Kubernetes Concepts: Pods, Services, Deployments, and CRDs

Kubernetes (K8s) is a powerful container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. While Kubernetes offers a wide range of features, understanding its core building blocks is essential to effectively working with it. This write-up focuses on four fundamental concepts: **Pods**, **Services**, **Deployments**, and **Custom Resource Definitions (CRDs)**.

🌐 2. Services

Definition: A Service is an abstraction that exposes a set of Pods as a network service.

Types of Services:

- **ClusterIP**: Exposes the service internally within the cluster (default).
- **NodePort**: Exposes the service on each Node's IP at a static port.
- **LoadBalancer**: Provisions an external load balancer (cloud environments).
- **ExternalName**: Maps the service to an external DNS name.

Benefits:

- Provides a stable endpoint for a set of ephemeral Pods.
- Performs basic load balancing.

Example YAML:

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 8080
  type: ClusterIP
```

⚙️ 4. Custom Resource Definitions (CRDs)

Definition: CRDs allow users to extend Kubernetes by defining their own resource types.

Use Case:

- Build custom controllers/operators.
- Add domain-specific APIs (e.g., `MySQLCluster`, `KafkaTopic`).

Workflow:

1. Define a CRD.
2. Deploy a controller to manage the lifecycle of the custom resource.
3. Use `kubectl` and the Kubernetes API to create/manage these new types.

Example YAML:

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
```

```
metadata:
  name: crontabs.stable.example.com
spec:
  group: stable.example.com
  versions:
    - name: v1
      served: true
      storage: true
      schema:
        openAPIV3Schema:
          type: object
          properties:
            spec:
              type: object
              properties:
                cronSpec:
                  type: string
                image:
                  type: string
  scope: Namespaced
  names:
    plural: crontabs
    singular: crontab
    kind: CronTab
    shortNames:
    - ct
```

✅ Conclusion

Kubernetes provides a modular and extensible system through these primitives. Understanding **Pods** for container execution, **Services** for networking, **Deployments** for declarative management, and **CRDs** for custom workflows equips you to build robust, scalable systems on Kubernetes.