Docker Certified Associate (DCA) Certification Practice Test (Sample)

Study Guide

Everything you need from our exam experts!

Table of Contents

Introduction

Preparing for a certification exam can feel overwhelming, but with the right tools, it becomes an opportunity to build confidence, sharpen your skills, and move one step closer to your goals. At Examzify, we believe that effective exam preparation isn't just about memorization, it's about understanding the material, identifying knowledge gaps, and building the test-taking strategies that lead to success.

This guide was designed to help you do exactly that.

Whether you're preparing for a licensing exam, professional certification, or entry-level qualification, this book offers structured practice to reinforce key concepts. You'll find a wide range of multiple-choice questions, each followed by clear explanations to help you understand not just the right answer, but why it's correct.

The content in this guide is based on real-world exam objectives and aligned with the types of questions and topics commonly found on official tests. It's ideal for learners who want to:

- **Practice answering questions under realistic conditions,**
- **Improve accuracy and speed,**
- **Review explanations to strengthen weak areas, and**
- **Approach the exam with greater confidence.**

We recommend using this book not as a stand-alone study tool, but alongside other resources like flashcards, textbooks, or hands-on training. For best results, we recommend working through each question, reflecting on the explanation provided, and revisiting the topics that challenge you most.

Remember: successful test preparation isn't about getting every question right the first time, it's about learning from your mistakes and improving over time. Stay focused, trust the process, and know that every page you turn brings you closer to success.

Let's begin.

How to Use This Guide

This guide is designed to help you study more effectively and approach your exam with confidence. Whether you're reviewing for the first time or doing a final refresh, here’s how to get the most out of your Examzify study guide:

1. Start with a Diagnostic Review

Skim through the questions to get a sense of what you know and what you need to focus on. Your goal is to identify knowledge gaps early.

2. Study in Short, Focused Sessions

Break your study time into manageable blocks (e.g. 30 – 45 minutes). Review a handful of questions, reflect on the explanations.

3. Learn from the Explanations

After answering a question, always read the explanation, even if you got it right. It reinforces key points, corrects misunderstandings, and teaches subtle distinctions between similar answers.

4. Track Your Progress

Use bookmarks or notes (if reading digitally) to mark difficult questions. Revisit these regularly and track improvements over time.

5. Simulate the Real Exam

Once you're comfortable, try taking a full set of questions without pausing. Set a timer and simulate test-day conditions to build confidence and time management skills.

6. Repeat and Review

Don’t just study once, repetition builds retention. Re-attempt questions after a few days and revisit explanations to reinforce learning. Pair this guide with other Examzify tools like flashcards, and digital practice tests to strengthen your preparation across formats.

There’s no single right way to study, but consistent, thoughtful effort always wins. Use this guide flexibly, adapt the tips above to fit your pace and learning style. You've got this!

Questions

1. What is the role of a Dockerfile?

A. To log events

B. To define the build instructions for an image

C. To perform system checks

D. To manage container data volumes

2. What detail does 'docker network inspect' provide?

A. High-level overview of all containers

B. Resource usage of each container

C. Detailed information about a specified network

D. Logs from container execution

3. Which is a secure method for allowing a Docker client to authenticate with a registry that uses a self-signed certificate?

A. Configuring the Docker daemon to allow anonymous access

B. Adding the self-signed certificate to /etc/docker/certs.d/

C. Running Docker with elevated privileges

D. Using an external authentication service

4. Which command is used to remove all stopped containers?

A. docker rm --all

B. docker container prune

C. docker stop all

D. docker delete --stopped

5. Why is Docker preferred over traditional virtual machines for isolating applications?

A. Docker containers are heavier and slower than VMs

B. Docker containers share the host OS kernel

C. Docker requires more system resources than VMs

D. Docker is not compatible with microservices

6. What does 'docker stats' do?

- **A. Shows container network configurations**
- **B. Displays real-time resource usage statistics for Docker containers**
- **C. Deletes stopped containers**
- **D. Shows the list of images on the host**

7. Which of the following best describes the relationship between Docker and cloud services?

- **A. Docker is incompatible with cloud services**
- **B. Docker is used mainly for local development and has no cloud capabilities**
- **C. Docker can run on cloud infrastructure to enhance deployability**
- **D. Docker eliminates the need for any cloud services**

8. What command is used to build a Docker image?

- **A. docker create**
- **B. docker image build**
- **C. docker build**
- **D. docker make**

9. What is the primary purpose of a Docker container?

- **A. To store application source code**
- **B. To run applications in isolated environments**
- **C. To backup data securely**
- **D. To manage network traffic**

10. Which statement describes the purpose of Docker volumes?

- **A. To improve the performance of containers**
- **B. To provide persistent storage for container data**
- **C. To manage container networking**
- **D. To control resource limits on containers**

Answers

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

Explanations

1. What is the role of a Dockerfile?

A. To log events

B. To define the build instructions for an image

C. To perform system checks

D. To manage container data volumes

A Dockerfile plays a crucial role in the Docker ecosystem by defining the build instructions for a Docker image. It is essentially a script composed of a series of commands and instructions that specify how to create a Docker image. Each line in a Dockerfile represents a step in the image building process, which could include installing software, copying files, setting environment variables, and configuring the application to run. By using a Dockerfile, developers can automate the process of creating Docker images, ensuring consistency and reproducibility. The instructions in the Dockerfile allow Docker to understand how to set up the environment and run the application contained within the image. This makes it easier to share and distribute applications across various environments without worrying about differences in setup or dependencies. The other options describe functions that are not directly related to the purpose of a Dockerfile. Logging events, performing system checks, and managing container data volumes relate to operational aspects of using containers rather than the foundational role of defining how images are built.

2. What detail does 'docker network inspect' provide?

A. High-level overview of all containers

B. Resource usage of each container

C. Detailed information about a specified network

D. Logs from container execution

The command 'docker network inspect' is specifically designed to provide detailed information about a specified network within the Docker environment. When you run this command, it returns comprehensive data regarding various attributes of the network, such as its name, ID, driver, configuration settings, and the containers that are connected to that network. This information is essential for understanding how your containers communicate with each other across different networks. The other options do not accurately reflect the function of this command. For example, a high-level overview of all containers would require a different command that focuses specifically on container status and details. Similarly, resource usage data would typically be retrieved using commands that monitor container performance metrics. Finally, logs from container execution are accessed through logging commands rather than network inspection, as they deal with the runtime output of containerized applications. Therefore, the choice that best describes the output of 'docker network inspect' is the detailed information about a specified network.

3. Which is a secure method for allowing a Docker client to authenticate with a registry that uses a self-signed certificate?

A. Configuring the Docker daemon to allow anonymous access

B. Adding the self-signed certificate to /etc/docker/certs.d/

C. Running Docker with elevated privileges

D. Using an external authentication service

Adding the self-signed certificate to /etc/docker/certs.d/ is a secure method for enabling a Docker client to authenticate with a registry that uses a self-signed certificate. When Docker communicates with a registry over HTTPS, it verifies the server's SSL certificate to ensure a secure connection. By placing the self-signed certificate in the specified directory, you are effectively telling Docker to trust that certificate, allowing for secure communication without exposing the client to potential security risks associated with untrusted connections. This method ensures that the communication remains encrypted and the identity of the registry is validated, which is essential for protecting sensitive data and maintaining the integrity of the communication between the client and the registry. In contrast, the other methods would either compromise security or do not directly address the issue of certificate validation. For instance, anonymous access would potentially expose the registry to unauthorized access, running Docker with elevated privileges doesn't specifically resolve certificate trust issues, and using an external authentication service does not directly relate to the handling of self-signed certificates.

4. Which command is used to remove all stopped containers?

A. docker rm --all

B. docker container prune

C. docker stop all

D. docker delete --stopped

The command used to remove all stopped containers is indeed "docker container prune." This command efficiently cleans up your Docker environment by removing any stopped containers, freeing up space without affecting any running containers or other elements such as images or volumes. When you run "docker container prune," Docker will prompt you for confirmation and, once confirmed, will delete all containers that are not currently running. This is particularly useful for maintaining a clean environment and ensuring that you do not waste unnecessary resources on containers that are no longer active. In contrast, the other options do not achieve the intended outcome. For example, "docker rm --all" is not a valid command as the correct usage requires specifying individual container IDs or names without a general flag. "docker stop all" does not exist as a command; instead, you would have to specify each container to be stopped. Lastly, "docker delete --stopped" is also incorrect since "delete" is not a recognized Docker command for handling containers. The "docker container prune" command stands out as a powerful tool for Docker management, contributing to a more efficient workspace by removing unnecessary stopped containers in one simple operation.

5. Why is Docker preferred over traditional virtual machines for isolating applications?

A. Docker containers are heavier and slower than VMs

B. Docker containers share the host OS kernel

C. Docker requires more system resources than VMs

D. Docker is not compatible with microservices

Docker is preferred over traditional virtual machines primarily because Docker containers share the host operating system (OS) kernel. This architecture allows for significant efficiency gains compared to virtual machines, which require each instance to run its own complete OS, making them heavier and more resource-intensive. By leveraging the host OS, Docker containers can start up almost instantaneously and utilize system resources more efficiently. This results in faster deployment times and improved performance for applications running in containers. Furthermore, sharing the kernel means that containers can communicate with each other more efficiently as they operate at a lower level than traditional virtual machines. The other options do not reflect the advantages of Docker correctly. For instance, stating that Docker containers are heavier and slower than VMs contradicts the very reason for Docker's popularity; containers are lightweight and quick to start. Similarly, Docker's architecture is designed to use fewer system resources, and it is fully compatible with microservices, which are one of the main use cases for containerization.

6. What does 'docker stats' do?

A. Shows container network configurations

B. Displays real-time resource usage statistics for Docker containers

C. Deletes stopped containers

D. Shows the list of images on the host

The command 'docker stats' is designed to display real-time resource usage statistics for running Docker containers. When you execute this command, it provides metrics such as CPU usage, memory usage, network I/O, and disk I/O for each container. This functionality is crucial for monitoring the performance of containers and ensuring that they are operating within expected resource limits. By providing these statistics, 'docker stats' enables users to diagnose performance issues, identify which containers are consuming excessive resources, and make informed decisions about resource allocation and scaling. It is particularly advantageous in production environments where performance and resource management are vital for maintaining service quality and availability. The other options, while related to Docker functionality, do not pertain to what 'docker stats' specifically does. For example, network configurations, deleting containers, and listing images are managed by different commands, each serving a unique purpose within the Docker ecosystem.

7. Which of the following best describes the relationship between Docker and cloud services?

A. Docker is incompatible with cloud services

B. Docker is used mainly for local development and has no cloud capabilities

C. Docker can run on cloud infrastructure to enhance deployability

D. Docker eliminates the need for any cloud services

The relationship between Docker and cloud services is best characterized by the ability of Docker to run on cloud infrastructure, which enhances the deployability of applications. Docker containers package applications and their dependencies in a lightweight, portable format, making them easily deployable across various environments, including cloud platforms like AWS, Azure, and Google Cloud. By leveraging cloud services, users can benefit from the scalability, flexibility, and resource management capabilities that cloud providers offer. Docker enables developers to create a consistent runtime environment, which simplifies the deployment of applications in different cloud environments. This compatibility allows for seamless integration and optimization of resources, leading to better performance and cost efficiency. In this context, the other options do not accurately represent the relationship between Docker and cloud services. The notion that Docker is incompatible with cloud services is incorrect, as there are numerous integrations available. Additionally, suggesting that Docker is limited to local development overlooks how widely it is used in cloud-native applications. Finally, the idea that Docker eliminates the need for any cloud service is misleading; while Docker provides benefits to application deployment, it does not negate the advantages and functions that cloud services offer for managing infrastructure and resources.

8. What command is used to build a Docker image?

A. docker create

B. docker image build

C. docker build

D. docker make

The command used to build a Docker image is indeed "docker build." This command instructs Docker to read a Dockerfile that contains the specifications for the image creation process. The Dockerfile outlines the steps needed to assemble the image, including the base image to use, files to copy, commands to run, and environment configurations. When you execute this command, Docker processes the instructions sequentially and creates a new image that can be used to run containers. The "docker build" command is fundamental in Docker's workflow, making it essential for developers who need to package applications along with their dependencies into a containerized format. Without this command, you wouldn't be able to create the custom images necessary for deployment in various environments. The other options do not represent the correct command for building an image. For instance, "docker create" is used for creating a container from an image but does not involve building an image itself. Similarly, "docker image build" is not a valid Docker command; instead, "docker build" is the correct syntax to initiate the image creation process. The term "docker make" is not a recognized Docker command, further emphasizing why it isn't applicable here. Understanding the correct command for building images is crucial for successfully utilizing Docker in development and

9. What is the primary purpose of a Docker container?

A. To store application source code

B. To run applications in isolated environments

C. To backup data securely

D. To manage network traffic

The primary purpose of a Docker container is to run applications in isolated environments. This is one of the cornerstone features that Docker offers, enabling developers to package their applications and all of their dependencies into a lightweight, portable container. By encapsulating the application along with its environment, Docker containers ensure that the application runs consistently across different computing environments, whether it's on a developer's local machine, a testing server, or in a production environment in the cloud. This isolation allows multiple containers to run on the same host without interfering with each other, providing an efficient way to utilize system resources while maintaining application performance and security. In contrast, the other choices focus on different functionalities that are not the primary intent of containers. For example, while storing application source code is critical in software development, it is not the main function of a Docker container. Similarly, data backup and secure management of network traffic could involve containers but are not the primary purpose. The essence of containers revolves around the capability to execute applications with all necessary dependencies in a secure, isolated manner, leading to enhanced portability and consistency.

10. Which statement describes the purpose of Docker volumes?

A. To improve the performance of containers

B. To provide persistent storage for container data

C. To manage container networking

D. To control resource limits on containers

The purpose of Docker volumes is to provide persistent storage for data generated and used by Docker containers. Unlike the container filesystem, which is ephemeral and tied to the lifecycle of the container, volumes exist independently and can be reused across multiple containers. This means that even if a container is removed, the data in the volume remains intact, allowing for data continuity and persistence. Volumes also facilitate easier data sharing among containers and can be managed more effectively through Docker commands. They are commonly used for databases, application logs, and other forms of data that need to be retained beyond the lifecycle of a single container instance. This characteristic of volumes is crucial for applications that require data consistency and durability, ensuring that data is not lost when containers are stopped or removed.

Next Steps

Congratulations on reaching the final section of this guide. You've taken a meaningful step toward passing your certification exam and advancing your career.

As you continue preparing, remember that consistent practice, review, and self-reflection are key to success. Make time to revisit difficult topics, simulate exam conditions, and track your progress along the way.

If you need help, have suggestions, or want to share feedback, we'd love to hear from you. Reach out to our team at hello@examzify.com.

Or visit your dedicated course page for more study tools and resources:

https://dockerassociate.examzify.com

We wish you the very best on your exam journey. You've got this!