kubernetes集群中部署openLooKeng操作指导

# 获取openLooKeng镜像

1. docker镜像下载路径：

x86： <https://download.openlookeng.io/dockerimages/>openlookeng-x86-docker.tar

arm： <https://download.openlookeng.io/dockerimages/>openlookeng-arm-docker.tar

1. 将镜像加载到本地：

docker load -i <镜像文件名>

# 配置部署文件

1. 获取配置模板

配置文件下载路径：<https://download.openlookeng.io/dockerimages/>deploy-openlk.tar

解压后配置模板目录结构如下：

openlk

----Chart.yaml

----config-catalog：数据源配置文件及其依赖文件存放目录

----config-coordinator：coordinator节点配置文件存放目录

----config-coordinator-ha：HA集群的coordinator节点配置文件存放目录

----config-filesystem：文件系统相关配置存放目录

----config-worker：worker节点配置文件存放目录

----config-worker-ha：HA集群的worker节点配置文件存放目录

----README.md

----templates：该目录下有配置文件deployment.yaml

----values.yaml：部署集群时相关参数在该文件中设置

1. 修改配置文件

**values.yaml文件**

docker:

 registry: ''

 imageTag: unknown -----------此处修改为openlookeng镜像的TAG

 .................

configMap:

 coordinator:

 name: config-coordinator

 srcPath: config-coordinator -------------可改为config-coordinator-ha使用HA的配置

 worker:

 name: config-worker

 srcPath: config-worker -------------可改为config-worker-ha使用HA的配置

 catalog:

 name: config-catalog

 srcPath: config-catalog

 filesystem:

 name: config-filesystem

 srcPath: config-filesystem

 .................

hpa:

 coordinator:

 enabled: false --------为true时开启coordinator节点自动伸缩功能

 minReplicas: 1 --------coordinator的pod的最小个数

 maxReplicas: 2 --------coordinator的pod的最大个数

 cpu:

 averageUtilization: 50 ------coordinator的伸缩条件为cpu的50%

 stabilizationWindowSeconds: 300

 worker:

 enabled: false --------为true时开启worker节点自动伸缩功能

 minReplicas: 1 --------worker的pod的最小个数

 maxReplicas: 5 --------worker的pod的最大个数

 cpu:

 averageUtilization: 50 ------worker的伸缩条件为cpu的50%

 stabilizationWindowSeconds: 300

**templates/deployment.yaml文件**

#### {{ $key }} ####

---

apiVersion: apps/v1

kind: Deployment

metadata:

 name: {{ $key }}

 namespace: {{ $namespace }}

spec:

 replicas: {{ .replica }}

 selector:

 matchLabels:

 app: {{ $key }}

 strategy:

 type: RollingUpdate

 rollingUpdate:

 maxSurge: 25%

 maxUnavailable: 0%

 template:

 metadata:

 labels:

 app: {{ $key }}

 annotations:

 timestamp: "{{ date "20060102150405" $.Release.Time }}"

 spec:

 {{- if eq $key "worker"}}

 # Coexist on the same node with the coordinator from the same namespace

 affinity:

 podAffinity:

 requiredDuringSchedulingIgnoredDuringExecution:

 - labelSelector:

 matchLabels:

 app: coordinator

 topologyKey: kubernetes.io/hostname

 {{- end}}

 {{- if .gracefulShutdown }}

 terminationGracePeriodSeconds: {{ .terminationGracePeriodSeconds }}

 {{- end }}

 {{- if ne $.Values.env "local" }}

 imagePullSecrets:

 - name: default-secret

 {{- end }}

 containers:

 - name: {{ $key }}

 image: {{ $.Values.docker.registry }}openlookeng:{{ $.Values.docker.imageTag }}

 imagePullPolicy: IfNotPresent

 args: ["-t", "{{ $key }}", "-configDir", "/customConfig", "-jvmXmx", "3500M"] ---------3500M分配给jvm的内存，可根据需要修改

 resources:

 requests:

 cpu: 500m -------cn和worker运行时需要的cpu，可根据需要修改

 memory: 1G -------cn和worker运行时需要的内存，可根据需要修改

 limits:

 cpu: 2 -----cn和worker运行时可使用的最大cpu，可根据需要修改

 memory: 4G -----cn和worker运行时可使用的最大内存，可根据需要修改

 {{- if .gracefulShutdown }}

 lifecycle:

 preStop:

 exec:

 command:

 - /usr/lib/hetu/bin/shutdown-hetu

 {{- end }}

 env:

 - name: CLUSTER\_ID

 value: "{{ $namespace }}"

 volumeMounts:

 {{- range $volume, $map := .volume }}

 {{- with $map }}

 - mountPath: {{ .mountPath }}

 name: {{ .name }}

 {{- end }}

 {{- end }}

 volumes:

 {{- range $volume, $map := .volume }}

 {{- with $map }}

 - name: {{ .name }}

 configMap:

 name: {{ .name }}

 optional: {{ .optional }}

 {{- end }}

 {{- end }}

{{- end }}

{{- end }}

{{- range $key, $val := .Values.hpa }}

{{- with $val }}

{{- if .enabled }}

#### HPA for {{ $key }} ####

---

apiVersion: autoscaling/v2beta2

kind: HorizontalPodAutoscaler

...............

 metrics:

 - type: Resource

 resource:

 name: cpu

 target:

 type: Utilization

 averageUtilization: {{ .cpu.averageUtilization }}

 # TODO: other scaling conditions, e.g. memory

# behavior:

# scaleDown:

# stabilizationWindowSeconds: {{ .stabilizationWindowSeconds }}

（注：kubernetes版本为1.17以下时，将最后三行注释掉，如上）

**若要调整openLooKeng服务的参数配置，修改kubernetes/openlk/config-\*目录下的配置文件。**

# 部署openlookeng

1. 部署openlookeng

openlk目录下执行：helm upgrade --install openlk .

（卸载：helm delete openlk）

1. 验证

*[root@openlookeng-qatest ~]# kubectl get pods -n openlk*

*NAME READY STATUS RESTARTS AGE*

*coordinator-6446df699b-ngxs7 1/1 Running 0 39m*

*worker-56c975744f-64lmm 1/1 Running 0 39m*

# openLooKeng的使用

执行命令进入pod：

kubectl exec -ti <coordinator-pod-name> -n openlk -- /bin/bash

再执行命令登录openlookeng客户端即可执行sql：

openlk

*[root@openlookeng-qatest ~]# kubectl exec -ti coordinator-6446df699b-ngxs7 -A -- /bin/bash*

*[openlkadmin@coordinator-6446df699b-ngxs7 /]$ openlk*

*lk> show catalogs;*

*......*