Apuntes 31 de marzo del 2023
Erick Madrigal

Proyecto

Para conseguir los urls de conexión a la base de datos se puede usar el comando:

```
terraform apply --var-file=conf/group.tfvars
```

si aparecen sensitivos se debe poner
terraform output <nombre de la salida sensitiva>
el usuario y el password están en el db.tf
Para conectarse a la base de datos se debe ingresar al sql server, a la sección de networking y agregar la ip de los integrantes
IAM: Identity and Access Management
Un instance role me da acceso a la base de datos, esta manera de verificación permite que, aunque alguien logre obtener este dato, no pueda ingresar a la base de datos ya que para acceder se debe pasar el valor del instance role y se debe realizar una verificación de la máquina.
Para que la aplicación de Python se pueda verificar se debe instalar un módulo de la siguiente manera:

```
pip install azure-storage-blob azure identity
```

Y hacer los siguientes imports:

```
from azure.identity import DefaultAzureCredential
```

```
from azure.storage.blob import BlobServiceClient, BlobClient, ContainerClient
```

Este codigo toma la informacion almacenada en los config files creados al realizar el comando

```
Az login
```

También puede obtener la información de identities, varibales de entorno y en parámetros

Organización de trabajo

- Base de datos en sql: todo el grupo
- Firebase y Thunkable: 1 persona
- Aplicación(node js): 1 persona
- Conexión Azure blob storage con API: 1 persona
- APIS con Thunkable: 1 persona
  Para definir la interacción entre las partes de los sistemas se utilizan contratos, donde definimos como el API y las aplicaciones o la base de datos y el API se van a comunicar.

Mongo

Standalone

1 server con colecciones completas, su crecimiento es vertical (mejorando el hardware del servidor). Un problema de este modelo es que tenemos un único punto de fallo, que se podría resolver teniendo backups y cargándolos en otro servidor, pero se debe tomar en cuenta el

tiempo que tarda en cargar esos datos.
Para evitar esos tiempos de carga se pueden tener N Replica Set así también elimino el único punto de falla. El problema de estas replicas es que si actualizo el hardware del servidor principal igual debo hacerlo en los demás servidores.

Sharded Cluster

Se particiona una colección y sus réplicas para luego repartir los shards entre los servidores llamados comodity. La estrategia con la que se reparten los shards permite tener servidores pequeños capaces de contener uno o más shard. Esta posibilidad de agregar servidores es llamado escalamiento vertical. Si no se realiza bien la partición de los datos puede que se desperdicie la potencia de los servidores.
Para separar los shards se genera un shard key con la que se indica donde se almacena el shard, con este dato un router utiliza ese key para enviar el shard a donde corresponde, esta técnica de escoger donde van los shards según su key es llamada Sharding Strategy. Esta a su vez se dividide en Hashed Sharding, Ranged Sharding y Zonal Sharding

- Hashed Sharding: se calculó un hash en base del time stamp del shardkey y según lo calculado se decide en cual shard se almacena el dato. Un mal calculo puede hacer que una gran cantidad los datos terminen en un solo shard
- Ranged Sharding: con un conocimiento de los datos se puede definir la repartición de los datos según ciertas características de los mimos
- Zonal Sharding: la información se divide según la ubicación geográfica definida.
  Para realizar el almacenamiento se puede usar mas de un Sharding Strategy a la vez

Mongo Atlas

La mayor diferencia de Mongo Atlas frente a Mongo es que Atlas es un sistema administrado, lo que significa que el usuario no se preocupa de la forma en que se almacenan los datos ni el hardware necesario ni las configuraciones, ya que todo esto está controlado por los servicios de Mongo.
Mongo permite al usuario escoger el cloud provider y las regiones que desea utilizar tanto el lugar como la cantidad. Mongo incluso le permite al usuario tener una réplica de la base de datos en cada uno de los cloud providers. Luego de eso Mongo se encarga de realizar los replica set y distribuirlos en las regiones para darle al usuario el endpoint para acceder a la base de datos.

Trabajando con Mongo

- Query API: extensiones de un lenguaje de programación con drivers
- Pipelines de Agregación: permite transformar y analizar datos
- Multi etapas: diferentes etapas de transformación de datos
- Transformación de documentos: agrupación de resultados después de transformados
- Transacciones ACID: se puede establecer un write concern, esto lo que permite establecer que tanto se desea que las transacciones de Mongo sean ACID. Se configura W (cantidad de nodos que deben confirmar que se dio una escritura, J (se utiliza para establecer si con solo enviar los datos se puede continuar o si debo esperar la confirmación de escritura en el log de la transacción) y WTIMEOUT (tiempo que se espera antes de indicar que hubo un error)

- Strong data consistency
- Change stream: hay una aplicación en repodo y cuando hay un cambio en la información se le notifica para que esta realice el trabajo necesario