kubectl create deployment nginx --image=nginx

kubectl create service clusterip my-service --tcp=80:8080

kubectl create namespace my-namespace

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

kubectl create secret generic my-secret --from-literal=username=admin --from-literal=password=password123

kubectl create -f pod.yaml

kubectl create -f deployment.yaml

kubectl create -f service.yaml

kubectl create -f persistentvolume.yaml

kubectl create -f persistentvolumeclaim.yaml

kubectl get pods

kubectl get services

kubectl get deployments

kubectl get nodes

kubectl get namespaces

kubectl get configmaps

kubectl get secrets

kubectl get pv

kubectl get pvc

kubectl describe pod my-pod

kubectl describe service my-service

kubectl describe deployment my-deployment

kubectl describe node my-node

kubectl describe namespace my-namespace

kubectl describe pod my-pod

kubectl describe service my-service

kubectl describe deployment my-deployment

kubectl describe node my-node

kubectl describe namespace my-namespace

kubectl delete pod my-pod

kubectl delete service my-service

kubectl delete deployment my-deployment

kubectl delete node my-node

kubectl delete namespace my-namespace

kubectl apply -f pod.yaml

kubectl apply -f deployment.yaml

kubectl apply -f service.yaml

kubectl scale deployment/my-deployment --replicas=3

kubectl exec -it my-pod -- bash

kubectl exec my-pod -- ls /app

kubectl exec my-pod -- wget [http://example.com/file.txt -O /tmp/file.txt](http://example.com/file.txt%20-O%20/tmp/file.txt)

kubectl logs my-pod

kubectl logs -f my-pod

kubectl rollout status deployment/my-deployment

kubectl rollout history deployment/my-deployment

Sure, here's a list of common Kubernetes commands along with five examples for each:

1. \*\*kubectl create\*\*: This command creates a new Kubernetes resource.

- Example 1: `kubectl create deployment nginx --image=nginx` creates a deployment named "nginx" with the nginx container image.

- Example 2: `kubectl create namespace my-namespace` creates a new namespace named "my-namespace".

- Example 3: `kubectl create -f pod.yaml` creates a pod using the configuration specified in the YAML file.

- Example 4: `kubectl create service nodeport my-service --tcp=8080:80` creates a NodePort service named "my-service" exposing port 8080 on the nodes.

- Example 5: `kubectl create configmap my-config --from-file=config.txt` creates a ConfigMap named "my-config" with data from the file "config.txt".

2. \*\*kubectl apply\*\*: This command creates or updates Kubernetes resources using a configuration file.

- Example 1: `kubectl apply -f deployment.yaml` applies the configuration in the YAML file to create or update a deployment.

- Example 2: `kubectl apply -f service.yaml` applies the configuration in the YAML file to create or update a service.

- Example 3: `kubectl apply -f namespace.yaml` applies the configuration in the YAML file to create or update a namespace.

- Example 4: `kubectl apply -f secret.yaml` applies the configuration in the YAML file to create or update a secret.

- Example 5: `kubectl apply -f configmap.yaml` applies the configuration in the YAML file to create or update a configmap.

3. \*\*kubectl get\*\*: This command retrieves one or more Kubernetes resources.

- Example 1: `kubectl get pods` retrieves a list of all pods in the current namespace.

- Example 2: `kubectl get nodes` retrieves a list of all nodes in the cluster.

- Example 3: `kubectl get deployments` retrieves a list of all deployments in the current namespace.

- Example 4: `kubectl get services` retrieves a list of all services in the current namespace.

- Example 5: `kubectl get configmaps` retrieves a list of all configmaps in the current namespace.

4. \*\*kubectl describe\*\*: This command provides detailed information about a Kubernetes resource.

- Example 1: `kubectl describe pod my-pod` provides detailed information about the pod named "my-pod".

- Example 2: `kubectl describe node my-node` provides detailed information about the node named "my-node".

- Example 3: `kubectl describe deployment my-deployment` provides detailed information about the deployment named "my-deployment".

- Example 4: `kubectl describe service my-service` provides detailed information about the service named "my-service".

- Example 5: `kubectl describe configmap my-configmap` provides detailed information about the configmap named "my-configmap".

5. \*\*kubectl delete\*\*: This command deletes one or more Kubernetes resources.

- Example 1: `kubectl delete pod my-pod` deletes the pod named "my-pod".

- Example 2: `kubectl delete deployment my-deployment` deletes the deployment named "my-deployment".

- Example 3: `kubectl delete service my-service` deletes the service named "my-service".

- Example 4: `kubectl delete namespace my-namespace` deletes the namespace named "my-namespace" and all resources within it.

- Example 5: `kubectl delete -f pod.yaml` deletes the pod specified in the YAML file.

These examples cover some of the most commonly used Kubernetes commands and demonstrate their usage in various scenarios.

Certainly! Here are more Kubernetes commands with examples:

6. \*\*kubectl exec\*\*: This command executes a command in a running container of a pod.

- Example 1: `kubectl exec -it my-pod -- /bin/bash` opens a bash shell in the pod named "my-pod".

- Example 2: `kubectl exec my-pod -- ls /app` lists the contents of the "/app" directory in the pod named "my-pod".

- Example 3: `kubectl exec my-pod -- wget http://example.com/file.txt -O /tmp/file.txt` downloads a file from a URL into the "/tmp" directory in the pod named "my-pod".

- Example 4: `kubectl exec -it my-pod -- ps aux` lists all processes running inside the container in the pod named "my-pod".

- Example 5: `kubectl exec my-pod -- curl http://localhost:8080/api` sends a GET request to the specified URL from within the pod named "my-pod".

7. \*\*kubectl logs\*\*: This command prints the logs from a container in a pod.

- Example 1: `kubectl logs my-pod` prints the logs from all containers in the pod named "my-pod".

- Example 2: `kubectl logs my-pod -c my-container` prints the logs from the container named "my-container" in the pod named "my-pod".

- Example 3: `kubectl logs -f my-pod` prints the logs from all containers in the pod named "my-pod" and continues to stream new logs as they are generated.

- Example 4: `kubectl logs --tail=100 my-pod` prints the last 100 lines of logs from all containers in the pod named "my-pod".

- Example 5: `kubectl logs --since=1h my-pod` prints logs from all containers in the pod named "my-pod" generated in the last hour.

8. \*\*kubectl scale\*\*: This command scales the number of replicas of a deployment, replica set, or stateful set.

- Example 1: `kubectl scale deployment/my-deployment --replicas=3` scales the deployment named "my-deployment" to have 3 replicas.

- Example 2: `kubectl scale rs/my-replicaset --replicas=5` scales the replica set named "my-replicaset" to have 5 replicas.

- Example 3: `kubectl scale statefulset/my-statefulset --replicas=2` scales the stateful set named "my-statefulset" to have 2 replicas.

- Example 4: `kubectl scale deployment/my-deployment --replicas=0` scales the deployment named "my-deployment" down to 0 replicas, effectively stopping it.

- Example 5: `kubectl scale deployment/my-deployment --replicas=1` scales the deployment named "my-deployment" to have 1 replica, effectively starting it if it was stopped.

9. \*\*kubectl edit\*\*: This command edits a Kubernetes resource in real-time using the default editor.

- Example 1: `kubectl edit deployment/my-deployment` opens the default editor to edit the deployment named "my-deployment".

- Example 2: `kubectl edit configmap/my-configmap` opens the default editor to edit the configmap named "my-configmap".

- Example 3: `kubectl edit pod/my-pod` opens the default editor to edit the pod named "my-pod".

- Example 4: `kubectl edit service/my-service` opens the default editor to edit the service named "my-service".

- Example 5: `kubectl edit secret/my-secret` opens the default editor to edit the secret named "my-secret".

These additional examples cover more Kubernetes commands that are commonly used for managing and interacting with Kubernetes resources.

Certainly! Here are more Kubernetes commands with examples:

10. \*\*kubectl port-forward\*\*: This command creates a secure tunnel between a local machine and a pod, deployment, or service running on Kubernetes, allowing direct access to them.

- Example 1: `kubectl port-forward pod/my-pod 8080:80` forwards local port 8080 to port 80 on pod "my-pod".

- Example 2: `kubectl port-forward deployment/my-deployment 8080:80` forwards local port 8080 to port 80 on a pod managed by the deployment "my-deployment".

- Example 3: `kubectl port-forward service/my-service 8080:80` forwards local port 8080 to port 80 on a pod selected by the service "my-service".

- Example 4: `kubectl port-forward pod/my-pod 8080:80 9090:9090` forwards local port 8080 to port 80 and port 9090 to port 9090 on pod "my-pod".

- Example 5: `kubectl port-forward deployment/my-deployment 8080:80 --context=my-context` forwards local port 8080 to port 80 on a pod managed by the deployment "my-deployment" in the specified context.

11. \*\*kubectl rollout\*\*: This command manages rollouts of updates to Kubernetes resources.

- Example 1: `kubectl rollout status deployment/my-deployment` checks the status of the rollout for the deployment "my-deployment".

- Example 2: `kubectl rollout history deployment/my-deployment` displays the revision history of the deployment "my-deployment".

- Example 3: `kubectl rollout undo deployment/my-deployment` rolls back the deployment "my-deployment" to the previous revision.

- Example 4: `kubectl rollout restart deployment/my-deployment` restarts the rollout of the deployment "my-deployment".

- Example 5: `kubectl rollout pause deployment/my-deployment` pauses the rollout of the deployment "my-deployment".

12. \*\*kubectl label\*\*: This command adds or updates labels on Kubernetes resources.

- Example 1: `kubectl label pods my-pod app=backend` adds the label "app=backend" to the pod "my-pod".

- Example 2: `kubectl label nodes my-node environment=production` adds the label "environment=production" to the node "my-node".

- Example 3: `kubectl label deployments my-deployment tier=frontend` adds the label "tier=frontend" to the deployment "my-deployment".

- Example 4: `kubectl label services my-service app=web` adds the label "app=web" to the service "my-service".

- Example 5: `kubectl label namespace my-namespace division=marketing` adds the label "division=marketing" to the namespace "my-namespace".

These examples cover more Kubernetes commands for various management and interaction tasks, expanding your toolkit for Kubernetes administration and deployment.

Of course! Here are more Kubernetes commands along with examples:

13. \*\*kubectl rollout\*\*: This command is used to manage rollouts of updates to Kubernetes resources.

- Example 1: `kubectl rollout status deployment/my-deployment` checks the status of the rollout for the deployment named "my-deployment".

- Example 2: `kubectl rollout history deployment/my-deployment` displays the revision history of the deployment named "my-deployment".

- Example 3: `kubectl rollout undo deployment/my-deployment` rolls back the deployment named "my-deployment" to the previous revision.

- Example 4: `kubectl rollout restart deployment/my-deployment` restarts the rollout of the deployment named "my-deployment".

- Example 5: `kubectl rollout pause deployment/my-deployment` pauses the rollout of the deployment named "my-deployment".

14. \*\*kubectl annotate\*\*: This command adds or updates annotations on Kubernetes resources.

- Example 1: `kubectl annotate pod/my-pod description="My Pod Description"` adds an annotation with the key "description" and value "My Pod Description" to the pod named "my-pod".

- Example 2: `kubectl annotate node/my-node environment=production` adds an annotation with the key "environment" and value "production" to the node named "my-node".

- Example 3: `kubectl annotate deployment/my-deployment owner=John` adds an annotation with the key "owner" and value "John" to the deployment named "my-deployment".

- Example 4: `kubectl annotate service/my-service app=web` adds an annotation with the key "app" and value "web" to the service named "my-service".

- Example 5: `kubectl annotate namespace/my-namespace division=marketing` adds an annotation with the key "division" and value "marketing" to the namespace named "my-namespace".

15. \*\*kubectl rollout\*\*: This command manages rollouts of updates to Kubernetes resources.

- Example 1: `kubectl rollout status deployment/my-deployment` checks the status of the rollout for the deployment named "my-deployment".

- Example 2: `kubectl rollout history deployment/my-deployment` displays the revision history of the deployment named "my-deployment".

- Example 3: `kubectl rollout undo deployment/my-deployment` rolls back the deployment named "my-deployment" to the previous revision.

- Example 4: `kubectl rollout restart deployment/my-deployment` restarts the rollout of the deployment named "my-deployment".

- Example 5: `kubectl rollout pause deployment/my-deployment` pauses the rollout of the deployment named "my-deployment".

These commands provide additional functionality for managing and interacting with Kubernetes resources, enhancing your ability to handle various deployment and administrative tasks.

Certainly! Here are more Kubernetes commands along with examples:

16. \*\*kubectl scale\*\*: This command scales the number of replicas of a deployment, replica set, or stateful set.

- Example 1: `kubectl scale deployment/my-deployment --replicas=3` scales the deployment named "my-deployment" to have 3 replicas.

- Example 2: `kubectl scale rs/my-replicaset --replicas=5` scales the replica set named "my-replicaset" to have 5 replicas.

- Example 3: `kubectl scale statefulset/my-statefulset --replicas=2` scales the stateful set named "my-statefulset" to have 2 replicas.

- Example 4: `kubectl scale deployment/my-deployment --replicas=0` scales the deployment named "my-deployment" down to 0 replicas, effectively stopping it.

- Example 5: `kubectl scale deployment/my-deployment --replicas=1` scales the deployment named "my-deployment" to have 1 replica, effectively starting it if it was stopped.

17. \*\*kubectl edit\*\*: This command edits a Kubernetes resource in real-time using the default editor.

- Example 1: `kubectl edit deployment/my-deployment` opens the default editor to edit the deployment named "my-deployment".

- Example 2: `kubectl edit configmap/my-configmap` opens the default editor to edit the configmap named "my-configmap".

- Example 3: `kubectl edit pod/my-pod` opens the default editor to edit the pod named "my-pod".

- Example 4: `kubectl edit service/my-service` opens the default editor to edit the service named "my-service".

- Example 5: `kubectl edit secret/my-secret` opens the default editor to edit the secret named "my-secret".

18. \*\*kubectl label\*\*: This command adds or updates labels on Kubernetes resources.

- Example 1: `kubectl label pods my-pod app=backend` adds the label "app=backend" to the pod "my-pod".

- Example 2: `kubectl label nodes my-node environment=production` adds the label "environment=production" to the node "my-node".

- Example 3: `kubectl label deployments my-deployment tier=frontend` adds the label "tier=frontend" to the deployment "my-deployment".

- Example 4: `kubectl label services my-service app=web` adds the label "app=web" to the service "my-service".

- Example 5: `kubectl label namespace my-namespace division=marketing` adds the label "division=marketing" to the namespace "my-namespace".

These examples provide further insight into Kubernetes commands for scaling deployments, editing resources, and managing labels, enabling more control and customization over your Kubernetes environment.

Certainly! Here are more Kubernetes commands along with examples:

19. \*\*kubectl exec\*\*: This command executes a command in a running container of a pod.

- Example 1: `kubectl exec -it my-pod -- /bin/bash` opens a bash shell in the pod named "my-pod".

- Example 2: `kubectl exec my-pod -- ls /app` lists the contents of the "/app" directory in the pod named "my-pod".

- Example 3: `kubectl exec my-pod -- wget http://example.com/file.txt -O /tmp/file.txt` downloads a file from a URL into the "/tmp" directory in the pod named "my-pod".

- Example 4: `kubectl exec -it my-pod -- ps aux` lists all processes running inside the container in the pod named "my-pod".

- Example 5: `kubectl exec my-pod -- curl http://localhost:8080/api` sends a GET request to the specified URL from within the pod named "my-pod".

20. \*\*kubectl logs\*\*: This command prints the logs from a container in a pod.

- Example 1: `kubectl logs my-pod` prints the logs from all containers in the pod named "my-pod".

- Example 2: `kubectl logs my-pod -c my-container` prints the logs from the container named "my-container" in the pod named "my-pod".

- Example 3: `kubectl logs -f my-pod` prints the logs from all containers in the pod named "my-pod" and continues to stream new logs as they are generated.

- Example 4: `kubectl logs --tail=100 my-pod` prints the last 100 lines of logs from all containers in the pod named "my-pod".

- Example 5: `kubectl logs --since=1h my-pod` prints logs from all containers in the pod named "my-pod" generated in the last hour.

21. \*\*kubectl delete\*\*: This command deletes one or more Kubernetes resources.

- Example 1: `kubectl delete pod my-pod` deletes the pod named "my-pod".

- Example 2: `kubectl delete deployment my-deployment` deletes the deployment named "my-deployment".

- Example 3: `kubectl delete service my-service` deletes the service named "my-service".

- Example 4: `kubectl delete namespace my-namespace` deletes the namespace named "my-namespace" and all resources within it.

- Example 5: `kubectl delete -f pod.yaml` deletes the pod specified in the YAML file.

These examples cover more Kubernetes commands essential for interacting with pods, managing logs, and deleting resources, providing comprehensive control over your Kubernetes environment.

Certainly! Here are more Kubernetes commands along with examples:

22. \*\*kubectl rollout\*\*: This command manages rollouts of updates to Kubernetes resources.

- Example 1: `kubectl rollout status deployment/my-deployment` checks the status of the rollout for the deployment named "my-deployment".

- Example 2: `kubectl rollout history deployment/my-deployment` displays the revision history of the deployment named "my-deployment".

- Example 3: `kubectl rollout undo deployment/my-deployment` rolls back the deployment named "my-deployment" to the previous revision.

- Example 4: `kubectl rollout restart deployment/my-deployment` restarts the rollout of the deployment named "my-deployment".

- Example 5: `kubectl rollout pause deployment/my-deployment` pauses the rollout of the deployment named "my-deployment".

23. \*\*kubectl annotate\*\*: This command adds or updates annotations on Kubernetes resources.

- Example 1: `kubectl annotate pod/my-pod description="My Pod Description"` adds an annotation with the key "description" and value "My Pod Description" to the pod named "my-pod".

- Example 2: `kubectl annotate node/my-node environment=production` adds an annotation with the key "environment" and value "production" to the node named "my-node".

- Example 3: `kubectl annotate deployment/my-deployment owner=John` adds an annotation with the key "owner" and value "John" to the deployment named "my-deployment".

- Example 4: `kubectl annotate service/my-service app=web` adds an annotation with the key "app" and value "web" to the service named "my-service".

- Example 5: `kubectl annotate namespace/my-namespace division=marketing` adds an annotation with the key "division" and value "marketing" to the namespace named "my-namespace".

24. \*\*kubectl rollout\*\*: This command manages rollouts of updates to Kubernetes resources.

- Example 1: `kubectl rollout status deployment/my-deployment` checks the status of the rollout for the deployment named "my-deployment".

- Example 2: `kubectl rollout history deployment/my-deployment` displays the revision history of the deployment named "my-deployment".

- Example 3: `kubectl rollout undo deployment/my-deployment` rolls back the deployment named "my-deployment" to the previous revision.

- Example 4: `kubectl rollout restart deployment/my-deployment` restarts the rollout of the deployment named "my-deployment".

- Example 5: `kubectl rollout pause deployment/my-deployment` pauses the rollout of the deployment named "my-deployment".

These commands provide additional functionality for managing and interacting with Kubernetes resources, enhancing your ability to handle various deployment and administrative tasks.

Sure, here are more Kubernetes commands with examples:

25. \*\*kubectl create\*\*: This command creates a new Kubernetes resource.

- Example 1: `kubectl create deployment nginx --image=nginx` creates a deployment named "nginx" with the nginx container image.

- Example 2: `kubectl create namespace my-namespace` creates a new namespace named "my-namespace".

- Example 3: `kubectl create -f pod.yaml` creates a pod using the configuration specified in the YAML file.

- Example 4: `kubectl create service nodeport my-service --tcp=8080:80` creates a NodePort service named "my-service" exposing port 8080 on the nodes.

- Example 5: `kubectl create configmap my-config --from-file=config.txt` creates a ConfigMap named "my-config" with data from the file "config.txt".

26. \*\*kubectl apply\*\*: This command creates or updates Kubernetes resources using a configuration file.

- Example 1: `kubectl apply -f deployment.yaml` applies the configuration in the YAML file to create or update a deployment.

- Example 2: `kubectl apply -f service.yaml` applies the configuration in the YAML file to create or update a service.

- Example 3: `kubectl apply -f namespace.yaml` applies the configuration in the YAML file to create or update a namespace.

- Example 4: `kubectl apply -f secret.yaml` applies the configuration in the YAML file to create or update a secret.

- Example 5: `kubectl apply -f configmap.yaml` applies the configuration in the YAML file to create or update a configmap.

27. \*\*kubectl get\*\*: This command retrieves one or more Kubernetes resources.

- Example 1: `kubectl get pods` retrieves a list of all pods in the current namespace.

- Example 2: `kubectl get nodes` retrieves a list of all nodes in the cluster.

- Example 3: `kubectl get deployments` retrieves a list of all deployments in the current namespace.

- Example 4: `kubectl get services` retrieves a list of all services in the current namespace.

- Example 5: `kubectl get configmaps` retrieves a list of all configmaps in the current namespace.

28. \*\*kubectl describe\*\*: This command provides detailed information about a Kubernetes resource.

- Example 1: `kubectl describe pod my-pod` provides detailed information about the pod named "my-pod".

- Example 2: `kubectl describe node my-node` provides detailed information about the node named "my-node".

- Example 3: `kubectl describe deployment my-deployment` provides detailed information about the deployment named "my-deployment".

- Example 4: `kubectl describe service my-service` provides detailed information about the service named "my-service".

- Example 5: `kubectl describe configmap my-configmap` provides detailed information about the configmap named "my-configmap".

These examples cover more Kubernetes commands that are essential for managing and interacting with Kubernetes resources, providing a comprehensive overview of its capabilities.

Certainly! Here are more Kubernetes commands with examples:

29. \*\*kubectl edit\*\*: This command edits a Kubernetes resource in real-time using the default editor.

- Example 1: `kubectl edit deployment/my-deployment` opens the default editor to edit the deployment named "my-deployment".

- Example 2: `kubectl edit configmap/my-configmap` opens the default editor to edit the configmap named "my-configmap".

- Example 3: `kubectl edit pod/my-pod` opens the default editor to edit the pod named "my-pod".

- Example 4: `kubectl edit service/my-service` opens the default editor to edit the service named "my-service".

- Example 5: `kubectl edit secret/my-secret` opens the default editor to edit the secret named "my-secret".

30. \*\*kubectl rollout\*\*: This command manages rollouts of updates to Kubernetes resources.

- Example 1: `kubectl rollout status deployment/my-deployment` checks the status of the rollout for the deployment named "my-deployment".

- Example 2: `kubectl rollout history deployment/my-deployment` displays the revision history of the deployment named "my-deployment".

- Example 3: `kubectl rollout undo deployment/my-deployment` rolls back the deployment named "my-deployment" to the previous revision.

- Example 4: `kubectl rollout restart deployment/my-deployment` restarts the rollout of the deployment named "my-deployment".

- Example 5: `kubectl rollout pause deployment/my-deployment` pauses the rollout of the deployment named "my-deployment".

31. \*\*kubectl label\*\*: This command adds or updates labels on Kubernetes resources.

- Example 1: `kubectl label pods my-pod app=backend` adds the label "app=backend" to the pod "my-pod".

- Example 2: `kubectl label nodes my-node environment=production` adds the label "environment=production" to the node "my-node".

- Example 3: `kubectl label deployments my-deployment tier=frontend` adds the label "tier=frontend" to the deployment "my-deployment".

- Example 4: `kubectl label services my-service app=web` adds the label "app=web" to the service "my-service".

- Example 5: `kubectl label namespace my-namespace division=marketing` adds the label "division=marketing" to the namespace "my-namespace".

32. \*\*kubectl exec\*\*: This command executes a command in a running container of a pod.

- Example 1: `kubectl exec -it my-pod -- /bin/bash` opens a bash shell in the pod named "my-pod".

- Example 2: `kubectl exec my-pod -- ls /app` lists the contents of the "/app" directory in the pod named "my-pod".

- Example 3: `kubectl exec my-pod -- wget http://example.com/file.txt -O /tmp/file.txt` downloads a file from a URL into the "/tmp" directory in the pod named "my-pod".

- Example 4: `kubectl exec -it my-pod -- ps aux` lists all processes running inside the container in the pod named "my-pod".

- Example 5: `kubectl exec my-pod -- curl http://localhost:8080/api` sends a GET request to the specified URL from within the pod named "my-pod".

These examples extend your knowledge of Kubernetes commands, providing more options for managing resources, performing rollouts, modifying labels, and executing commands within pods.

Sure, here's a list of some common Kubernetes commands along with examples and short explanations for each:

1. \*\*kubectl create\*\*: Creates a resource from a file or from stdin.

- Example: `kubectl create deployment nginx --image=nginx`

Explanation: Creates a deployment named "nginx" using the nginx Docker image.

2. \*\*kubectl apply\*\*: Applies a configuration to a resource by filename or stdin.

- Example: `kubectl apply -f deployment.yaml`

Explanation: Applies the configuration defined in the "deployment.yaml" file to the cluster.

3. \*\*kubectl get\*\*: Displays one or many resources.

- Example: `kubectl get pods`

Explanation: Retrieves information about all pods in the current namespace.

4. \*\*kubectl describe\*\*: Shows details of a specific resource.

- Example: `kubectl describe pod nginx`

Explanation: Provides detailed information about the pod named "nginx".

5. \*\*kubectl delete\*\*: Deletes resources by filenames, stdin, resources, and names, or by resources and label selector.

- Example: `kubectl delete pod nginx`

Explanation: Deletes the pod named "nginx".

6. \*\*kubectl exec\*\*: Executes a command in a container.

- Example: `kubectl exec -it <pod\_name> -- /bin/bash`

Explanation: Opens an interactive shell in the specified pod.

7. \*\*kubectl logs\*\*: Displays logs for a specific container in a pod.

- Example: `kubectl logs <pod\_name>`

Explanation: Retrieves logs from the default container in the specified pod.

8. \*\*kubectl scale\*\*: Changes the number of replicas of a resource.

- Example: `kubectl scale deployment/nginx --replicas=3`

Explanation: Scales the deployment named "nginx" to have 3 replicas.

9. \*\*kubectl rollout\*\*: Manages the rollout of a resource's changes.

- Example: `kubectl rollout status deployment/nginx`

Explanation: Checks the status of the rollout for the deployment named "nginx".

10. \*\*kubectl edit\*\*: Edits a resource live.

- Example: `kubectl edit pod/nginx`

Explanation: Opens the default editor to modify the configuration of the pod named "nginx" directly.

These are just a few examples of common Kubernetes commands and their usage. There are many more commands and options available depending on your specific use case and requirements.

Certainly! Here are some more Kubernetes commands along with examples and short explanations for each:

11. \*\*kubectl port-forward\*\*: Forward one or more local ports to a pod.

- Example: `kubectl port-forward pod/nginx 8080:80`

Explanation: Forwards local port 8080 to port 80 on pod "nginx", allowing access to the pod's service.

12. \*\*kubectl create namespace\*\*: Creates a new namespace.

- Example: `kubectl create namespace development`

Explanation: Creates a new namespace named "development" to logically isolate resources.

13. \*\*kubectl label\*\*: Adds or updates labels on resources.

- Example: `kubectl label pod/nginx app=web`

Explanation: Adds the label "app=web" to the pod named "nginx".

14. \*\*kubectl get events\*\*: Displays events related to resources.

- Example: `kubectl get events --sort-by='.metadata.creationTimestamp'`

Explanation: Retrieves events sorted by creation timestamp, providing insights into resource changes and status updates.

15. \*\*kubectl rollout restart\*\*: Restarts a rollout of an existing deployment.

- Example: `kubectl rollout restart deployment/nginx`

Explanation: Triggers a restart of the deployment named "nginx", effectively rolling out the latest changes.

16. \*\*kubectl exec -ti\*\*: Starts an interactive session in a specific container.

- Example: `kubectl exec -ti <pod\_name> -c <container\_name> -- /bin/bash`

Explanation: Initiates an interactive shell session in a specified container within a pod.

17. \*\*kubectl autoscale\*\*: Automatically scales a deployment based on CPU utilization.

- Example: `kubectl autoscale deployment/nginx --min=2 --max=5 --cpu-percent=80`

Explanation: Configures horizontal pod autoscaling for the deployment named "nginx" to maintain CPU utilization at 80%, with a minimum of 2 and a maximum of 5 replicas.

18. \*\*kubectl rollout history\*\*: Displays rollout history for a deployment.

- Example: `kubectl rollout history deployment/nginx`

Explanation: Retrieves the rollout history of the deployment named "nginx", showing revisions and their statuses.

19. \*\*kubectl get services\*\*: Retrieves information about services.

- Example: `kubectl get services`

Explanation: Lists all services in the current namespace, including their types, cluster IPs, and ports.

20. \*\*kubectl apply -f - --dry-run\*\*: Performs a dry run of a resource configuration file without actually applying it.

- Example: `kubectl apply -f deployment.yaml --dry-run=client`

Explanation: Validates the deployment configuration defined in "deployment.yaml" without making any changes to the cluster.

These additional Kubernetes commands cover various aspects of managing resources, deployments, scaling, and troubleshooting within a Kubernetes cluster.

Of course! Here are some more Kubernetes commands with examples and explanations:

21. \*\*kubectl create secret\*\*: Creates a secret using literal values, files, or standard input.

- Example: `kubectl create secret generic my-secret --from-literal=username=admin --from-literal=password=pass123`

Explanation: Creates a secret named "my-secret" with username and password data.

22. \*\*kubectl rollout pause\*\*: Pauses a rollout of an existing deployment.

- Example: `kubectl rollout pause deployment/nginx`

Explanation: Pauses the rollout of the deployment named "nginx", preventing further updates.

23. \*\*kubectl rollout resume\*\*: Resumes a paused rollout of a deployment.

- Example: `kubectl rollout resume deployment/nginx`

Explanation: Resumes the rollout of the deployment named "nginx" after it was paused.

24. \*\*kubectl annotate\*\*: Adds or updates annotations to resources.

- Example: `kubectl annotate pod/nginx description="Production pod"`

Explanation: Adds an annotation to the pod named "nginx" with a description.

25. \*\*kubectl get nodes\*\*: Retrieves information about nodes in the cluster.

- Example: `kubectl get nodes`

Explanation: Lists all nodes in the cluster along with their status, roles, and other details.

26. \*\*kubectl top\*\*: Displays resource usage statistics for nodes or pods.

- Example: `kubectl top pods`

Explanation: Shows CPU and memory usage of pods in the current namespace.

27. \*\*kubectl rollout undo\*\*: Rolls back a deployment to a previous revision.

- Example: `kubectl rollout undo deployment/nginx`

Explanation: Reverts the deployment named "nginx" to the previous revision, effectively undoing the latest rollout.

28. \*\*kubectl exec -ti --container\*\*: Starts an interactive session in a specific container within a pod.

- Example: `kubectl exec -ti <pod\_name> --container=<container\_name> -- /bin/bash`

Explanation: Initiates an interactive shell session in the specified container within a pod.

29. \*\*kubectl cp\*\*: Copies files and directories to and from containers.

- Example: `kubectl cp /local/path <namespace>/<pod\_name>:/remote/path`

Explanation: Copies files from the local machine to a specific path within a pod.

30. \*\*kubectl rollout status\*\*: Checks the status of a rollout for a deployment.

- Example: `kubectl rollout status deployment/nginx`

Explanation: Retrieves the status of the rollout for the deployment named "nginx", indicating whether it's in progress or has completed.

These commands provide additional functionalities for managing resources, troubleshooting, and interacting with Kubernetes clusters effectively.

kubectl create deployment nginx --image=nginx

kubectl create service nodeport nginx --tcp=80:80 --node-port=30080

kubectl create namespace test

kubectl create secret generic my-secret --from-literal=username=admin --from-literal=password=admin123

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

kubectl get pods

kubectl get services

kubectl get deployments

kubectl get nodes

kubectl get namespaces

kubectl describe pod my-pod

kubectl describe service my-service

kubectl describe deployment my-deployment

kubectl describe node my-node

kubectl describe namespace my-namespace

kubectl apply -f pod.yaml

kubectl apply -f deployment.yaml

kubectl apply -f service.yaml

kubectl apply -f configmap.yaml

kubectl apply -f secret.yaml

kubectl delete pod my-pod

kubectl delete service my-service

kubectl delete deployment my-deployment

kubectl delete node my-node

kubectl delete namespace my-namespace

Sure, here's a list of some common Kubernetes commands along with examples of how to use them:

1. \*\*kubectl get\*\*: This command is used to retrieve information about Kubernetes resources.

Example: `kubectl get pods`

2. \*\*kubectl create\*\*: Creates a resource from a file or from stdin.

Example: `kubectl create -f pod.yaml`

3. \*\*kubectl apply\*\*: Creates or updates a resource based on the configuration file.

Example: `kubectl apply -f deployment.yaml`

4. \*\*kubectl delete\*\*: Deletes resources by name or based on the information in a configuration file.

Example: `kubectl delete pod my-pod`

5. \*\*kubectl describe\*\*: Provides detailed information about a specific resource.

Example: `kubectl describe pod my-pod`

6. \*\*kubectl exec\*\*: Executes a command in a running container.

Example: `kubectl exec -it my-pod -- bash`

7. \*\*kubectl logs\*\*: Displays the logs from a container in a pod.

Example: `kubectl logs my-pod`

8. \*\*kubectl scale\*\*: Scales the number of replicas of a deployment, replica set, or stateful set.

Example: `kubectl scale deployment/my-deployment --replicas=3`

9. \*\*kubectl rollout\*\*: Manages the rollout of changes to a deployment.

Example: `kubectl rollout status deployment/my-deployment`

10. \*\*kubectl port-forward\*\*: Forwards one or more local ports to a pod.

Example: `kubectl port-forward my-pod 8080:80`

These are just a few of the many commands available in Kubernetes. They are essential for managing and interacting with your Kubernetes cluster and its resources.

Certainly! Here's a list of some common Kubernetes commands:

1. `kubectl create`: Create a resource from a file, URL, or stdin.

2. `kubectl apply`: Apply a configuration to a resource by filename or stdin.

3. `kubectl get`: Display one or many resources.

4. `kubectl describe`: Show details of a specific resource or group of resources.

5. `kubectl delete`: Delete resources by filenames, stdin, resources, and names, or by resources and label selector.

6. `kubectl exec`: Execute a command in a container.

7. `kubectl logs`: Print the logs for a container in a pod.

8. `kubectl scale`: Scale a resource.

9. `kubectl rollout`: Manage a rollout of a resource.

10. `kubectl edit`: Edit a resource on the server.

11. `kubectl label`: Add or update the labels of resources.

12. `kubectl annotate`: Add or update the annotations of resources.

13. `kubectl port-forward`: Forward one or more local ports to a pod.

14. `kubectl proxy`: Run a proxy to Kubernetes API server.

15. `kubectl cluster-info`: Display cluster info.

16. `kubectl config`: Modify kubeconfig files.

17. `kubectl version`: Print the client and server version information for the current context.

18. `kubectl top`: Display Resource (CPU/Memory/Storage) usage.

19. `kubectl apply -f`: Create or update resources from file.

These are just a few of the many commands available in Kubernetes. There are more specific commands for managing different types of resources, troubleshooting, debugging, and interacting with the Kubernetes cluster.

1. **kubectl create**: This command creates a new Kubernetes resource.
   * Example 1: kubectl create deployment nginx --image=nginx creates a deployment named "nginx" with the nginx container image.
   * Example 2: kubectl create namespace my-namespace creates a new namespace named "my-namespace".
   * Example 3: kubectl create -f pod.yaml creates a pod using the configuration specified in the YAML file.
   * Example 4: kubectl create service nodeport my-service --tcp=8080:80 creates a NodePort service named "my-service" exposing port 8080 on the nodes.
   * Example 5: kubectl create configmap my-config --from-file=config.txt creates a ConfigMap named "my-config" with data from the file "config.txt".
2. **kubectl apply**: This command creates or updates Kubernetes resources using a configuration file.
   * Example 1: kubectl apply -f deployment.yaml applies the configuration in the YAML file to create or update a deployment.
   * Example 2: kubectl apply -f service.yaml applies the configuration in the YAML file to create or update a service.
   * Example 3: kubectl apply -f namespace.yaml applies the configuration in the YAML file to create or update a namespace.
   * Example 4: kubectl apply -f secret.yaml applies the configuration in the YAML file to create or update a secret.
   * Example 5: kubectl apply -f configmap.yaml applies the configuration in the YAML file to create or update a configmap.
3. **kubectl get**: This command retrieves one or more Kubernetes resources.
   * Example 1: kubectl get pods retrieves a list of all pods in the current namespace.
   * Example 2: kubectl get nodes retrieves a list of all nodes in the cluster.
   * Example 3: kubectl get deployments retrieves a list of all deployments in the current namespace.
   * Example 4: kubectl get services retrieves a list of all services in the current namespace.
   * Example 5: kubectl get configmaps retrieves a list of all configmaps in the current namespace.
4. **kubectl describe**: This command provides detailed information about a Kubernetes resource.
   * Example 1: kubectl describe pod my-pod provides detailed information about the pod named "my-pod".
   * Example 2: kubectl describe node my-node provides detailed information about the node named "my-node".
   * Example 3: kubectl describe deployment my-deployment provides detailed information about the deployment named "my-deployment".
   * Example 4: kubectl describe service my-service provides detailed information about the service named "my-service".
   * Example 5: kubectl describe configmap my-configmap provides detailed information about the configmap named "my-configmap".
5. **kubectl delete**: This command deletes one or more Kubernetes resources.
   * Example 1: kubectl delete pod my-pod deletes the pod named "my-pod".
   * Example 2: kubectl delete deployment my-deployment deletes the deployment named "my-deployment".
   * Example 3: kubectl delete service my-service deletes the service named "my-service".
   * Example 4: kubectl delete namespace my-namespace deletes the namespace named "my-namespace" and all resources within it.
   * Example 5: kubectl delete -f pod.yaml deletes the pod specified in the YAML file.

1. **kubectl exec**: This command executes a command in a running container of a pod.
   * Example 1: kubectl exec -it my-pod -- /bin/bash opens a bash shell in the pod named "my-pod".
   * Example 2: kubectl exec my-pod -- ls /app lists the contents of the "/app" directory in the pod named "my-pod".
   * Example 3: kubectl exec my-pod -- wget http://example.com/file.txt -O /tmp/file.txt downloads a file from a URL into the "/tmp" directory in the pod named "my-pod".
   * Example 4: kubectl exec -it my-pod -- ps aux lists all processes running inside the container in the pod named "my-pod".
   * Example 5: kubectl exec my-pod -- curl http://localhost:8080/api sends a GET request to the specified URL from within the pod named "my-pod".
2. **kubectl logs**: This command prints the logs from a container in a pod.
   * Example 1: kubectl logs my-pod prints the logs from all containers in the pod named "my-pod".
   * Example 2: kubectl logs my-pod -c my-container prints the logs from the container named "my-container" in the pod named "my-pod".
   * Example 3: kubectl logs -f my-pod prints the logs from all containers in the pod named "my-pod" and continues to stream new logs as they are generated.
   * Example 4: kubectl logs --tail=100 my-pod prints the last 100 lines of logs from all containers in the pod named "my-pod".
   * Example 5: kubectl logs --since=1h my-pod prints logs from all containers in the pod named "my-pod" generated in the last hour.
3. **kubectl scale**: This command scales the number of replicas of a deployment, replica set, or stateful set.
   * Example 1: kubectl scale deployment/my-deployment --replicas=3 scales the deployment named "my-deployment" to have 3 replicas.
   * Example 2: kubectl scale rs/my-replicaset --replicas=5 scales the replica set named "my-replicaset" to have 5 replicas.
   * Example 3: kubectl scale statefulset/my-statefulset --replicas=2 scales the stateful set named "my-statefulset" to have 2 replicas.
   * Example 4: kubectl scale deployment/my-deployment --replicas=0 scales the deployment named "my-deployment" down to 0 replicas, effectively stopping it.
   * Example 5: kubectl scale deployment/my-deployment --replicas=1 scales the deployment named "my-deployment" to have 1 replica, effectively starting it if it was stopped.
4. **kubectl edit**: This command edits a Kubernetes resource in real-time using the default editor.
   * Example 1: kubectl edit deployment/my-deployment opens the default editor to edit the deployment named "my-deployment".
   * Example 2: kubectl edit configmap/my-configmap opens the default editor to edit the configmap named "my-configmap".
   * Example 3: kubectl edit pod/my-pod opens the default editor to edit the pod named "my-pod".
   * Example 4: kubectl edit service/my-service opens the default editor to edit the service named "my-service".
   * Example 5: kubectl edit secret/my-secret opens the default editor to edit the secret named "my-secret".

1. **kubectl port-forward**: This command creates a secure tunnel between a local machine and a pod, deployment, or service running on Kubernetes, allowing direct access to them.

* Example 1: kubectl port-forward pod/my-pod 8080:80 forwards local port 8080 to port 80 on pod "my-pod".
* Example 2: kubectl port-forward deployment/my-deployment 8080:80 forwards local port 8080 to port 80 on a pod managed by the deployment "my-deployment".
* Example 3: kubectl port-forward service/my-service 8080:80 forwards local port 8080 to port 80 on a pod selected by the service "my-service".
* Example 4: kubectl port-forward pod/my-pod 8080:80 9090:9090 forwards local port 8080 to port 80 and port 9090 to port 9090 on pod "my-pod".
* Example 5: kubectl port-forward deployment/my-deployment 8080:80 --context=my-context forwards local port 8080 to port 80 on a pod managed by the deployment "my-deployment" in the specified context.

1. **kubectl rollout**: This command manages rollouts of updates to Kubernetes resources.

* Example 1: kubectl rollout status deployment/my-deployment checks the status of the rollout for the deployment "my-deployment".
* Example 2: kubectl rollout history deployment/my-deployment displays the revision history of the deployment "my-deployment".
* Example 3: kubectl rollout undo deployment/my-deployment rolls back the deployment "my-deployment" to the previous revision.
* Example 4: kubectl rollout restart deployment/my-deployment restarts the rollout of the deployment "my-deployment".
* Example 5: kubectl rollout pause deployment/my-deployment pauses the rollout of the deployment "my-deployment".

1. **kubectl label**: This command adds or updates labels on Kubernetes resources.

* Example 1: kubectl label pods my-pod app=backend adds the label "app=backend" to the pod "my-pod".
* Example 2: kubectl label nodes my-node environment=production adds the label "environment=production" to the node "my-node".
* Example 3: kubectl label deployments my-deployment tier=frontend adds the label "tier=frontend" to the deployment "my-deployment".
* Example 4: kubectl label services my-service app=web adds the label "app=web" to the service "my-service".
* Example 5: kubectl label namespace my-namespace division=marketing adds the label "division=marketing" to the namespace "my-namespace".

1. **kubectl rollout**: This command is used to manage rollouts of updates to Kubernetes resources.

* Example 1: kubectl rollout status deployment/my-deployment checks the status of the rollout for the deployment named "my-deployment".
* Example 2: kubectl rollout history deployment/my-deployment displays the revision history of the deployment named "my-deployment".
* Example 3: kubectl rollout undo deployment/my-deployment rolls back the deployment named "my-deployment" to the previous revision.
* Example 4: kubectl rollout restart deployment/my-deployment restarts the rollout of the deployment named "my-deployment".
* Example 5: kubectl rollout pause deployment/my-deployment pauses the rollout of the deployment named "my-deployment".

1. **kubectl annotate**: This command adds or updates annotations on Kubernetes resources.

* Example 1: kubectl annotate pod/my-pod description="My Pod Description" adds an annotation with the key "description" and value "My Pod Description" to the pod named "my-pod".
* Example 2: kubectl annotate node/my-node environment=production adds an annotation with the key "environment" and value "production" to the node named "my-node".
* Example 3: kubectl annotate deployment/my-deployment owner=John adds an annotation with the key "owner" and value "John" to the deployment named "my-deployment".
* Example 4: kubectl annotate service/my-service app=web adds an annotation with the key "app" and value "web" to the service named "my-service".
* Example 5: kubectl annotate namespace/my-namespace division=marketing adds an annotation with the key "division" and value "marketing" to the namespace named "my-namespace".

1. **kubectl rollout**: This command manages rollouts of updates to Kubernetes resources.

* Example 1: kubectl rollout status deployment/my-deployment checks the status of the rollout for the deployment named "my-deployment".
* Example 2: kubectl rollout history deployment/my-deployment displays the revision history of the deployment named "my-deployment".
* Example 3: kubectl rollout undo deployment/my-deployment rolls back the deployment named "my-deployment" to the previous revision.
* Example 4: kubectl rollout restart deployment/my-deployment restarts the rollout of the deployment named "my-deployment".
* Example 5: kubectl rollout pause deployment/my-deployment pauses the rollout of the deployment named "my-deployment".

1. **kubectl scale**: This command scales the number of replicas of a deployment, replica set, or stateful set.

* Example 1: kubectl scale deployment/my-deployment --replicas=3 scales the deployment named "my-deployment" to have 3 replicas.
* Example 2: kubectl scale rs/my-replicaset --replicas=5 scales the replica set named "my-replicaset" to have 5 replicas.
* Example 3: kubectl scale statefulset/my-statefulset --replicas=2 scales the stateful set named "my-statefulset" to have 2 replicas.
* Example 4: kubectl scale deployment/my-deployment --replicas=0 scales the deployment named "my-deployment" down to 0 replicas, effectively stopping it.
* Example 5: kubectl scale deployment/my-deployment --replicas=1 scales the deployment named "my-deployment" to have 1 replica, effectively starting it if it was stopped.

1. **kubectl edit**: This command edits a Kubernetes resource in real-time using the default editor.

* Example 1: kubectl edit deployment/my-deployment opens the default editor to edit the deployment named "my-deployment".
* Example 2: kubectl edit configmap/my-configmap opens the default editor to edit the configmap named "my-configmap".
* Example 3: kubectl edit pod/my-pod opens the default editor to edit the pod named "my-pod".
* Example 4: kubectl edit service/my-service opens the default editor to edit the service named "my-service".
* Example 5: kubectl edit secret/my-secret opens the default editor to edit the secret named "my-secret".

1. **kubectl label**: This command adds or updates labels on Kubernetes resources.

* Example 1: kubectl label pods my-pod app=backend adds the label "app=backend" to the pod "my-pod".
* Example 2: kubectl label nodes my-node environment=production adds the label "environment=production" to the node "my-node".
* Example 3: kubectl label deployments my-deployment tier=frontend adds the label "tier=frontend" to the deployment "my-deployment".
* Example 4: kubectl label services my-service app=web adds the label "app=web" to the service "my-service".
* Example 5: kubectl label namespace my-namespace division=marketing adds the label "division=marketing" to the namespace "my-namespace".

1. **kubectl exec**: This command executes a command in a running container of a pod.

* Example 1: kubectl exec -it my-pod -- /bin/bash opens a bash shell in the pod named "my-pod".
* Example 2: kubectl exec my-pod -- ls /app lists the contents of the "/app" directory in the pod named "my-pod".
* Example 3: kubectl exec my-pod -- wget http://example.com/file.txt -O /tmp/file.txt downloads a file from a URL into the "/tmp" directory in the pod named "my-pod".
* Example 4: kubectl exec -it my-pod -- ps aux lists all processes running inside the container in the pod named "my-pod".
* Example 5: kubectl exec my-pod -- curl http://localhost:8080/api sends a GET request to the specified URL from within the pod named "my-pod".

1. **kubectl logs**: This command prints the logs from a container in a pod.

* Example 1: kubectl logs my-pod prints the logs from all containers in the pod named "my-pod".
* Example 2: kubectl logs my-pod -c my-container prints the logs from the container named "my-container" in the pod named "my-pod".
* Example 3: kubectl logs -f my-pod prints the logs from all containers in the pod named "my-pod" and continues to stream new logs as they are generated.
* Example 4: kubectl logs --tail=100 my-pod prints the last 100 lines of logs from all containers in the pod named "my-pod".
* Example 5: kubectl logs --since=1h my-pod prints logs from all containers in the pod named "my-pod" generated in the last hour.

1. **kubectl delete**: This command deletes one or more Kubernetes resources.

* Example 1: kubectl delete pod my-pod deletes the pod named "my-pod".
* Example 2: kubectl delete deployment my-deployment deletes the deployment named "my-deployment".
* Example 3: kubectl delete service my-service deletes the service named "my-service".
* Example 4: kubectl delete namespace my-namespace deletes the namespace named "my-namespace" and all resources within it.
* Example 5: kubectl delete -f pod.yaml deletes the pod specified in the YAML file.

1. **kubectl rollout**: This command manages rollouts of updates to Kubernetes resources.

* Example 1: kubectl rollout status deployment/my-deployment checks the status of the rollout for the deployment named "my-deployment".
* Example 2: kubectl rollout history deployment/my-deployment displays the revision history of the deployment named "my-deployment".
* Example 3: kubectl rollout undo deployment/my-deployment rolls back the deployment named "my-deployment" to the previous revision.
* Example 4: kubectl rollout restart deployment/my-deployment restarts the rollout of the deployment named "my-deployment".
* Example 5: kubectl rollout pause deployment/my-deployment pauses the rollout of the deployment named "my-deployment".

1. **kubectl annotate**: This command adds or updates annotations on Kubernetes resources.

* Example 1: kubectl annotate pod/my-pod description="My Pod Description" adds an annotation with the key "description" and value "My Pod Description" to the pod named "my-pod".
* Example 2: kubectl annotate node/my-node environment=production adds an annotation with the key "environment" and value "production" to the node named "my-node".
* Example 3: kubectl annotate deployment/my-deployment owner=John adds an annotation with the key "owner" and value "John" to the deployment named "my-deployment".
* Example 4: kubectl annotate service/my-service app=web adds an annotation with the key "app" and value "web" to the service named "my-service".
* Example 5: kubectl annotate namespace/my-namespace division=marketing adds an annotation with the key "division" and value "marketing" to the namespace named "my-namespace".

1. **kubectl rollout**: This command manages rollouts of updates to Kubernetes resources.

* Example 1: kubectl rollout status deployment/my-deployment checks the status of the rollout for the deployment named "my-deployment".
* Example 2: kubectl rollout history deployment/my-deployment displays the revision history of the deployment named "my-deployment".
* Example 3: kubectl rollout undo deployment/my-deployment rolls back the deployment named "my-deployment" to the previous revision.
* Example 4: kubectl rollout restart deployment/my-deployment restarts the rollout of the deployment named "my-deployment".
* Example 5: kubectl rollout pause deployment/my-deployment pauses the rollout of the deployment named "my-deployment".

1. **kubectl create**: This command creates a new Kubernetes resource.

* Example 1: kubectl create deployment nginx --image=nginx creates a deployment named "nginx" with the nginx container image.
* Example 2: kubectl create namespace my-namespace creates a new namespace named "my-namespace".
* Example 3: kubectl create -f pod.yaml creates a pod using the configuration specified in the YAML file.
* Example 4: kubectl create service nodeport my-service --tcp=8080:80 creates a NodePort service named "my-service" exposing port 8080 on the nodes.
* Example 5: kubectl create configmap my-config --from-file=config.txt creates a ConfigMap named "my-config" with data from the file "config.txt".

1. **kubectl apply**: This command creates or updates Kubernetes resources using a configuration file.

* Example 1: kubectl apply -f deployment.yaml applies the configuration in the YAML file to create or update a deployment.
* Example 2: kubectl apply -f service.yaml applies the configuration in the YAML file to create or update a service.
* Example 3: kubectl apply -f namespace.yaml applies the configuration in the YAML file to create or update a namespace.
* Example 4: kubectl apply -f secret.yaml applies the configuration in the YAML file to create or update a secret.
* Example 5: kubectl apply -f configmap.yaml applies the configuration in the YAML file to create or update a configmap.

1. **kubectl get**: This command retrieves one or more Kubernetes resources.

* Example 1: kubectl get pods retrieves a list of all pods in the current namespace.
* Example 2: kubectl get nodes retrieves a list of all nodes in the cluster.
* Example 3: kubectl get deployments retrieves a list of all deployments in the current namespace.
* Example 4: kubectl get services retrieves a list of all services in the current namespace.
* Example 5: kubectl get configmaps retrieves a list of all configmaps in the current namespace.

1. **kubectl describe**: This command provides detailed information about a Kubernetes resource.

* Example 1: kubectl describe pod my-pod provides detailed information about the pod named "my-pod".
* Example 2: kubectl describe node my-node provides detailed information about the node named "my-node".
* Example 3: kubectl describe deployment my-deployment provides detailed information about the deployment named "my-deployment".
* Example 4: kubectl describe service my-service provides detailed information about the service named "my-service".
* Example 5: kubectl describe configmap my-configmap provides detailed information about the configmap named "my-configmap".

1. **kubectl edit**: This command edits a Kubernetes resource in real-time using the default editor.

* Example 1: kubectl edit deployment/my-deployment opens the default editor to edit the deployment named "my-deployment".
* Example 2: kubectl edit configmap/my-configmap opens the default editor to edit the configmap named "my-configmap".
* Example 3: kubectl edit pod/my-pod opens the default editor to edit the pod named "my-pod".
* Example 4: kubectl edit service/my-service opens the default editor to edit the service named "my-service".
* Example 5: kubectl edit secret/my-secret opens the default editor to edit the secret named "my-secret".

1. **kubectl rollout**: This command manages rollouts of updates to Kubernetes resources.

* Example 1: kubectl rollout status deployment/my-deployment checks the status of the rollout for the deployment named "my-deployment".
* Example 2: kubectl rollout history deployment/my-deployment displays the revision history of the deployment named "my-deployment".
* Example 3: kubectl rollout undo deployment/my-deployment rolls back the deployment named "my-deployment" to the previous revision.
* Example 4: kubectl rollout restart deployment/my-deployment restarts the rollout of the deployment named "my-deployment".
* Example 5: kubectl rollout pause deployment/my-deployment pauses the rollout of the deployment named "my-deployment".

1. **kubectl label**: This command adds or updates labels on Kubernetes resources.

* Example 1: kubectl label pods my-pod app=backend adds the label "app=backend" to the pod "my-pod".
* Example 2: kubectl label nodes my-node environment=production adds the label "environment=production" to the node "my-node".
* Example 3: kubectl label deployments my-deployment tier=frontend adds the label "tier=frontend" to the deployment "my-deployment".
* Example 4: kubectl label services my-service app=web adds the label "app=web" to the service "my-service".
* Example 5: kubectl label namespace my-namespace division=marketing adds the label "division=marketing" to the namespace "my-namespace".

1. **kubectl exec**: This command executes a command in a running container of a pod.

* Example 1: kubectl exec -it my-pod -- /bin/bash opens a bash shell in the pod named "my-pod".
* Example 2: kubectl exec my-pod -- ls /app lists the contents of the "/app" directory in the pod named "my-pod".
* Example 3: kubectl exec my-pod -- wget http://example.com/file.txt -O /tmp/file.txt downloads a file from a URL into the "/tmp" directory in the pod named "my-pod".
* Example 4: kubectl exec -it my-pod -- ps aux lists all processes running inside the container in the pod named "my-pod".
* Example 5: kubectl exec my-pod -- curl http://localhost:8080/api sends a GET request to the specified URL from within the pod named "my-pod".