# AWS DevOps Engineer Professional Practice Test (Sample)

**Study Guide**

BY EXAMZIFY

**Everything you need from our exam experts!**

## Questions

1. **What type of database is Amazon DynamoDB?**
   - **A. A relational database service**
   - **B. A fully managed NoSQL database service**
   - **C. An in-memory database service**
   - **D. A data warehousing service**

2. **What is required to trigger a CodeBuild build daily at a specific time?**
   - **A. Create a Lambda function**
   - **B. Configure a scheduled CloudWatch Event Rule**
   - **C. Set a cron job on an EC2 instance**
   - **D. Use AWS Step Functions**

3. **What is required to allow for passing queue messages that are 1GB in size?**
   - **A. Use S3 as a storage mechanism**
   - **B. Use the SQS Extended Client Library**
   - **C. Both of the above**
   - **D. Limit messages to 256KB**

4. **What is the main advantage of using Blue/Green deployments in AWS?**
   - **A. Shorter downtime during updates**
   - **B. Better cost management**
   - **C. Improved security features**
   - **D. Automatic scaling of resources**

5. **What extension must configuration files for Elastic Beanstalk extensions use?**
   - **A. .json**
   - **B. .config**
   - **C. .yml**
   - **D. .txt**

6. What is the most cost-effective way to utilize Jenkins with AWS CodeBuild?
   - A. Use a single Jenkins master instance
   - B. Deploy Jenkins in a multi-master setup across multiple Availability Zones
   - C. Run Jenkins in a Docker container
   - D. Implement Jenkins on-premise and connect via VPN

7. What is the main benefit of using Amazon Elastic Beanstalk?
   - A. Enhanced security features
   - B. Simplified application deployment and management
   - C. Increased storage capacity
   - D. Better cost control for billing

8. What service is used to automate the deployment of applications in AWS?
   - A. AWS CodeBuild
   - B. AWS Lambda
   - C. AWS CodeDeploy
   - D. Amazon EKS

9. What should be prioritized when designing a partition key for a DynamoDB table?
   - A. Using a complex data structure
   - B. Ensuring data is evenly distributed based on access patterns
   - C. Selecting a single User ID as the only key
   - D. Avoiding the use of partition keys entirely

10. How can you create notifications for CodeBuild events?
   - A. Use Amazon SNS along with CloudWatch Events
   - B. Directly through the CodeBuild console
   - C. With AWS CloudTrail logs
   - D. By sending emails through SES

## Answers

**1. B**
**2. B**
**3. C**
**4. A**
**5. B**
**6. B**
**7. B**
**8. C**
**9. B**
**10. A**

## Explanations

## 1. What type of database is Amazon DynamoDB?

**A. A relational database service**

**B. A fully managed NoSQL database service**

**C. An in-memory database service**

**D. A data warehousing service**

**Amazon DynamoDB is a fully managed NoSQL database service designed for high performance and scalability. It provides seamless scalability, allowing you to handle large amounts of data and high-traffic workloads without the need for complex database administration. This NoSQL database is ideal for applications that require consistent, single-digit millisecond latency at any scale.  DynamoDB's architecture is built to support both document and key-value data models, making it highly flexible for various application requirements. By abstracting away the underlying infrastructure management tasks such as hardware provisioning, setup, and configuration, DynamoDB enables developers to focus on building and optimizing their applications instead of managing database servers.  This differentiates it significantly from relational database services, which rely on structured query language (SQL) and predefined schemas, and from in-memory database services, which prioritize data storage in RAM for faster access but do not provide the persistent storage features of DynamoDB. Additionally, it’s not a data warehousing service, which specializes in analytical workloads rather than transactional operations that DynamoDB is focused on.**

## 2. What is required to trigger a CodeBuild build daily at a specific time?

**A. Create a Lambda function**

**B. Configure a scheduled CloudWatch Event Rule**

**C. Set a cron job on an EC2 instance**

**D. Use AWS Step Functions**

**To trigger a CodeBuild build daily at a specific time, configuring a scheduled CloudWatch Event Rule is the correct approach. CloudWatch Events (now part of Amazon EventBridge) allows you to set rules that can initiate actions based on a defined schedule, using cron or rate expressions.   By creating a CloudWatch Events Rule with a cron expression, you can specify exactly when the event should trigger, including the time of day and the frequency (for example, daily). Once the rule is set up, it can target the CodeBuild project, automatically starting a new build at the defined time.  Utilizing a Lambda function, while potentially useful for other operations, would require additional setups, such as creating and managing the Lambda function’s execution role, and manually invoking the build process, making it more complex than necessary for a simple scheduled build.   Setting a cron job on an EC2 instance is not a serverless solution and requires maintaining an EC2 instance, which is not ideal when a fully managed service like CloudWatch Events can handle the task.  AWS Step Functions are designed for orchestrating multiple AWS services into serverless workflows and would be overkill for a simple task of scheduling a build in CodeBuild. They are more suited for complex workflows rather than**

## 3. What is required to allow for passing queue messages that are 1GB in size?

**A. Use S3 as a storage mechanism**

**B. Use the SQS Extended Client Library**

**C. Both of the above**

**D. Limit messages to 256KB**

**To facilitate the passing of queue messages that are 1GB in size, both utilizing S3 as a storage mechanism and employing the SQS Extended Client Library is necessary. The SQS Extended Client Library allows you to use Amazon Simple Queue Service (SQS) for message queuing, while storing the payload of larger messages, like 1GB, in Amazon S3. This approach is essential because SQS natively supports a maximum message size of 256KB. By leveraging the SQS Extended Client Library, you can send and receive messages where the message body is stored in S3, and SQS only stores a reference to that data. The library handles the complexity of integrating SQS with S3, enabling applications to work with larger payloads seamlessly. Using S3 as a storage mechanism specifically addresses the limitations of message size in SQS. Since S3 can handle very large files, it becomes an effective solution for storing content that exceeds the direct limitations imposed by SQS message size constraints. Therefore, both mechanisms are integral to successfully managing large messages, making it essential to implement both the Extended Client Library and S3 together for effective message delivery in these scenarios.**

## 4. What is the main advantage of using Blue/Green deployments in AWS?

**A. Shorter downtime during updates**

**B. Better cost management**

**C. Improved security features**

**D. Automatic scaling of resources**

**The primary advantage of using Blue/Green deployments in AWS is the reduction of downtime during updates. This deployment strategy involves maintaining two identical environments, referred to as the "Blue" environment (the current production environment) and the "Green" environment (the new version with the update). When the new version is ready for deployment, it is launched in the Green environment, while the Blue environment remains fully operational and serving user traffic. Once the Green environment is successfully tested and validated, traffic can be seamlessly routed from Blue to Green with a simple change in configuration. This rapid switch significantly minimizes any potential downtime that users may experience during the deployment process. In the event of issues or bugs, it is easy to roll back to the Blue environment by redirecting traffic back, ensuring application availability. While the other options mention factors such as cost management, security features, and automatic scaling, these are not the primary benefits of the Blue/Green deployment approach. The focus here is primarily on deploying updates with high availability and minimal service interruption, highlighting the advantage of shorter downtime during updates.**

## 5. What extension must configuration files for Elastic Beanstalk extensions use?

A. .json

B. .config

C. .yml

D. .txt

**Configuration files for Elastic Beanstalk extensions specifically utilize the .config extension. This is a requirement established by Elastic Beanstalk to define configurations for environments, including settings for software configurations, environment properties, and resource management. When you place a .config file in the .ebextensions directory of your application source bundle, Elastic Beanstalk processes it during the deployment phase, allowing you to customize and modify the environment settings to suit your application's needs.  The other file types listed—like .json, .yml, and .txt—do not align with Elastic Beanstalk's specifications for configuration settings. While .json and .yml are commonly used for various configuration files in AWS and other tools, they are not the designated format for Elastic Beanstalk extensions. Similarly, .txt files are plain text and do not serve the purpose of defining structured configurations that Elastic Beanstalk requires. This specificity helps streamline the deployment process and ensures that configuration settings are interpreted correctly by the platform.**

## 6. What is the most cost-effective way to utilize Jenkins with AWS CodeBuild?

A. Use a single Jenkins master instance

B. Deploy Jenkins in a multi-master setup across multiple Availability Zones

C. Run Jenkins in a Docker container

D. Implement Jenkins on-premise and connect via VPN

**The most cost-effective way to utilize Jenkins with AWS CodeBuild is to run Jenkins in a Docker container. This approach benefits from enhanced resource utilization and minimizes overhead costs associated with maintaining dedicated infrastructure. By leveraging containers, you can easily scale Jenkins based on demand, allowing for efficient resource management and reduced costs. Containerized applications can also be quickly deployed, updated, and torn down, providing significant operational flexibility.  Using a single Jenkins master instance is less cost-effective because it may become a bottleneck as demand increases. It also lacks redundancy in case of failures, potentially leading to downtime and additional costs associated with recovery.  Deploying Jenkins in a multi-master setup across multiple Availability Zones, while it enhances redundancy and availability, incurs additional costs for the infrastructure and requires more management overhead, making it less suitable for cost-effective implementations.  Implementing Jenkins on-premise and connecting via VPN could lead to high costs due to hardware, maintenance, and network setup requirements. This configuration also inherently lacks the scalability and flexibility provided by cloud services like AWS.  In contrast, running Jenkins in a Docker container not only optimizes resource usage but aligns well with cloud-based deployment strategies, making it the most cost-effective choice for integrating Jenkins with AWS CodeBuild.**

## 7. What is the main benefit of using Amazon Elastic Beanstalk?

A. Enhanced security features

B. Simplified application deployment and management

C. Increased storage capacity

D. Better cost control for billing

The primary benefit of using Amazon Elastic Beanstalk is its ability to simplify application deployment and management. Elastic Beanstalk is a Platform as a Service (PaaS) that allows developers to deploy and manage applications without needing to worry about the underlying infrastructure. This service automates many of the complexity involved in deploying applications, including load balancing, scaling, and monitoring.  With Elastic Beanstalk, developers can focus on writing code instead of managing servers and configurations, which significantly accelerates the deployment process. It supports multiple programming languages and frameworks, making it versatile for different development environments. The platform automatically handles the deployment details such as provisioning capacity, deploying the application, scaling up or down based on demand, and monitoring the application health.  This ease of use and automation helps organizations streamline their development and delivery processes, reducing the time to market for new features and applications.

## 8. What service is used to automate the deployment of applications in AWS?

A. AWS CodeBuild

B. AWS Lambda

C. AWS CodeDeploy

D. Amazon EKS

The use of AWS CodeDeploy is critical for automating the deployment of applications in AWS. This service enables developers to deploy application code to a variety of compute services, including Amazon EC2 instances, AWS Fargate, and Lambda functions, without the need for manual intervention. It supports the deployment of updates to one or multiple instances, can automatically roll back changes if there are issues, and integrates seamlessly with other AWS services, making it an essential tool for continuous delivery and deployment in DevOps practices.  By utilizing features like in-place and blue/green deployments, CodeDeploy ensures minimal downtime during application updates and allows for easier management of application versions. This enhances operational efficiency and reliability, ensuring that deployments are consistent and repeatable across development, staging, and production environments.

## 9. What should be prioritized when designing a partition key for a DynamoDB table?

A. Using a complex data structure

**B. Ensuring data is evenly distributed based on access patterns**

C. Selecting a single User ID as the only key

D. Avoiding the use of partition keys entirely

Prioritizing the even distribution of data based on access patterns is crucial when designing a partition key for a DynamoDB table. This is because DynamoDB is designed for scalability and high performance, and evenly distributing data across partitions helps achieve this goal. If data is not evenly distributed, certain partitions can become hotspots, leading to throttling and performance issues. By choosing a partition key that allows for balanced access, you ensure that read and write operations can be handled efficiently across the underlying storage infrastructure. This approach supports optimal performance and scales with growth in both the amount of data and the number of requests. In contrast to the other options, using a complex data structure can complicate access patterns and hinder performance. Selecting a single User ID as the only key may lead to uneven data distribution, particularly if many operations are focused on a limited number of users. Lastly, avoiding the use of partition keys entirely goes against the design principles of DynamoDB, as partition keys are fundamental to how data is organized and accessed in the table. Thus, prioritizing even data distribution based on access patterns is essential for effective DynamoDB design.

## 10. How can you create notifications for CodeBuild events?

**A. Use Amazon SNS along with CloudWatch Events**

B. Directly through the CodeBuild console

C. With AWS CloudTrail logs

D. By sending emails through SES

Creating notifications for CodeBuild events is effectively achieved by using Amazon Simple Notification Service (SNS) in conjunction with CloudWatch Events. This approach leverages the event-driven architecture of AWS, allowing you to respond to various events that occur in CodeBuild, such as the start and completion of build projects. When CodeBuild generates events, these can be captured and sent to CloudWatch Events, which then routes the event to an SNS topic. By subscribing to this SNS topic, you can receive notifications through various protocols, including SMS, email, or even HTTP endpoints. This method provides a versatile and scalable solution for monitoring build statuses and results across different AWS services. Other options, while perhaps feasible in some contexts, do not provide the same level of automation or integration. For instance, the CodeBuild console allows users to see build progress and results but does not facilitate automated notifications based on events. AWS CloudTrail logs record API calls, but they are primarily for auditing and not for immediate notifications about build status. Finally, sending emails through SES is a more manual and less integrated approach compared to the automation and event-driven nature of SNS with CloudWatch Events.