Instituto Tecnológico de Costa Rica

IC4302 - Bases de Datos II

Documentación Proyecto 2

Profesor: Nereo Campos Araya

Estudiantes:

- Fiorella Zelaya Coto - 2021453615
- Isaac Araya Solano - 2018151703
- Melany Salas Fernández - 2021121147
- Moisés Solano Espinoza - 2021144322
- Pablo Arias Navarro - 2021024635

Instrucciones para ejecutar el proyecto

Pruebas

MongoDB

Backup

Restauración

MariaDB

Backup

Restauración

PostgreSQL

Backup

Restauración

ElasticSearch

Backup

Restauración

Neo4j

Backup

Restauración

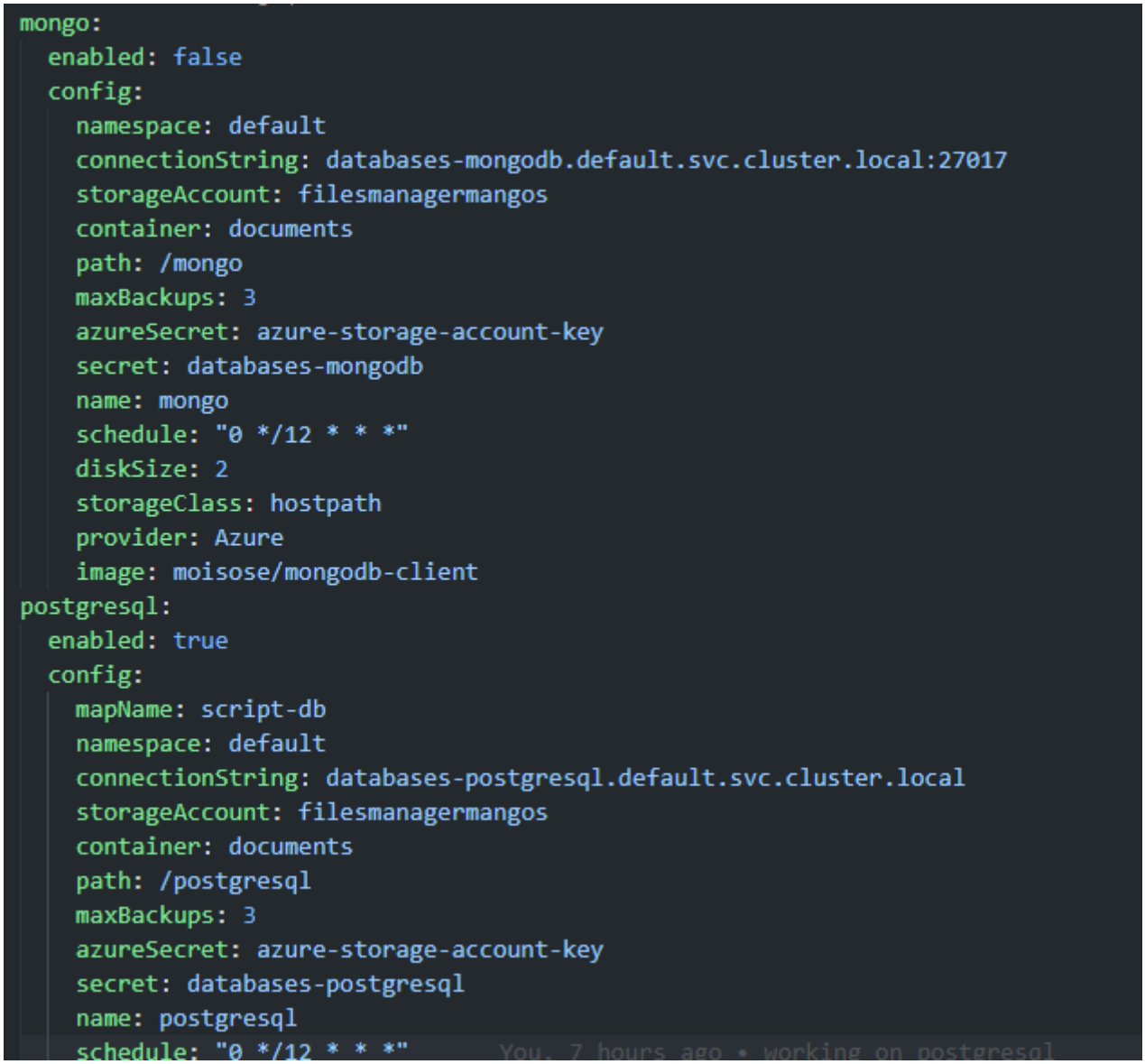
CouchDB

Backup

Restauración

Componentes

Para cada una de las bases de datos, se agregan los valores necesarios en el archivo "values.yaml", de la carpeta templetes, en la sección de backups.



```yaml
mongo:
  enabled: false
  config:
    namespace: default
    connectionString: databases-mongodb.default.svc.cluster.local:27017
    storageAccount: filesmanagermangos
    container: documents
    path: /mongo
    maxBackups: 3
    azureSecret: azure-storage-account-key
    secret: databases-mongodb
    name: mongo
    schedule: "0 */12 * * *"
    diskSize: 2
    storageClass: hostpath
    provider: Azure
    image: moisose/mongodb-client
postgresql:
  enabled: true
  config:
    mapName: script-db
    namespace: default
    connectionString: databases-postgresql.default.svc.cluster.local
    storageAccount: filesmanagermangos
    container: documents
    path: /postgresql
    maxBackups: 3
    azureSecret: azure-storage-account-key
    secret: databases-postgresql
    name: postgresql
    schedule: "0 */12 * * *"      You, 7 hours ago • working on postgresql
```

MongoDB

Para el backup de MongoDB se utilizó el archivo "backup.sh" en el que se realizan:

- ConfigMap: Se guardan aspectos de configuración, como el namestapce y el script correspondiente.

```
{{- if .Values.mongo.enabled }}      You, last week • Archivos

apiVersion: v1
kind: ConfigMap
metadata:
  name: mongo-backups-script
  namespace: {{ .Values.namespace }}
data:
{{ (.Files.Glob "scripts/mongodb.sh").AsConfig | indent 2 }}
```

- PersistentVolumeClaim: Se solicita el almacenamiento necesario y se define el modo de lectura-escritura.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-snapshotter
  namespace: {{ .Values.mongo.config.namespace }}
spec:
  storageClassName: {{ .Values.mongo.config.storageClass }}
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: {{ .Values.mongo.config.diskSize }}
```

- CronJob: Utilizado para automarizar los backups, tambien se definen variables de entorno y demás.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-snapshotter
  namespace: {{ .Values.mongo.config.namespace }}
spec:
  storageClassName: {{ .Values.mongo.config.storageClass }}
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: {{ .Values.mongo.config.diskSize }}
```

- Job: Se crea el pod para realizar la tarea del respaldo de MongoDB, tambien se definen variables de entorno y demás.

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: mongo-backup-0
  namespace: {{ .Values.mongo.config.namespace }}
spec:
  template:
    spec:
      serviceAccountName: {{ .Values.mongo.config.serviceAccount }}
      volumes:
      - name: scripts
        configMap:
          name: mongo-backups-script
          defaultMode: 0777
      - name: mongo-snapshotter
        persistentVolumeClaim:
          claimName: mongo-snapshotter
      containers:
        - image: mcr.microsoft.com/azure-cli
          name: mongo-snapshotter
          env:
            - name: MONGO_CONNECTION_STRING
              value: "{{ .Values.mongo.config.connectionString }}"
            - name: AZURE_STORAGE_ACCOUNT
              value: "{{ .Values.mongo.config.storageAccount }}"
```

Por otro lado, en este mismo archivo se definen un configmap y un job para hacer el cargado de la base de datos y los datos de prueba necesarios para comprobar el funcionamiento del backup.

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: mongo-backup-0
  namespace: {{ .Values.mongo.config.namespace }}
spec:
  template:
    spec:
      serviceAccountName: {{ .Values.mongo.config.serviceAccount }}
      volumes:
      - name: scripts
        configMap:
          name: mongo-backups-script
          defaultMode: 0777
      - name: mongo-snapshotter
        persistentVolumeClaim:
          claimName: mongo-snapshotter
      containers:
        - image: mcr.microsoft.com/azure-cli
          name: mongo-snapshotter
          env:
            - name: MONGO_CONNECTION_STRING
              value: "{{ .Values.mongo.config.connectionString }}"
            - name: AZURE_STORAGE_ACCOUNT
              value: "{{ .Values.mongo.config.storageAccount }}"
```

Backup

```bash
#!/bin/bash
# get the current date and time
DATE=$(date '+%Y%m%d%H%M')
# create a directory with the current date
# the [-p] arg is used bs /mongodump may not exist
mkdir -p /mongodump/$DATE
apk update
apk upgrade
# The MongoDB tools provide import, export, and diagnostic capabilities.
apk add  mongodb-tools
# Azure CLI command for enable dynamic install without a prompt.
az config set extension.use_dynamic_install=yes_without_prompt
# mongodump configuration for connect to an instance
# --host, -u or --username, -p or --password, --gzip(compress the output), --archive(Writes the output to a specified archive file)
mongodump --host="$MONGO_CONNECTION_STRING" -u $MONGO_USERNAME -p $MONGO_PASSWORD --gzip --archive=/mongodump/$DATE
az storage blob directory upload --container $CONTAINER -s /mongodump/$DATE -d $BACKUP_PATH --auth-mode key --recursive
rm -rf /mongodump/$DATE
```

Para el script del backup de MongoDB se optiene la fecha, y se crea el directorio (en caso de que no exista), ademas, se hace un update de los paquetes y se importa **mongodb-tools**, este brinda las herramientas necesarias para manejar Mongo y poder hacer el dump.

Posteriormente se hace el **mongodump** a Mongo mediante el conection string, el username y el password.

Finalmente, se sube el dump al blobstorage de Azure.

Restauración

MariaDB

Backup

Restauración

PostgreSQL

Para el backup de PostgreSQL se utilizó el archivo "backuppostgresql.yaml", en el que se realizan:

- ConfigMap: Se guardan aspectos de configuración, como el namestapce y el script correspondiente.

```
{{ if .Values.postgresql.enabled }}
apiVersion: v1
kind: ConfigMap
metadata:
  name: postgresql-backups-script
  namespace: {{ .Values.namespace }}
data:
{{ (.Files.Glob "scripts/postgresql.sh").AsConfig | indent 2 }}
```

- PersistentVolumeClaim: Se solicita el almacenamiento necesario y se define el modo de lectura-escritura.

```
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: postgresql-snapshotter
  namespace: {{ .Values.postgresql.config.namespace }}
spec:
  storageClassName: {{ .Values.postgresql.config.storageClass }}
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: {{ .Values.postgresql.config.diskSize }}
```

- CronJob: Utilizado para automarizar los backups, tambien se definen variables de entorno y demás.

```yaml
---
apiVersion: batch/v1
kind: CronJob
metadata:
  name: "postgresql-backup"
  namespace: {{ .Values.postgresql.config.namespace }}
spec:
  schedule: {{ .Values.postgresql.config.schedule }}
  successfulJobsHistoryLimit: 1
  failedJobsHistoryLimit: 1
  jobTemplate:
    spec:
      backoffLimit: 3
      template:
        spec:
          terminationGracePeriodSeconds: 0
          restartPolicy: Never
          volumes:
          - name: scripts
            configMap:      You, 7 hours ago • working on postgresql
              name: postgresql-backups-script
              defaultMode: 0777
          - name: postgresql-snapshotter
            persistentVolumeClaim:
              claimName: postgresql-snapshotter
          containers:
            - image: mcr.microsoft.com/azure-cli
              name: postgresql-snapshotter
              env:
                - name: POSTGRESQL_CONNECTION_STRING
                  value: "{{ .Values.postgresql.config.connectionString }}"
                - name: AZURE_STORAGE_ACCOUNT
```

- Job: Se crea el pod para realizar la tarea del respaldo de PostgreSQL, tambien se definen variables de entorno y demás. Para la parte de la restauración, tambien se crea un job que se encarga de realizar la tarea de restaurar la base de datos.

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: postgresql-backup-0
  namespace: {{ .Values.postgresql.config.namespace }}
spec:
  template:
    spec:
      serviceAccountName: {{ .Values.postgresql.config.serviceAccount }}
      volumes:
      - name: scripts
        configMap:
          name: postgresql-backups-script
          defaultMode: 0777
      - name: postgresql-snapshotter
        persistentVolumeClaim:
          claimName: postgresql-snapshotter
      containers:
        - image: mcr.microsoft.com/azure-cli
          name: postgresql-snapshotter
          env:
            - name: POSTGRESQL_CONNECTION_STRING
              value: "{{ .Values.postgresql.config.connectionString }}"
            - name: AZURE_STORAGE_ACCOUNT
              value: "{{ .Values.postgresql.config.storageAccount }}"
            - name: CONTAINER
              value: "{{ .Values.postgresql.config.container }}"
            - name: BACKUP_PATH
              value: "{{ .Values.postgresql.config.path }}"
            - name: MAX_BACKUPS       You, 7 hours ago • working on postgresql
              value: "{{ .Values.postgresql.config.maxBackups }}"
            - name: DB_HOST
              value: "databases-postgresql"
            - name: POSTGRESQL_USERNAME
```

Por otro lado, en este mismo archivo se definen un configmap y un job para hacer el cargado de la base de datos y los datos de prueba necesarios para comprobar el funcionamiento del backup.

```yaml
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: postgresql-data
  namespace: {{ .Values.namespace }}
data:
{{ (.Files.Glob "sql/postgres.sql").AsConfig | indent 2 }}
---
apiVersion: batch/v1
kind: Job
metadata:
  name: postgresql-data-0
spec:
  template:
    spec:
      volumes:
      - name: scripts
        configMap:
          name: postgresql-data
          defaultMode: 0777
      containers:
      - name: pi
        image: {{ .Values.postgresql.config.image }}
        volumeMounts:
        - name: scripts
          mountPath: /scripts
        env:
          - name: POSTGRESQL_CONNECTION_STRING
```

Backup

```bash
#!/bin/bash
# get the current date and time
DATE=$(date '+%Y%m%d%H%M')
# create a directory with the current date
# the [-p] arg is used bs /pgdump may not exist
mkdir -p /pgdump/$DATE
# Update the packages
apk update
apk upgrade
# Install the PostgreSQL client
apk add postgresql-client
# Azure CLI command for enable dynamic install without a prompt.
az config set extension.use_dynamic_install=yes_without_prompt
# pg_dump configuration for connect to an instance
PGPASSWORD="$POSTGRESQL_PASSWORD" pg_dumpall --host $DB_HOST -U $POSTGRESQL_USERNAME --file=/pgdump/$DATE/db_backup.dump
# Upload the backup to the blob storage
az storage blob directory upload --container $CONTAINER -s /pgdump/$DATE -d $BACKUP_PATH --auth-mode key --recursive
rm -rf /pgdump/$DATE
```

Para el script del backup de PostGreSQL se optiene la fecha, y se crea el directorio (en caso de que no exista), ademas, se hace un update de los paquetes y se importa **postgresql-client**, este brinda las herramientas necesarias para manejar postgresql y poder hacer el dump.

Posteriormente se hace el **pg_dump** a la base de datos en postgresql mendiante el username, el password y el host.

Finalmente, se sube el dump al blobstorage de Azure.

Restauración

```
#!/bin/bash
# Configuration variables
BACKUP_NAME="202306152142"       # folder name in the blob storage
ARCHIVE_NAME="db_backup.dump"    # file name in the blob storage
# Connection string for Azure Blob Storage
CONNECTION_STRING_AZURE="DefaultEndpointsProtocol=https;AccountName=filesmanagermangos;AccountKey=71ms2t3YFnW7Qu4KllgC1PR5
# Creates the directory
mkdir -p postgresqlrestore/$BACKUP_NAME
# Update and upgrade the packages
apk update
apk upgrade
apk add postgresql-client
# Azure CLI command for enable dynamic install without a prompt.
az config set extension.use_dynamic_install=yes_without_prompt
# Dowlnoad the backup from the blob storage
az storage blob download --container $CONTAINER --name postgresql/$BACKUP_NAME/$ARCHIVE_NAME --file postgresqlrestore/$BAC
# Restore a PostgreSQL database from a backup using psql
PGPASSWORD="$POSTGRESQL_PASSWORD" psql --set ON_ERROR_STOP=off -h $DB_HOST -U $POSTGRESQL_USERNAME -f postgresqlrestore/$B
```

Para el script de restauración tambien se crea un directorio, se hace un update de los paquetes y se importa **postgresql-client**.

Posteriormente, se descarga el archivo desde el blob de azure, para que luego se pueda restaurar la base de datos con el comando correspondiente.

ElasticSearch

Backup

Restauración

Neo4j

Backup

Restauración

CouchDB

Backup

Restauración

Conclusiones

1- Es fundamental la comunicación para un buen desarrollo del proyecto.

2- Se debe mantener una buena organización para poder realizar el trabajo.

3- Es de gran importancia entender los conceptos básicos para realizar el proyecto.

4- El tener un buen control de versiones y la correcta utilización de github facilita el trabajo en equipo.

5- Se deben aplicar buenas prácticas de programación para mantener el orden.

6- Mantener la estructura definida del proyecto es esencial para evitar el desorden.

7- Se debe desarollar un código legible y entendible.

8- Se debe organizar el equipo de trabajo desde el día 1.

9- Se debe tener una estructura clara y ordenada del proyecto y lo que requiere.

10- Es importante que cada miembro del equipo entienda la tarea a realizar.

Recomendaciones

1- Hacer reuniones periódicas para discutir los avances del proyecto y mejorar la comunicación.

2- Mantener la organización de la tarea.

3- Dividir el trabajo es importante, pero tambien es importante que cada persona del equipo entienda su tarea.

4- Hacer uso de github para el control de versiones y trabajo en conjunto.

5- Seguir un estándar de código.

6- Seguir aprendiendo y enriqueciendo el conocimiento después de finalizar el trabajo.

7- Investigar sobre las herramientas que se usan en el proyecto.

8- Tener una buena estructura del proyecto y dividir el proyecto de forma funcional para avanzar progresivamente.

9- Mantener la comunicación durante el desarrollo del proyecto y nunca quedarse con dudas.

10- Definir roles en el equipo de trabajo para mantener el orden y procurar buena dinámica de trabajo.

Referencias Bibliograficas