and the number of requests made^{6,7}. AWS Glue charges you based on the compute time and the data processed by your ETL jobs⁸. Amazon Redshift Serverless charges you based on the amount of data scanned by your queries and the compute time used by your workloads⁹. These costs can add up quickly, especially if you have large volumes of IoT data and frequent schema changes. Moreover, using AWS Glue and Amazon Redshift Serverless would introduce additional latency and complexity, as you would have to ingest the data from Amazon S3 to Amazon Redshift Serverless, and then query it from there, instead of querying it directly from Amazon S3 using Athena and Spark.

Option B is not the best solution, as creating an Amazon Redshift provisioned cluster, creating an Amazon Redshift Spectrum database for the analytics department to explore the data that is in Amazon S3, and creating Redshift stored procedures to load the data into Amazon Redshift, would incur more costs and complexity than using Athena and Spark. Amazon Redshift provisioned clusters are clusters that you create and manage by specifying the number and type of nodes, and the amount of storage and compute capacity¹⁰. Amazon Redshift Spectrum is a feature of Amazon Redshift that allows you to query and join data across your data warehouse and your data lake using standard SQL¹¹. Redshift stored procedures are SQL statements that you can define and store in Amazon Redshift, and then call them by using the CALL command¹². While these features are powerful and useful for many data warehousing scenarios, they are not necessary or cost-effective for creating a data catalog and indexing the IoT data in Amazon S3. Amazon Redshift provisioned clusters charge you based on the node type, the number of nodes, and the duration of the cluster¹⁰. Amazon Redshift Spectrum charges you based on the amount of data scanned by your queries¹¹. These costs can add up quickly, especially if you have large volumes of IoT data and frequent schema changes. Moreover, using Amazon Redshift provisioned clusters and Spectrum would introduce additional latency and complexity, as you would have to provision and manage the cluster, create an external schema and database for the data in Amazon S3, and load the data into the cluster using stored procedures, instead of querying it directly from Amazon S3 using Athena and Spark. Option D is not the best solution, as creating an AWS Glue Data Catalog, configuring an AWS Glue Schema Registry, creating AWS Lambda user defined functions (UDFs) by using the Amazon Redshift Data API, and creating an AWS Step Functions job to orchestrate the ingestion of the data that the analytics department will use into Amazon Redshift Serverless, would incur more costs and complexity than using Athena and Spark. AWS Lambda is a serverless compute service that lets you run code without provisioning or managing servers¹³. AWS Lambda UDFs are Lambda functions that you can invoke from within an Amazon Redshift query. Amazon Redshift Data API is a service that allows you to run SQL statements on Amazon Redshift clusters using HTTP requests, without needing a persistent connection. AWS Step Functions is a service that lets you coordinate multiple AWS services into serverless workflows. While these services are powerful and useful for many data engineering scenarios, they are not necessary or cost-effective for creating a data catalog and indexing the IoT data in Amazon S3. AWS Glue Data Catalog and Schema Registry charge you based on the number of objects stored and the number of requests made^{6,7}. AWS Lambda charges you based on the number of requests and the duration of your functions¹³. Amazon Redshift Serverless charges you based on the amount of data scanned by your queries and the compute time used by your workloads⁹. AWS Step Functions charges you based on the number of state transitions in your workflows. These costs can add up quickly, especially if you have large volumes of IoT data and frequent schema changes. Moreover, using AWS Glue, AWS Lambda, Amazon Redshift Data API, and AWS Step Functions would introduce additional latency and complexity, as you would have to create and invoke Lambda functions to ingest the data from Amazon S3 to Amazon Redshift Serverless using the Data API, and coordinate the ingestion process using Step Functions, instead of querying it directly from Amazon S3 using Athena and Spark. References:

- ? What is Amazon Athena?
- ? Apache Spark on Amazon Athena
- ? Creating tables, updating the schema, and adding new partitions in the Data Catalog from AWS Glue ETL jobs
- ? Managing Athena workgroups
- ? Using Amazon QuickSight to visualize data in Amazon Athena
- ? AWS Glue Data Catalog
- ? AWS Glue Schema Registry
- ? What is AWS Glue?
- ? Amazon Redshift Serverless
- ? Amazon Redshift provisioned clusters
- ? Querying external data using Amazon Redshift Spectrum
- ? Using stored procedures in Amazon Redshift
- ? What is AWS Lambda?
- ? [Creating and using AWS Lambda UDFs]
- ? [Using the Amazon Redshift Data API]
- ? [What is AWS Step Functions?]
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

NEW QUESTION 8

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

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

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

Answer: A

Explanation:

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

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

You also do not need to provision or manage any servers or clusters, as Step Functions scales automatically based on the demand.

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

? AWS Step Functions

? AWS Glue

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

NEW QUESTION 9

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

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

Which solution will meet these requirements with the LEAST latency?

- A. Schedule an AWS Glue crawler to run every morning.
- B. Manually run the AWS Glue CreatePartition API twice each day.
- C. Use code that writes data to Amazon S3 to invoke the Boto3 AWS Glue create partition API call.
- D. Run the MSCK REPAIR TABLE command from the AWS Glue console.

Answer: C

Explanation:

The best solution to ensure that the AWS Glue Data Catalog synchronizes with the S3 storage when the company adds new partitions to the bucket with the least latency is to use code that writes data to Amazon S3 to invoke the Boto3 AWS Glue create partition API call. This way, the Data Catalog is updated as soon as new data is written to S3, and the partition information is immediately available for querying by other

services. The Boto3 AWS Glue create partition API call allows you to create a new partition in the Data Catalog by specifying the table name, the database name, and the partition values¹. You can use this API call in your code that writes data to S3, such as a Python script or an AWS Glue ETL job, to create a partition for each new S3 object key that matches the partitioning scheme.

Option A is not the best solution, as scheduling an AWS Glue crawler to run every morning would introduce a significant latency between the time new data is written to S3 and the time the Data Catalog is updated. AWS Glue crawlers are processes that connect to a data store, progress through a prioritized list of classifiers to determine the schema for your data, and then create metadata tables in the Data Catalog². Crawlers can be scheduled to run periodically, such as daily or hourly, but they cannot run continuously or in real-time. Therefore, using a crawler to synchronize the Data Catalog with the S3 storage would not meet the requirement of the least latency.

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

Option D is not the best solution, as running the MSCK REPAIR TABLE command from the AWS Glue console would also introduce a significant latency between the time new data is written to S3 and the time the Data Catalog is updated. The MSCK REPAIR TABLE command is a SQL command that you can run in the AWS Glue console to add partitions to the Data Catalog based on the S3 object keys that match the partitioning scheme³. However, this command is not meant to be run frequently or in real-time, as it can take a long time to scan the entire S3 bucket and add the partitions. Therefore, using this command to synchronize the Data Catalog with the S3 storage would not meet the requirement of the least latency. References:

? AWS Glue CreatePartition API

? Populating the AWS Glue Data Catalog

? MSCK REPAIR TABLE Command

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

NEW QUESTION 10

A data engineer must use AWS services to ingest a dataset into an Amazon S3 data lake. The data engineer profiles the dataset and discovers that the dataset contains personally identifiable information (PII). The data engineer must implement a solution to profile the dataset and obfuscate the PII.

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

- A. Use an Amazon Kinesis Data Firehose delivery stream to process the dataset
- B. Create an AWS Lambda transform function to identify the PII
- C. Use an AWS SDK to obfuscate the PII
- D. Set the S3 data lake as the target for the delivery stream.
- E. Use the Detect PII transform in AWS Glue Studio to identify the PII
- F. Obfuscate the PII
- G. Use an AWS Step Functions state machine to orchestrate a data pipeline to ingest the data into the S3 data lake.
- H. Use the Detect PII transform in AWS Glue Studio to identify the PII
- I. Create a rule in AWS Glue Data Quality to obfuscate the PII
- J. Use an AWS Step Functions state machine to orchestrate a data pipeline to ingest the data into the S3 data lake.
- K. Ingest the dataset into Amazon DynamoDB
- L. Create an AWS Lambda function to identify and obfuscate the PII in the DynamoDB table and to transform the data
- M. Use the same Lambda function to ingest the data into the S3 data lake.

Answer: C

Explanation:

AWS Glue is a fully managed service that provides a serverless data integration platform for data preparation, data cataloging, and data loading. AWS Glue Studio is a graphical interface that allows you to easily author, run, and monitor AWS Glue ETL jobs. AWS Glue Data Quality is a feature that enables you to validate, cleanse, and enrich your data using predefined or custom rules. AWS Step Functions is a service that allows you to coordinate multiple AWS services into serverless workflows.

Using the Detect PII transform in AWS Glue Studio, you can automatically identify and label the PII in your dataset, such as names, addresses, phone numbers, email addresses, etc. You can then create a rule in AWS Glue Data Quality to obfuscate the PII, such as masking, hashing, or replacing the values with dummy data. You can also use other rules to validate and cleanse your data, such as checking for null values, duplicates, outliers, etc. You can then use an AWS Step Functions state machine to orchestrate a data pipeline to ingest the data into the S3 data lake. You can use AWS Glue DataBrew to visually explore and transform the data, AWS Glue crawlers to discover and catalog the data, and AWS Glue jobs to load the data into the S3 data lake.

This solution will meet the requirement with the least operational effort, as it leverages the serverless and managed capabilities of AWS Glue, AWS Glue Studio, AWS Glue Data Quality, and AWS Step Functions. You do not need to write any code to identify or obfuscate the PII, as you can use the built-in transforms and

rules in AWS Glue Studio and AWS Glue Data Quality. You also do not need to provision or manage any servers or clusters, as AWS Glue and AWS Step Functions scale automatically based on the demand. The other options are not as efficient as using the Detect PII transform in AWS Glue Studio, creating a rule in AWS Glue Data Quality, and using an AWS Step Functions state machine. Using an Amazon Kinesis Data Firehose delivery stream to process the dataset, creating an AWS Lambda transform function to identify the PII, using an AWS SDK to obfuscate the PII, and setting the S3 data lake as the target for the delivery stream will require more operational effort, as you will need to write and maintain code to identify and obfuscate the PII, as well as manage the Lambda function and its resources. Using the Detect PII transform in AWS Glue Studio to identify the PII, obfuscating the PII, and using an AWS Step Functions state machine to orchestrate a data pipeline to ingest the data into the S3 data lake will not be as effective as creating a rule in AWS Glue Data Quality to obfuscate the PII, as you will need to manually obfuscate the PII after identifying it, which can be error-prone and time-consuming. Ingesting the dataset into Amazon DynamoDB, creating an AWS Lambda function to identify and obfuscate the PII in the DynamoDB table and to transform the data, and using the same Lambda function to ingest the data into the S3 data lake will require more operational effort, as you will need to write and maintain code to identify and obfuscate the PII, as well as manage the Lambda function and its resources. You will also incur additional costs and complexity by using DynamoDB as an intermediate data store, which may not be necessary for your use case. References:

? AWS Glue

? AWS Glue Studio

? AWS Glue Data Quality

? [AWS Step Functions]

? [AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide], Chapter 6: Data Integration and Transformation, Section 6.1: AWS Glue

NEW QUESTION 10

A financial services company stores financial data in Amazon Redshift. A data engineer wants to run real-time queries on the financial data to support a web-based trading application. The data engineer wants to run the queries from within the trading application.

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

A. Establish WebSocket connections to Amazon Redshift.

B. Use the Amazon Redshift Data API.

C. Set up Java Database Connectivity (JDBC) connections to Amazon Redshift.

D. Store frequently accessed data in Amazon S3. Use Amazon S3 Select to run the queries.

Answer: B

Explanation:

The Amazon Redshift Data API is a built-in feature that allows you to run SQL queries on Amazon Redshift data with web services-based applications, such as AWS Lambda, Amazon SageMaker notebooks, and AWS Cloud9. The Data API does not require a persistent connection to your database, and it provides a secure HTTP endpoint and integration with AWS SDKs. You can use the endpoint to run SQL statements without managing connections. The Data API also supports both Amazon Redshift provisioned clusters and Redshift Serverless workgroups. The Data API is the best solution for running real-time queries on the financial data from within the trading application, as it has the least operational overhead compared to the other options.

Option A is not the best solution, as establishing WebSocket connections to Amazon Redshift would require more configuration and maintenance than using the Data API. WebSocket connections are also not supported by Amazon Redshift clusters or serverless workgroups.

Option C is not the best solution, as setting up JDBC connections to Amazon Redshift would also require more configuration and maintenance than using the Data API. JDBC connections are also not supported by Redshift Serverless workgroups.

Option D is not the best solution, as storing frequently accessed data in Amazon S3 and using Amazon S3 Select to run the queries would introduce additional latency and complexity than using the Data API. Amazon S3 Select is also not optimized for real-time queries, as it scans the entire object before returning the results. References:

? Using the Amazon Redshift Data API

? Calling the Data API

? Amazon Redshift Data API Reference

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

NEW QUESTION 14

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

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

A. Use Amazon Aurora for data storage

B. Use an Amazon Redshift provisioned cluster for data analysis.

C. Use Amazon S3 for data storage

D. Use Amazon Athena for data analysis.

E. Use AWS Glue DataBrew for centralized data governance and access control.

F. Use Amazon RDS for data storage

G. Use Amazon EMR for data analysis.

H. Use AWS Lake Formation for centralized data governance and access control.

Answer: BE

Explanation:

A data mesh is an architectural framework that organizes data into domains and treats data as products that are owned and offered for consumption by different teams¹. A data mesh requires a centralized layer for data governance and access control, as well as a distributed layer for data storage and analysis. AWS Glue can provide data catalogs and ETL operations for the data mesh, but it cannot provide data governance and access control by itself². Therefore, the company needs to use another AWS service for this purpose. AWS Lake Formation is a service that allows you to create, secure, and manage data lakes on AWS³. It integrates with AWS Glue and other AWS services to provide centralized data governance and access control for the data mesh. Therefore, option E is correct.

For data storage and analysis, the company can choose from different AWS services depending on their needs and preferences. However, one of the benefits of a data mesh is that it enables data to be stored and processed in a decoupled and scalable way¹. Therefore, using serverless or managed services that can handle large volumes and varieties of data is preferable. Amazon S3 is a highly scalable, durable, and secure object storage service that can store any type of data.

Amazon Athena is a serverless interactive query service that can analyze data in Amazon S3 using standard SQL. Therefore, option B is a good choice for data storage and analysis in a data mesh. Option A, C, and D are not optimal because they either use relational databases that are not suitable for storing diverse and unstructured data, or they require more management and provisioning than serverless services. References:

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

? 2: AWS Glue - Developer Guide

? 3: AWS Lake Formation - Features

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

? [5]: Amazon S3 - Features

? [6]: Amazon Athena - Features

NEW QUESTION 15

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

Which solution will meet these requirements MOST cost-effectively?

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

Answer: D

Explanation:

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

? AWS Glue

? Working with Python Shell Jobs

? pandas

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

NEW QUESTION 17

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

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

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

Answer: A

Explanation:

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

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

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

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

References:

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

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

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

NEW QUESTION 18

A company loads transaction data for each day into Amazon Redshift tables at the end of each day. The company wants to have the ability to track which tables have been loaded and which tables still need to be loaded.

A data engineer wants to store the load statuses of Redshift tables in an Amazon DynamoDB table. The data engineer creates an AWS Lambda function to publish the details of the load statuses to DynamoDB.

How should the data engineer invoke the Lambda function to write load statuses to the DynamoDB table?

- A. Use a second Lambda function to invoke the first Lambda function based on Amazon CloudWatch events.
- B. Use the Amazon Redshift Data API to publish an event to Amazon EventBridge
- C. Configure an EventBridge rule to invoke the Lambda function.
- D. Use the Amazon Redshift Data API to publish a message to an Amazon Simple Queue Service (Amazon SQS) queue
- E. Configure the SQS queue to invoke the Lambda function.
- F. Use a second Lambda function to invoke the first Lambda function based on AWS CloudTrail events.

Answer: B

Explanation:

The Amazon Redshift Data API enables you to interact with your Amazon Redshift data warehouse in an easy and secure way. You can use the Data API to run SQL commands, such as loading data into tables, without requiring a persistent connection to the cluster. The Data API also integrates with Amazon EventBridge, which allows you to monitor the execution status of your SQL commands and trigger actions based on events. By using the Data API to publish an event to EventBridge, the data engineer can invoke the Lambda function that writes the load statuses to the DynamoDB table. This solution is scalable, reliable, and cost-effective. The other options are either not possible or not optimal. You cannot use a second Lambda function to invoke the first Lambda function based on CloudWatch or CloudTrail events, as these services do not capture the load status of Redshift tables. You can use the Data API to publish a message to an SQS queue, but this would require additional configuration and polling logic to invoke the Lambda function from the queue. This would also introduce additional latency and cost. References:

? Using the Amazon Redshift Data API

? Using Amazon EventBridge with Amazon Redshift

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

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 no minimum 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 24

A company extracts approximately 1 TB of data every day from data sources such as SAP HANA, Microsoft SQL Server, MongoDB, Apache Kafka, and Amazon

DynamoDB. Some of the data sources have undefined data schemas or data schemas that change.

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

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

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

Answer: B

Explanation:

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

? AWS Glue

? [AWS Glue Data Catalog]

? [AWS Glue Developer Guide]

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

NEW QUESTION 27

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

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

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

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

Answer: B

Explanation:

AWS Database Migration Service (AWS DMS) is a cloud service that makes it possible to migrate relational databases, data warehouses, NoSQL databases, and other types of data stores to AWS quickly, securely, and with minimal downtime and zero data loss¹. AWS DMS supports migration between 20-plus database and analytics engines, such as Microsoft SQL Server to Amazon RDS for SQL Server². AWS DMS takes over many of the difficult or tedious tasks involved in a migration project, such as capacity analysis, hardware and software procurement, installation and administration, testing and debugging, and ongoing replication and monitoring¹. AWS DMS is a cost-effective solution, as you only pay for the compute resources and additional log storage used during the migration process². AWS DMS is the best solution for the company to migrate the financial transaction data from the on-premises Microsoft SQL Server database to AWS, as it meets the requirements of minimal downtime, zero data loss, and low cost.

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

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

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

References:

? Database Migration - AWS Database Migration Service - AWS

? What is AWS Database Migration Service?

? AWS Database Migration Service Documentation

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

NEW QUESTION 30

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

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

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

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

Answer: A

Explanation:

AWS Data Exchange is a service that makes it easy to find, subscribe to, and use third-party data in the cloud. It provides a secure and reliable way to access and

integrate data from various sources, such as data providers, public datasets, or AWS services. Using AWS Data Exchange, you can browse and subscribe to data products that suit your needs, and then use API calls or the AWS Management Console to export the data to Amazon S3, where you can use it with your existing analytics platform. This solution minimizes the effort and time required to incorporate third-party datasets, as you do not need to set up and manage data pipelines, storage, or access controls. You also benefit from the data quality and freshness provided by the data providers, who can update their data products as frequently as needed¹².

The other options are not optimal for the following reasons:

? B. Use API calls to access and integrate third-party datasets from AWS. This option is vague and does not specify which AWS service or feature is used to access and integrate third-party datasets. AWS offers a variety of services and features that can help with data ingestion, processing, and analysis, but not all of them are suitable for the given scenario. For example, AWS Glue is a serverless data integration service that can help you discover, prepare, and combine data from various sources, but it requires you to create and run data extraction, transformation, and loading (ETL) jobs, which can add operational overhead³.

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

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

References:

? 1: AWS Data Exchange User Guide

? 2: AWS Data Exchange FAQs

? 3: AWS Glue Developer Guide

? : AWS CodeCommit User Guide

? : Amazon Kinesis Data Streams Developer Guide

? : Amazon Elastic Container Registry User Guide

? : Build a Continuous Delivery Pipeline for Your Container Images with Amazon ECR as Source

NEW QUESTION 31

A company is migrating on-premises workloads to AWS. The company wants to reduce overall operational overhead. The company also wants to explore serverless options.

The company's current workloads use Apache Pig, Apache Oozie, Apache Spark, Apache Hbase, and Apache Flink. The on-premises workloads process petabytes of data in seconds. The company must maintain similar or better performance after the migration to AWS.

Which extract, transform, and load (ETL) service will meet these requirements?

- A. AWS Glue
- B. Amazon EMR
- C. AWS Lambda
- D. Amazon Redshift

Answer: A

Explanation:

AWS Glue is a fully managed serverless ETL service that can handle petabytes of data in seconds. AWS Glue can run Apache Spark and Apache Flink jobs without requiring any infrastructure provisioning or management. AWS Glue can also integrate with Apache Pig, Apache Oozie, and Apache Hbase using AWS Glue Data Catalog and AWS Glue workflows. AWS Glue can reduce the overall operational overhead by automating the data discovery, data preparation, and data loading processes. AWS Glue can also optimize the cost and performance of ETL jobs by using AWS Glue Job Bookmarking, AWS Glue Crawlers, and AWS Glue Schema Registry. References:

? AWS Glue

? AWS Glue Data Catalog

? AWS Glue Workflows

? [AWS Glue Job Bookmarking]

? [AWS Glue Crawlers]

? [AWS Glue Schema Registry]

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

NEW QUESTION 33

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

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

Which solution will meet these requirements?

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

Answer: D

Explanation:

Amazon Kinesis Data Streams is a service that enables you to collect, process, and analyze real-time streaming data. You can use Kinesis Data Streams to ingest data from various sources, such as Amazon CloudWatch Logs, and deliver it to different destinations, such as Amazon S3 or Amazon Redshift. To use Kinesis Data Streams to deliver the security logs from the production AWS account to the security AWS account, you need to create a destination data stream in the security AWS account. This data stream will receive the log data from the CloudWatch Logs service in the production AWS account. To enable this cross-account data delivery, you need to create an IAM role and a trust policy in the security AWS account. The IAM role defines the permissions that the CloudWatch

Logs service needs to put data into the destination data stream. The trust policy allows the production AWS account to assume the IAM role. Finally, you need to create a subscription filter in the production AWS account. A subscription filter defines the pattern to match log events and the destination to send the matching events. In this case, the destination is the destination data stream in the security AWS account. This solution meets the requirements of using Kinesis Data Streams to deliver the security logs to the security AWS account. The other options are either not possible or not optimal. You cannot create a destination data stream in the production AWS account, as this would not deliver the data to the security AWS account. You cannot create a subscription filter in the security AWS account, as this would not capture the log events from the production AWS account. References:

? Using Amazon Kinesis Data Streams with Amazon CloudWatch Logs

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

NEW QUESTION 34

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

Which solution will meet this requirement?

- A. Turn on concurrency scaling in workload management (WLM) for Redshift Serverless workgroups.
- B. Turn on concurrency scaling at the workload management (WLM) queue level in the Redshift cluster.
- C. Turn on concurrency scaling in the settings during the creation of a new Redshift cluster.
- D. Turn on concurrency scaling for the daily usage quota for the Redshift cluster.

Answer: B

Explanation:

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

References:

? Working with concurrency scaling - Amazon Redshift

? Amazon Redshift Concurrency Scaling - Amazon Web Services

? Configuring concurrency scaling queues - Amazon Redshift

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide (Chapter 6, page 163)

NEW QUESTION 38

A manufacturing company collects sensor data from its factory floor to monitor and enhance operational efficiency. The company uses Amazon Kinesis Data Streams to publish the data that the sensors collect to a data stream. Then Amazon Kinesis Data Firehose writes the data to an Amazon S3 bucket.

The company needs to display a real-time view of operational efficiency on a large screen in the manufacturing facility.

Which solution will meet these requirements with the LOWEST latency?

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

Answer: C

Explanation:

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

The other options are not optimal for the following reasons:

? A. Use Amazon Managed Service for Apache Flink (previously known as Amazon Kinesis Data Analytics) to process the sensor data. Use a connector for Apache Flink to write data to an Amazon Timestream database. Use the Timestream database as a source to create a Grafana dashboard. This option is similar to option C, but it uses Grafana instead of Amazon QuickSight to create the dashboard. Grafana is an open source visualization tool that can also connect to Amazon Timestream, but it requires additional steps to set up and configure, such as deploying a Grafana server on Amazon EC2, installing the Amazon Timestream plugin, and creating an IAM role for Grafana to access Timestream. These steps can increase the latency and complexity of the solution.

? B. Configure the S3 bucket to send a notification to an AWS Lambda function when any new object is created. Use the Lambda function to publish the data to Amazon Aurora. Use Aurora as a source to create an Amazon QuickSight dashboard. This option is not suitable for displaying a real-time view of operational efficiency, as it introduces unnecessary delays and costs in the data pipeline. First, the sensor data is written to an S3 bucket by Amazon Kinesis Data Firehose, which can have a buffering interval of up to 900 seconds. Then, the S3 bucket sends a notification to a Lambda function, which can incur additional invocation and execution time. Finally, the Lambda function publishes the data to Amazon Aurora, a relational database that is not optimized for time series data and can have higher storage and performance costs than Amazon Timestream.

? D. Use AWS Glue bookmarks to read sensor data from the S3 bucket in real time.

Publish the data to an Amazon Timestream database. Use the Timestream database as a source to create a Grafana dashboard. This option is also not suitable for displaying a real-time view of operational efficiency, as it uses AWS Glue bookmarks to read sensor data from the S3 bucket. AWS Glue bookmarks are a feature that helps AWS Glue jobs and crawlers keep track of the data that has already been processed, so that they can resume from where they left off. However, AWS Glue jobs and crawlers are not designed for real-time data processing, as they can have a minimum frequency of 5 minutes and a variable start-up time. Moreover, this option also uses Grafana instead of Amazon QuickSight to create the dashboard, which can increase the latency and complexity of the solution.

References:

? 1: Amazon Managed Streaming for Apache Flink

? 2: Amazon Timestream
? 3: Amazon QuickSight
? : Analyze data in Amazon Timestream using Grafana
? : Amazon Kinesis Data Firehose
? : Amazon Aurora
? : AWS Glue Bookmarks
? : AWS Glue Job and Crawler Scheduling

NEW QUESTION 42

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

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

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

- A. Configure an Amazon S3 Lifecycle policy to move data to the S3 Glacier Deep Archive storage class after 1 day
- B. Use the query result reuse feature of Amazon Athena for the SQL queries.
- C. Add an Amazon ElastiCache cluster between the BI application and Athena.
- D. Change the format of the files that are in the dataset to Apache Parquet.

Answer: B

Explanation:

The best solution to cost optimize the company's use of Amazon Athena without adding any additional infrastructure costs is to use the query result reuse feature of Amazon Athena for the SQL queries. This feature allows you to run the same query multiple times without incurring additional charges, as long as the underlying data has not changed and the query results are still in the query result location in Amazon S3¹. This feature is useful for scenarios where you have a petabyte-scale dataset that is updated infrequently, such as once a day, and you have a BI application that runs the same queries repeatedly, such as every hour. By using the query result reuse feature, you can reduce the amount of data scanned by your queries and save on the cost of running Athena. You can enable or disable this feature at the workgroup level or at the individual query level¹.

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

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

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

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

NEW QUESTION 43

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 48

A company uses an Amazon Redshift provisioned cluster as its database. The Redshift cluster has five reserved ra3.4xlarge nodes and uses key distribution.

A data engineer notices that one of the nodes frequently has a CPU load over 90%. SQL Queries that run on the node are queued. The other four nodes usually have a CPU load under 15% during daily operations.

The data engineer wants to maintain the current number of compute nodes. The data engineer also wants to balance the load more evenly across all five compute nodes.

Which solution will meet these requirements?

A. Change the sort key to be the data column that is most often used in a WHERE clause of the SQL SELECT statement.

B. Change the distribution key to the table column that has the largest dimension.

C. Upgrade the reserved node from ra3.4xlarge to ra3.16xlarge.

D. Change the primary key to be the data column that is most often used in a WHERE clause of the SQL SELECT statement.

Answer: B

Explanation:

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

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

The primary key is a constraint that enforces the uniqueness of the rows in a table, but it does not influence the data layout or the query optimization. References:

? Choosing a data distribution style

? Choosing a data sort key

? Working with primary keys

NEW QUESTION 52

A company maintains multiple extract, transform, and load (ETL) workflows that ingest data from the company's operational databases into an Amazon S3 based data lake. The ETL workflows use AWS Glue and Amazon EMR to process data.

The company wants to improve the existing architecture to provide automated orchestration and to require minimal manual effort.

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

A. AWS Glue workflows

B. AWS Step Functions tasks

C. AWS Lambda functions

D. Amazon Managed Workflows for Apache Airflow (Amazon MWAA) workflows

Answer: A

Explanation:

AWS Glue workflows are a feature of AWS Glue that enable you to create and visualize complex ETL pipelines using AWS Glue components, such as crawlers, jobs, triggers, and development endpoints. AWS Glue workflows provide automated orchestration and require minimal manual effort, as they handle dependency resolution, error handling, state management, and resource allocation for your ETL workflows. You can use AWS Glue workflows to ingest data from your operational databases into your Amazon S3 based data lake, and then use AWS Glue and Amazon EMR to process the data in the data lake. This solution will meet the requirements with the least operational overhead, as it leverages the serverless and fully managed nature of AWS Glue, and the scalability and flexibility of Amazon EMR.

The other options are not optimal for the following reasons:

? B. AWS Step Functions tasks. AWS Step Functions is a service that lets you coordinate multiple AWS services into serverless workflows. You can use AWS Step Functions tasks to invoke AWS Glue and Amazon EMR jobs as part of your ETL workflows, and use AWS Step Functions state machines to define the logic and flow of your workflows. However, this option would require more manual effort than AWS Glue workflows, as you would need to write JSON code to define your state machines, handle errors and retries, and monitor the execution history and status of your workflows.

? C. AWS Lambda functions. AWS Lambda is a service that lets you run code without provisioning or managing servers. You can use AWS Lambda functions to trigger AWS Glue and Amazon EMR jobs as part of your ETL workflows, and use AWS Lambda event sources and destinations to orchestrate the flow of your workflows. However, this option would also require more manual effort than AWS Glue workflows, as you would need to write code to implement your business logic, handle errors and retries, and monitor the invocation and execution of your Lambda functions. Moreover, AWS Lambda functions have limitations on the execution time, memory, and concurrency, which may affect the performance and scalability of your ETL workflows.

? D. Amazon Managed Workflows for Apache Airflow (Amazon MWAA) workflows.

Amazon MWAA is a managed service that makes it easy to run open source Apache Airflow on AWS. Apache Airflow is a popular tool for creating and managing complex ETL pipelines using directed acyclic graphs (DAGs). You can use Amazon MWAA workflows to orchestrate AWS Glue and Amazon EMR jobs as part of your ETL workflows, and use the Airflow web interface to visualize and monitor your workflows. However, this option would have more operational overhead than AWS Glue workflows, as you would need to set up and configure your Amazon MWAA environment, write Python code to define your DAGs, and manage the dependencies and versions of your Airflow plugins and operators.

References:

? 1: AWS Glue Workflows

? 2: AWS Glue and Amazon EMR

? 3: AWS Step Functions

? : AWS Lambda

? : Amazon Managed Workflows for Apache Airflow

NEW QUESTION 53

A company needs to build a data lake in AWS. The company must provide row-level data access and column-level data access to specific teams. The teams will access the data by using Amazon Athena, Amazon Redshift Spectrum, and Apache Hive from Amazon EMR.

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

- A. Use Amazon S3 for data lake storag
- B. Use S3 access policies to restrict data access by rows and column
- C. Provide data access throughAmazon S3.
- D. Use Amazon S3 for data lake storag
- E. Use Apache Ranger through Amazon EMR to restrict data access byrows and column
- F. Providedata access by using Apache Pig.
- G. Use Amazon Redshift for data lake storag
- H. Use Redshift security policies to restrict data access byrows and column
- I. Provide data accessby usingApache Spark and Amazon Athena federated queries.
- J. UseAmazon S3 for data lake storag
- K. Use AWS Lake Formation to restrict data access by rows and column
- L. Provide data access through AWS Lake Formation.

Answer: D

Explanation:

Option D is the best solution to meet the requirements with the least operational overhead because AWS Lake Formation is a fully managed service that simplifies the process of building, securing, and managing data lakes. AWS Lake Formation allows you to define granular data access policies at the row and column level for different users and groups. AWS Lake Formation also integrates with Amazon Athena, Amazon Redshift Spectrum, and Apache Hive on Amazon EMR, enabling these services to access the data in the data lake through AWS Lake Formation.

Option A is not a good solution because S3 access policies cannot restrict data access by rows and columns. S3 access policies are based on the identity and permissions of the requester, the bucket and object ownership, and the object prefix and tags. S3 access policies cannot enforce fine-grained data access control at the row and column level. Option B is not a good solution because it involves using Apache Ranger and Apache Pig, which are not fully managed services and require additional configuration and maintenance. Apache Ranger is a framework that provides centralized security administration for data stored in Hadoop clusters, such as Amazon EMR. Apache Ranger can enforce row-level and column-level access policies for Apache Hive tables. However, Apache Ranger is not a native AWS service and requires manual installation and configuration on Amazon EMR clusters. Apache Pig is a platform that allows you to analyze large data sets using a high-level scripting language called Pig Latin. Apache Pig can access data stored in Amazon S3 and process it using Apache Hive. However, Apache Pig is not a native AWS service and requires manual installation and configuration on Amazon EMR clusters.

Option C is not a good solution because Amazon Redshift is not a suitable service for data lake storage. Amazon Redshift is a fully managed data warehouse service that allows you to run complex analytical queries using standard SQL. Amazon Redshift can enforce row- level and column-level access policies for different users and groups. However, Amazon Redshift is not designed to store and process large volumes of unstructured or semi- structured data, which are typical characteristics of data lakes. Amazon Redshift is also more expensive and less scalable than Amazon S3 for data lake storage.

References:

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

? What Is AWS Lake Formation? - AWS Lake Formation

? Using AWS Lake Formation with Amazon Athena - AWS Lake Formation

? Using AWS Lake Formation with Amazon Redshift Spectrum - AWS Lake Formation

? Using AWS Lake Formation with Apache Hive on Amazon EMR - AWS Lake Formation

? Using Bucket Policies and User Policies - Amazon Simple Storage Service

? Apache Ranger

? Apache Pig

? What Is Amazon Redshift? - Amazon Redshift

NEW QUESTION 58

A company is migrating a legacy application to an Amazon S3 based data lake. A data engineer reviewed data that is associated with the legacy application. The data engineer found that the legacy data contained some duplicate information.

The data engineer must identify and remove duplicate information from the legacy application data.

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

- A. Write a custom extract, transform, and load (ETL) job in Pytho
- B. Use the DataFramedrop duplicatesf) function by importingthe Pandas library to perform data deduplication.
- C. Write an AWS Glue extract, transform, and load (ETL) jo
- D. Usethe FindMatches machine learning(ML) transform to transform the data to perform data deduplication.
- E. Write a custom extract, transform, and load (ETL) job in Pytho
- F. Import the Python dedupe librar
- G. Use the dedupe library to perform data deduplication.
- H. Write an AWS Glue extract, transform, and load (ETL) jo
- I. Import the Python dedupe librar
- J. Use the dedupe library to perform data deduplication.

Answer: B

Explanation:

AWS Glue is a fully managed serverless ETL service that can handle data deduplication with minimal operational overhead. AWS Glue provides a built-in ML transform called FindMatches, which can automatically identify and group similar records in a dataset. FindMatches can also generate a primary key for each group of records and remove duplicates. FindMatches does not require any coding or prior ML experience, as it can learn from a sample of labeled data provided by the user. FindMatches can also scale to handle large datasets and optimize the cost and performance of the ETL job. References:

? AWS Glue

? FindMatches ML Transform

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

NEW QUESTION 59

A healthcare company uses Amazon Kinesis Data Streams to stream real-time health data from wearable devices, hospital equipment, and patient records.

A data engineer needs to find a solution to process the streaming data. The data engineer needs to store the data in an Amazon Redshift Serverless warehouse. The solution must support near real-time analytics of the streaming data and the previous day's data. Which solution will meet these requirements with the LEAST operational overhead?

- A. Load data into Amazon Kinesis Data Firehose
- B. Load the data into Amazon Redshift.
- C. Use the streaming ingestion feature of Amazon Redshift.
- D. Load the data into Amazon S3. Use the COPY command to load the data into Amazon Redshift.
- E. Use the Amazon Aurora zero-ETL integration with Amazon Redshift.

Answer: B

Explanation:

The streaming ingestion feature of Amazon Redshift enables you to ingest data from streaming sources, such as Amazon Kinesis Data Streams, into Amazon Redshift tables in near real-time. You can use the streaming ingestion feature to process the streaming data from the wearable devices, hospital equipment, and patient records. The streaming ingestion feature also supports incremental updates, which means you can append new data or update existing data in the Amazon Redshift tables. This way, you can store the data in an Amazon Redshift Serverless warehouse and support near real-time analytics of the streaming data and the previous day's data. This solution meets the requirements with the least operational overhead, as it does not require any additional services or components to ingest and process the streaming data. The other options are either not feasible or not optimal. Loading data into Amazon Kinesis Data Firehose and then into Amazon Redshift (option A) would introduce additional latency and cost, as well as require additional configuration and management. Loading data into Amazon S3 and then using the COPY command to load the data into Amazon Redshift (option C) would also introduce additional latency and cost, as well as require additional storage space and ETL logic. Using the Amazon Aurora zero-ETL integration with Amazon Redshift (option D) would not work, as it requires the data to be stored in Amazon Aurora first, which is not the case for the streaming data from the healthcare company. References:

? Using streaming ingestion with Amazon Redshift

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 3: Data Ingestion and Transformation, Section 3.5: Amazon Redshift Streaming Ingestion

NEW QUESTION 61

A company has five offices in different AWS Regions. Each office has its own human resources (HR) department that uses a unique IAM role. The company stores employee records in a data lake that is based on Amazon S3 storage.

A data engineering team needs to limit access to the records. Each HR department should be able to access records for only employees who are within the HR department's Region.

Which combination of steps should the data engineering team take to meet this requirement with the LEAST operational overhead? (Choose two.)

- A. Use data filters for each Region to register the S3 paths as data locations.
- B. Register the S3 path as an AWS Lake Formation location.
- C. Modify the IAM roles of the HR departments to add a data filter for each department's Region.
- D. Enable fine-grained access control in AWS Lake Formation
- E. Add a data filter for each Region.
- F. Create a separate S3 bucket for each Region
- G. Configure an IAM policy to allow S3 access
- H. Restrict access based on Region.

Answer: BD

Explanation:

AWS Lake Formation is a service that helps you build, secure, and manage data lakes on Amazon S3. You can use AWS Lake Formation to register the S3 path as a data lake location, and enable fine-grained access control to limit access to the records based on the HR department's Region. You can use data filters to specify which S3 prefixes or partitions each HR department can access, and grant permissions to the IAM roles of the HR departments accordingly. This solution will meet the requirement with the least operational overhead, as it simplifies the data lake management and security, and leverages the existing IAM roles of the HR departments.

The other options are not optimal for the following reasons:

? A. Use data filters for each Region to register the S3 paths as data locations. This option is not possible, as data filters are not used to register S3 paths as data locations, but to grant permissions to access specific S3 prefixes or partitions within a data location. Moreover, this option does not specify how to limit access to the records based on the HR department's Region.

? C. Modify the IAM roles of the HR departments to add a data filter for each department's Region. This option is not possible, as data filters are not added to IAM roles, but to permissions granted by AWS Lake Formation. Moreover, this option does not specify how to register the S3 path as a data lake location, or how to enable fine-grained access control in AWS Lake Formation.

? E. Create a separate S3 bucket for each Region. Configure an IAM policy to allow S3 access. Restrict access based on Region. This option is not recommended, as it would require more operational overhead to create and manage multiple S3 buckets, and to configure and maintain IAM policies for each HR department. Moreover, this option does not leverage the benefits of AWS Lake Formation, such as data cataloging, data transformation, and data governance.

References:

? 1: AWS Lake Formation

? 2: AWS Lake Formation Permissions

? : AWS Identity and Access Management

? : Amazon S3

NEW QUESTION 64

A company is building an analytics solution. The solution uses Amazon S3 for data lake storage and Amazon Redshift for a data warehouse. The company wants to use Amazon Redshift Spectrum to query the data that is in Amazon S3.

Which actions will provide the FASTEST queries? (Choose two.)

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

Answer: BC

Explanation:

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

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

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

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

? Amazon Redshift Spectrum

? Choosing the Right Data Format

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

NEW QUESTION 69

A company has multiple applications that use datasets that are stored in an Amazon S3 bucket. The company has an ecommerce application that generates a dataset that contains personally identifiable information (PII). The company has an internal analytics application that does not require access to the PII.

To comply with regulations, the company must not share PII unnecessarily. A data engineer needs to implement a solution that with redact PII dynamically, based on the needs of each application that accesses the dataset.

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

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

Answer: B

Explanation:

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

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

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

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

References:

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

? Introducing Amazon S3 Object Lambda – Use Your Code to Process Data as It Is Being Retrieved from S3

? Using Bucket Policies and User Policies - Amazon Simple Storage Service

? AWS Glue Documentation

? What is Amazon API Gateway? - Amazon API Gateway

NEW QUESTION 72

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

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

Answer: B

Explanation:

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

? AWS Lake Formation

? Row-Level Security

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

NEW QUESTION 74

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

The data engineer's original query is as follows: `SELECT product_name, sum(sales_amount) FROM sales_data`

`WHERE year = 2023`

`GROUP BY product_name`

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

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

Answer: B

Explanation:

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

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

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

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

NEW QUESTION 78

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

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

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

Answer: C

Explanation:

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

? Using the AWS Glue Data Catalog as the metastore for Hive (Section: Specifying

AWS Glue Data Catalog as the metastore)

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

NEW QUESTION 80

A company stores data in a data lake that is in Amazon S3. Some data that the company stores in the data lake contains personally identifiable information (PII). Multiple user groups need to access the raw data. The company must ensure that user groups can access only the PII that they require. Which solution will meet these requirements with the LEAST effort?

- A. Use Amazon Athena to query the dat
- B. Set up AWS Lake Formation and create data filters to establish levels of access for the company's IAM role
- C. Assign each user to the IAM role that matches the user's PII access requirements.
- D. Use Amazon QuickSight to access the dat
- E. Use column-level security features in QuickSight to limit the PII that users can retrieve from Amazon S3 by using Amazon Athen
- F. Define QuickSight access levels based on the PII access requirements of the users.
- G. Build a custom query builder UI that will run Athena queries in the background to access the dat
- H. Create user groups in Amazon Cognit
- I. Assign access levels to the user groups based on the PII access requirements of the users.
- J. Create IAM roles that have different levels of granular acces
- K. Assign the IAM roles to IAM user group
- L. Use an identity-based policy to assign access levels to user groups at the column level.

Answer: A

Explanation:

Amazon Athena is a serverless, interactive query service that enables you to analyze data in Amazon S3 using standard SQL. AWS Lake Formation is a service that helps you build, secure, and manage data lakes on AWS. You can use AWS Lake Formation to create data filters that define the level of access for different IAM roles based on the columns, rows, or tags of the data. By using Amazon Athena to query the data and AWS Lake Formation to create data filters, the company can meet the requirements of ensuring that user groups can access only the PII that they require with the least effort. The solution is to use Amazon Athena to query the data in the data lake that is in Amazon S3. Then, set up AWS Lake Formation and create data filters to establish levels of access for the company's IAM roles. For example, a data filter can allow a user group to access only the columns that contain the PII that they need, such as name and email address, and deny access to the columns that contain the PII that they do not need, such as phone number and social security number. Finally, assign each user to the IAM role that matches the user's PII access requirements. This way, the user groups can access the data in the data lake securely and efficiently. The other options are either not feasible or not optimal. Using Amazon QuickSight to access the data (option B) would require the company to pay for the QuickSight service and to configure the column-level security features for each user. Building a custom query builder UI that will run Athena queries in the background to access the data (option C) would require the company to develop and maintain the UI and to integrate it with Amazon Cognito. Creating IAM roles that have different levels of granular access (option D) would require the company to manage multiple IAM roles and policies and to ensure that they are aligned with the data schema.

References:

- ? Amazon Athena
- ? AWS Lake Formation
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 4: Data Analysis and Visualization, Section 4.3: Amazon Athena

NEW QUESTION 82

.....

THANKS FOR TRYING THE DEMO OF OUR PRODUCT

Visit Our Site to Purchase the Full Set of Actual AWS-Certified-Data-Engineer-Associate Exam Questions With Answers.

We Also Provide Practice Exam Software That Simulates Real Exam Environment And Has Many Self-Assessment Features. Order the AWS-Certified-Data-Engineer-Associate Product From:

<https://www.2passeasy.com/dumps/AWS-Certified-Data-Engineer-Associate/>

Money Back Guarantee

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

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