In Kubernetes, a ConfigMap is an object that stores configuration data as key-value pairs. ConfigMaps allow you to decouple configuration data from application code, so that you can change the configuration data without having to rebuild the application image or redeploy the application.

ConfigMaps can be used to store any kind of configuration data, such as server addresses, database connection strings, environment variables, or command-line arguments. ConfigMaps can be used to configure applications that run in containers, as well as Kubernetes itself.

To use a ConfigMap in a container, you can pass the ConfigMap data to the container using environment variables or command-line arguments. You can also mount the ConfigMap data as files or directories in the container's filesystem.

ConfigMaps are created using a YAML or JSON file that specifies the key-value pairs to be stored in the ConfigMap. The ConfigMap can then be referenced in a pod's spec section using the env or volumes fields.

ConfigMaps are part of the Kubernetes API and can be managed using kubectl commands, as well as through the Kubernetes dashboard or API.

---

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: mongodb-configmap
data:
  database_url: mongodb-service


#database_url = mongodb_service(server name is name of the service)
```

This YAML file defines a Kubernetes ConfigMap object named mongodb-configmap that contains a single key-value pair. The key is database_url and the value is mongodb-service.

This ConfigMap can be used to store the URL of a MongoDB server that is running in a Kubernetes cluster. The URL is stored as a value in the mongodb-configmap ConfigMap with the key database_url.

To use this ConfigMap in a container, you can mount it as a volume or pass the database_url value to the container as an environment variable.