**Here are some CKAD-style questions based on the scenarios presented in the image:**

**Scenario 1: Namespace Listing and Redirection**

**Question:**

Write a Kubernetes command to list all namespaces within the cluster and redirect the output to a file named namespaces.txt.

**Scenario 2: Pod Creation with Labels**

**Question:**

Create a pod named my-pod with a container named my-container using the image nginx:latest. Assign the labels app=my-app and environment=dev to the pod.

**Scenario 3: Secret Creation and Mounting**

**Question:**

Create a secret named my-secret with the key MY\_SECRET and the value my-secret-value. Mount this secret as an environment variable named MY\_SECRET\_ENV to a pod named my-pod.

**Scenario 4: Deployment Creation and Tag Modification**

**Question:**

Create a deployment named my-deployment with the image my-image:v1. Change the tag of the image to v2 and record the changes.

**Scenario 5: Resource Requests and Limits**

**Question:**

Set a memory request of 512Mi and a memory limit of 1024Mi for the container in a pod named my-pod. Ensure the limit is half of the maximum allowed for pods in that namespace.

**Scenario 6: RBAC Concepts**

**Question:**

Explain the purpose of ClusterRole, ClusterRoleBinding, Role, RoleBinding, and Service Accounts in Kubernetes RBAC.

**Scenario 7: Sidecar Container and EmptyDir Volume**

**Question:**

Add a sidecar container to a pod named my-pod that prints the logs of the main container. Use an emptyDir volume to share data between the two containers. The command to print logs will be provided.

**Scenario 8: Multi-Container Pod**

**Question:**

Create a pod with two containers. The image names and other relevant information for the containers will be provided.

**Answers:**

**Scenario 1:**

Bash

kubectl get namespaces > namespaces.txt

**Scenario 2:**

Bash

kubectl create pod my-pod --image nginx:latest \

--labels app=my-app,environment=dev

**Scenario 3:**

Bash

kubectl create secret generic my-secret \

--from-literal MY\_SECRET=my-secret-value

kubectl create pod my-pod \

--env-from=secret=my-secret

**Scenario 4:**

Bash

kubectl create deployment my-deployment \

--image my-image:v1

kubectl set image deployment/my-deployment \

my-container=my-image:v2

**Scenario 5:**

YAML

apiVersion: v1

kind: Pod

metadata:

name: my-pod

spec:

containers:

- name: my-container

image: my-image

resources:

requests:

memory:1 "512Mi"

limits:

memory: "1024Mi"

**Scenario 6:**

* ClusterRole: Defines a set of permissions that can be granted to users or groups.
* ClusterRoleBinding: Assigns a ClusterRole to a user or group.
* Role: Defines a set of permissions within a namespace.
* RoleBinding: Assigns a Role to a user or group within a namespace.
* ServiceAccount: Represents a service account within a namespace, used for authentication and authorization.

**Scenario 7:**

YAML

apiVersion: v1

kind: Pod

metadata:

name: my-pod

spec:

volumes:

- name: log-volume

emptyDir: {}

containers:

- name: main-container

image: my-image

volumeMounts:

- name: log-volume

mountPath: /logs

- name: sidecar-container

image: my-sidecar-image

volumeMounts:

- name: log-volume

mountPath: /logs

command: ["cat", "/logs/main-container.log"]

**Scenario 8:**

YAML

apiVersion: v1

kind: Pod

metadata:

name: my-pod

spec:

containers:

- name: container1

image: image1

# other container1 configuration

- name: container2

image: image2

# other container2 configuration

**Remember to replace placeholders like image names, commands, and other specific details with the actual values provided in the problem.**

**Here are some CKAD-style questions based on the scenarios presented in the image:**

**Scenario 1: Network Policy**

**Question:**

Write a Kubernetes network policy that allows pods with the label key: value to receive traffic only from pods with labels key: Y and key: Z within the same namespace. Additionally, allow egress traffic on port 53.

**Scenario 2: Resource Limits**

**Question:**

How would you set CPU and memory limits for containers within a pod? Explain the syntax and the use of resource requests.

**Scenario 3: Ingress Troubleshooting**

**Question:**

An Ingress rule is not routing traffic as expected. What are some potential causes for this issue? How would you debug the Ingress, Service, and Service selector?

**Scenario 4: Security Context**

**Question:**

How would you set the security context of a container to disable privilege escalation and run the container as user 2000?

**Scenario 5: Canary Deployment**

**Question:**

Describe the process of creating a canary deployment that routes only 20% of traffic to a new version of an application.

**Scenario 6: NodePort Exposure**

**Question:**

Create a pod and expose it using a NodePort of 32000. Explain how clients can access the pod using this NodePort.

**Scenario 7: CronJob Configuration**

**Question:**

Create a CronJob that runs every 5 minutes, deletes pods after 10 seconds, and tolerates 3 successful and 3 failed completions.

**Answers (Briefly):**

**Scenario 1:**

YAML

apiVersion: networking.k8s.io/v1

kind: NetworkPolicy

metadata:

name: my-network-policy

spec:

podSelector:

matchLabels:

key: value

policyTypes:

- Ingress

- Egress

ingress:

- from:

- podSelector:

matchLabels:

key: Y

- podSelector:

matchLabels:

key: Z

egress:

- to:

- ipBlock:

ipBlock: 0.0.0.0/0

ports:

- portNumber: 53

protocol: TCP

**Scenario 2:**

Use the resources field in the container definition:

YAML

resources:

limits:

cpu: "2"

memory: "2Gi"

requests:

cpu: "1"

memory: "1Gi"

**Scenario 3:**

* Check Ingress rules, Service definitions, and Service selectors for errors.
* Use kubectl describe ingress, kubectl describe service, and kubectl describe pod to gather information.
* Verify network connectivity and firewall rules.

**Scenario 4:**

YAML

securityContext:

privileged: false

runAsUser: 2000

**Scenario 5:**

Use a Deployment with a Canary strategy and a suitable traffic splitting mechanism (e.g., Istio, Kubernetes Ingress).

**Scenario 6:**

YAML

apiVersion: v1

kind: Pod

metadata:

name: my-pod

spec:

containers:

- name: my-container

image: my-image

ports:

- containerPort: 80801

nodePort: 32000

Clients can access the pod using the NodeIP:NodePort (e.g., 192.168.1.100:32000).

**Scenario 7:**

YAML

apiVersion: batch/v1beta1

kind: CronJob

metadata:

name: my-cronjob

spec:

schedule: "\*/5 \* \* \* \*"

jobTemplate:

spec:

template:

spec:

restartPolicy: OnFailure

containers:

- name:2 my-job

image: my-image

tolerations:

- key: my-toleration

operator: Exists

ttlSecondsAfterFinished: 10

backoffLimit: 3

**Here are some CKAD-style questions based on the scenarios presented in the image:**

**Scenario 1: Pod Sorting and Filtering**

**Question:**

1. Write a Kubernetes command to list all pods sorted by their creation time, newest first.
2. Write a Kubernetes command to list all pods with the label app=my-app and write the results to a file named app-pods.txt.

**Scenario 2: Helm Chart Operations**

**Question:**

1. How would you deploy a Helm chart named my-chart with 3 replicas?
2. How would you upgrade the deployed Helm chart to the latest version?
3. How would you roll back a Helm chart to a previous revision?

**Scenario 3: Container Image Building and Pushing**

**Question:**

Write a command to build a container image from a Dockerfile named Dockerfile using Docker, tag it as my-image:latest, and push it to a container registry.

**Scenario 4: Service Account Permissions**

**Question:**

A mounted service account is unable to list Persistent Volume Claims (PVCs) in a namespace. How would you troubleshoot and fix this issue?

**Scenario 5: Service Account Token**

**Question:**

How would you copy the token of a service account to a file?

**Scenario 6: Namespace Migration**

**Question:**

Describe the steps involved in migrating a deployment and its associated service from one namespace to another.

**Answers (Briefly):**

**Scenario 1:**

1. kubectl get pods --sort-by=.metadata.creationTimestamp
2. kubectl get pods -l app=my-app -o name > app-pods.txt

**Scenario 2:**

1. helm install my-chart my-chart/chart --set replicas=3
2. helm upgrade my-chart my-chart/chart
3. helm rollback my-chart <revision-number>

**Scenario 3:**

Bash

docker build -t my-image:latest .

docker push my-image:latest

**Scenario 4:**

* Check the service account's RBAC permissions. Ensure it has the list permission for PVCs in the namespace.
* Verify the service account is mounted correctly in the pod.
* If the service account is part of a RoleBinding, ensure the Role has the necessary permissions.

**Scenario 5:**

Bash

kubectl describe secret <service-account-name> -n <namespace> | grep token | awk -F: '{ print $2 }' > token.txt

**Scenario 6:**

1. Create a new namespace.
2. Create a new ServiceAccount in the new namespace with necessary permissions.
3. Update the Deployment's spec to use the new ServiceAccount.
4. Update the Service's spec to use the new ServiceAccount.
5. Apply the updated Deployment and Service manifests.
6. Optionally, delete the resources from the old namespace.

**Remember to replace placeholders like chart names, image names, service account names, and namespaces with the actual values in your specific scenario.**

**CKAD Questions Based on the Image**

**Question 1: Job Creation and Completion**

**Prompt:**

Create a Job that runs an image and a command inside it. The job should complete successfully for 3 times. You can create multiple pods to finish your job faster.

**Question:** Write a Kubernetes Job manifest that will:

1. Run a specified image with a given command.
2. Ensure the Job completes successfully 3 times.
3. Allow for parallel execution using multiple pods.

**Question 2: Finding Pending Helm Releases**

**Prompt:**

Find the Helm release which is in pending state

**Question:** How would you identify a Helm release that is currently in a pending state? What command-line tool or API can be used to check the status of Helm releases?

**Question 3: Adding Readiness Probe**

**Prompt:**

Add readiness probe to the container to get http response from endpoint/health on port 5432. Probe should initially wait for 5s and periodically execute every 10 seconds

**Question:** Create a Kubernetes Pod manifest that includes a readiness probe for a container. The probe should:

1. Check the health endpoint /health on port 5432.
2. Initially wait for 5 seconds before starting the checks.
3. Perform checks every 10 seconds.

**Question 4: Fixing Deprecated Deployment API**

**Prompt:**

A deployment is using deprecated API. Fix it

**Question:** How would you identify and fix a Deployment that is using a deprecated API version? What are the common signs of a deprecated Deployment?

**Question 5: Creating a Deployment from a Template**

**Prompt:**

A pod template is given. Create a deployment of 3 replicas from this template

**Question:** Given a Pod template, create a Deployment manifest that:

1. Deploys 3 replicas of the Pod.
2. Configures appropriate labels and selectors for the Deployment and Pods.

**Question 6: Creating PV and PVC**

**Prompt:**

Create a PV and PVC. Mount this PVC at /tmp of the container

**Question:** Create a PersistentVolume (PV) and PersistentVolumeClaim (PVC) that:

1. Defines the storage class and capacity for the PV.
2. Claims the storage from the PV for the PVC.
3. Mounts the PVC to the /tmp directory of a container in a Pod.

**Question 7: Creating a TLS Secret**

**Prompt:**

Create a TLS secret. Cert and Key are provided

**Question:** How would you create a Kubernetes Secret to store a TLS certificate and private key? What command-line tool can be used to create this Secret?

**Question 8: Creating a ConfigMap from a File**

**Prompt:**

Create a configmap from a file and mount it as Volume

**Question:** Create a ConfigMap from a file named config.yaml and mount it as a volume to a container in a Pod. What is the command-line tool to create a ConfigMap?

**CKAD Questions Based on the Image**

**Question 1: Empty Service Endpoints**

**Prompt:**

Endpoints of a service are empty. Investigate and fix.

**Question:** What are the possible reasons for a Service's endpoints being empty? How would you troubleshoot and fix this issue?

**Question 2: Changing Service Type**

**Prompt:**

Change the service type of a service from ClusterIP to NodePort using kubectl patch. Write the command to a file.

**Question:** Write a kubectl patch command to change the type of a Service named "my-service" from ClusterIP to NodePort. Redirect the command to a file named "change-service-type.sh".

**Question 3: Creating ResourceQuota and Pod**

**Prompt:**

Create a ResourceQuota with the given specifications. Create a Pod with the resources given.

**Question:** Given specific CPU and memory limits, create a ResourceQuota for a namespace and then create a Pod that adheres to those limits.

**Question 4: Using Existing Service Account**

**Prompt:**

Use an existing service account X with an existing deployment.

**Question:** How would you modify a Deployment to use an existing ServiceAccount named "my-service-account"? What are the potential benefits of using ServiceAccounts?

**CKAD Exam Tips Based on the Image**

**Based on the provided image, here are some tips for preparing and taking the CKAD exam:**

**Preparation Tips:**

1. **Clear Understanding of Topics:** Ensure you have a strong grasp of all the core Kubernetes concepts.
2. **Practice Regularly:** Dedicate at least a month to consistent practice. Some questions might be tricky, so practice a variety of scenarios.
3. **Local Kubernetes Cluster:** Set up a local cluster using tools like Minikube to experiment hands-on.
4. **Refer to Official Documentation:** Use the official Kubernetes documentation as the primary source of information. Utilize the search bar to quickly find specific topics.
5. **YAML Definition Search:** Learn to quickly search for YAML definitions by entering kind: RESOURCE in the search bar of your browser or terminal.
6. **Practice with Exam Simulators:** Use exam simulators like the one provided by Killer Shell to get familiar with the exam format and time constraints.

**Exam Taking Tips:**

* **Time Management:** You'll have ample time to answer all questions. Many people complete the exam in under 1.5 hours.
* **Avoid Unnecessary Aliases:** Setting up aliases for kubectl commands might not be beneficial. You'll likely end up typing the full command anyway.
* **Focus on Fundamentals:** A solid understanding of Kubernetes core concepts is crucial.

**Additional Tips:**

* **Stay Updated:** Keep up with the latest Kubernetes releases and features.
* **Learn from the Community:** Engage with the Kubernetes community through forums, Slack channels, or meetups.
* **Practice with Real-World Scenarios:** Try to apply your knowledge to real-world use cases.
* **Stay Calm and Confident:** A calm and focused approach will help you perform your best.

By following these tips and dedicating sufficient time to practice, you can increase your chances of success on the CKAD exam.

**CKAD Exam Tips: Additional Insights**

**Based on the provided image, here are some additional tips for the CKAD exam:**

**Exam-Taking Strategies**

* **Use Imperative Commands for Quick YAML Creation:** Leverage imperative commands like kubectl run to quickly generate YAML files. This can save time during the exam.
* **Utilize kubectl describe and kubectl log:** These commands are invaluable for troubleshooting and verifying the state of your Kubernetes resources.
* **Flag Difficult Questions:** If you encounter a challenging question, flag it and move on. You can return to it later.
* **Manage Context Switching:** Before answering a new question, ensure you're in the correct context (namespace, resource, etc.).
* **Efficient Resource Deletion:** Use the --grace-period=0 --force flags to quickly delete resources without waiting for the grace period.
* **Display Resource Names Only:** Use the -o name flag to display only the names of resources, saving space and improving readability.

**General Tips**

* **Practice Regularly:** Consistent practice is key. Use online labs, local Kubernetes setups, or cloud-based platforms to hone your skills.
* **Understand Core Concepts:** Focus on understanding the fundamental concepts of Kubernetes, such as Pods, Services, Deployments, and Namespaces.
* **Master YAML:** YAML is the lingua franca of Kubernetes. Practice reading, writing, and understanding YAML manifests.
* **Learn to Troubleshoot:** Develop troubleshooting skills to diagnose and fix common Kubernetes issues.
* **Stay Updated:** Keep up with the latest Kubernetes features and best practices.

By following these tips and practicing consistently, you can increase your chances of success on the CKAD exam.

I cannot directly provide a Word document, but I can provide the questions and answers in a clear, formatted text format that you can easily copy and paste into a Word document.

**Here are the combined questions and answers from the previous responses:**

**Question 1: Pod Creation and Resource Limits**

Create a Pod named "my-pod" with the following specifications:

* Image: nginx:latest
* Name: my-container
* Resource Requests:
  + CPU: 500m
  + Memory: 1Gi
* Resource Limits:
  + CPU: 1
  + Memory: 2Gi

**Answer:**

YAML

apiVersion: v1

kind: Pod

metadata:

name: my-pod

spec:

containers:

- name: my-container

image: nginx:latest

resources:

requests:

cpu:1 "500m"

memory: "1Gi"

limits:

cpu: "1"

memory: "2Gi"

**Question 2: Service Creation and Exposure**

Create a Service named "my-service" that exposes the Pod "my-pod" on port 8080. Use a NodePort service type.

**Answer:**

YAML

apiVersion: v1

kind: Service

metadata:

name: my-service

spec:

selector:

app: my-app # Assuming the pod has a label "app: my-app"

ports:

- protocol: TCP

port: 80

targetPort: 8080

nodePort: 30000 # Choose a random port number

type: NodePort

**Question 3: Deployment Creation with Rolling Update**

Create a Deployment named "my-deployment" with 3 replicas of the Pod "my-pod". Configure a rolling update strategy with a maximum of 2 unavailable pods at a time.

**Answer:**

YAML

apiVersion: apps/v1

kind: Deployment

metadata:

name: my-deployment

spec:

replicas: 3

selector:

matchLabels:

app: my-app

template:

metadata:

labels:

app: my-app

spec:

containers:

- name: my-container2

image: nginx:latest

strategy:

type: RollingUpdate

rollingUpdate:

maxUnavailable:3 2

**Question 4: ConfigMap Creation and Mounting**

Create a ConfigMap named "my-configmap" from a file "config.yaml" and mount it as a volume to the container in the Deployment "my-deployment".

**Answer:**

Bash

kubectl create configmap my-configmap --from-file=config.yaml

# In the Deployment manifest:

spec:

template:

spec:

containers:

- name: my-container

volumeMounts:

- name: config-volume

mountPath: /etc/config

volumes:

- name: config-volume

configMap:

name: my-configmap

**Question 5: Troubleshooting Empty Service Endpoints**

You notice that the endpoints of a Service are empty. What are the possible causes and how would you troubleshoot this issue?

**Answer:**

Possible causes:

* Pod is not running or not ready.
* Service selector doesn't match the Pod's labels.
* Network issues.

Troubleshooting steps:

* Check Pod status using kubectl get pods.
* Verify Service selector and Pod labels.
* Inspect network configuration.
* Use kubectl describe service and kubectl describe pod for more details.

**You can copy and paste this text into a Word document and format it as needed.**

If you need more questions or have specific requirements, please let me know.