CockroachDB

Fernando Anselmo
http://fernandoanselmo.orgfree.com/wordpress/

Versão 1.00 em 12 de novembro de 2023

Resumo

CockroachDB[1] é um sistema de gerenciamento de banco de dados SQL distribuído, projetado para a nuvem e construído para ser resistente a falhas. Open-source, o que significa que qualquer pessoa pode usar, copiar, modificar e distribuir o software sem pagar taxas de licença.

1 Parte inicial

CockroachDB é um sistema de gerenciamento de banco de dados distribuído (DBMS) que foi desenvolvido para fornecer alta disponibilidade, escalabilidade horizontal e consistência forte. O nome "CockroachDB"é uma referência ao inseto "barata"(cockroach em inglês), conhecido por sua durabilidade e capacidade de sobreviver a diversas condições adversas, refletindo a resiliência e a robustez que o sistema busca oferecer.

Figura 1: *Logo do CockroachDB*

CockroachDB inicia em 2012, quando seus fundadores, Spencer Kimball, Peter Mattis e Ben Darnell, estavam trabalhando na equipe de engenharia do Google. Durante esse período, estiveram envolvidos no desenvolvimento do Google's Borg, um sistema de gerenciamento de contêineres em larga escala. Inspirados por essa experiência, começaram a conceber um sistema de banco de dados distribuído que pudesse oferecer alta disponibilidade, escalabilidade e consistência forte.

A equipe fundadora deixou o Google e fundou a Cockroach Labs em 2015, com o objetivo de transformar a visão do banco de dados distribuído em uma realidade prática. O nome "CockroachDB"foi escolhido como uma metáfora para a durabilidade e resiliência desejadas no sistema, inspirada na capacidade das baratas de sobreviver a quase qualquer condição.

O projeto CockroachDB foi iniciado como um esforço de código aberto desde o início, com o código-fonte disponível no GitHub. A primeira versão alfa do foi lançada em 2016, permitindo que desenvolvedores e empresas começassem a explorar e testar o sistema.

Ao longo dos anos seguintes, a Cockroach Labs continuou a desenvolver e aprimorar o CockroachDB, lançando várias versões estáveis e adicionando recursos significativos. Um marco importante foi o lançamento da versão 1.0 em 2017, indicando a maturidade suficiente para uso em ambientes de produção.

Estas são algumas de suas características:

- **Distribuído**: projetado desde o início para funcionar como um cluster distribuído, o que significa que ele pode aproveitar o poder de muitos servidores para processar consultas e transações de forma mais eficiente.
- **Resiliente**: projetado para sobreviver a falhas de hardware e software. Ele faz isso replicando automaticamente os dados em vários nós, para que se um nó falhar, os outros possam continuar a servir os dados sem interrupção.
- **Consistente**: usa um protocolo de consenso chamado Raft para garantir que todas as réplicas de um dado estejam em um estado consistente.
- **Compatível com SQL**: suporta a maioria das características do SQL padrão, o que significa que você pode usar a mesma linguagem de consulta que você já conhece para interagir com seus dados.
- **Escalável**: pode escalar automaticamente para lidar com mais tráfego, simplesmente adicionando mais nós ao seu cluster.
- **Georreplicação**: permite o controle da localização geográfica de seus dados em um nível muito granular. Isso é útil para garantir a conformidade com as regulamentações de privacidade de dados e para otimizar a latência das consultas.

Essas características fazem do CockroachDB uma escolha interessante para aplicativos que exigem alta disponibilidade, escalabilidade e consistência em ambientes distribuídos, como sistemas financeiros, aplicativos de comércio eletrônico e outros cenários onde a confiabilidade dos dados é crítica.

1.1 Criar o contêiner Docker

A forma mais simples de termos o Cockroach é através de um contêiner no Docker, assim facilmente podemos ter várias versões do banco instalada e controlar mais facilmente qual banco está ativo ou não. E ainda colhemos o benefício adicional de não termos absolutamente nada deixando sujeira em nosso sistema operacional ou áreas de memória.

Porém nosso desejo é bem mais complexo que um simples contêiner, vamos criar não apenas uma única versão isolada, mas três contêineres e assim descobrirmos o poder da distribuição.

Uma rede Docker é uma parte essencial para permitir a comunicação entre contêineres e com o mundo exterior. Docker fornece várias opções para gerenciar redes, permite que criemos ambientes complexos e interconectados. Podemos ver as redes já existentes com o comando:

```
$ docker network ls
```

E criar uma rede Docker para abrigar nossos contêineres:

```
$ docker network create -d bridge roachnet
```

Volumes é a maneira de persistir e compartilhar dados entre contêineres. Volumes são usados para armazenar dados fora do sistema de arquivos do contêiner, isso significa que os dados podem ser

preservados mesmo que o contêiner seja removido. É uma parte fundamental da estratégia da persistência de dados no Docker. Criamos um volume para cada banco:

```
$ docker volume create roach1
$ docker volume create roach2
$ docker volume create roach3
```

Baixar a imagem oficial do CockroachDB:

```
$ docker pull cockroachdb/cockroach
```

Criar nosso primeiro contêiner:

```
$ docker run -d \
--name=roach1 \
--hostname=roach1 \
--net=roachnet \
-p 26257:26257 -p 8080:8080  \
-v "roach1:/cockroach/cockroach-data"  \
cockroachdb/cockroach start \
--insecure \
--join=roach1,roach2,roach3
```

Criar o segundo e terceiro contêineres:

```
$ docker run -d \
--name=roach2 \
--hostname=roach2 \
--net=roachnet \
-v "roach2:/cockroach/cockroach-data" \
cockroachdb/cockroach start \
--insecure \
--join=roach1,roach2,roach3
```

```
$ docker run -d \
--name=roach3 \
--hostname=roach3 \
--net=roachnet \
-v "roach3:/cockroach/cockroach-data" \
cockroachdb/cockroach start \
--insecure \
--join=roach1,roach2,roach3
```

Temos os três contêineres sendo executados roach1, roach2 e roach3. Devemos agora inicializar o Cluster:

```
$ docker exec -it roach1 ./cockroach init --insecure
```

Que deve mostrar a mensagem:

```
Cluster successfully initialized
```

Para interrompermos os contêineres, usamos:

```
docker stop roach1 roach2 roach3
```

E para ativá-los:

```
docker start roach1 roach2 roach3
```

Com os contêineres ativos podemos ver a página administrativa em `http://localhost:8080`:

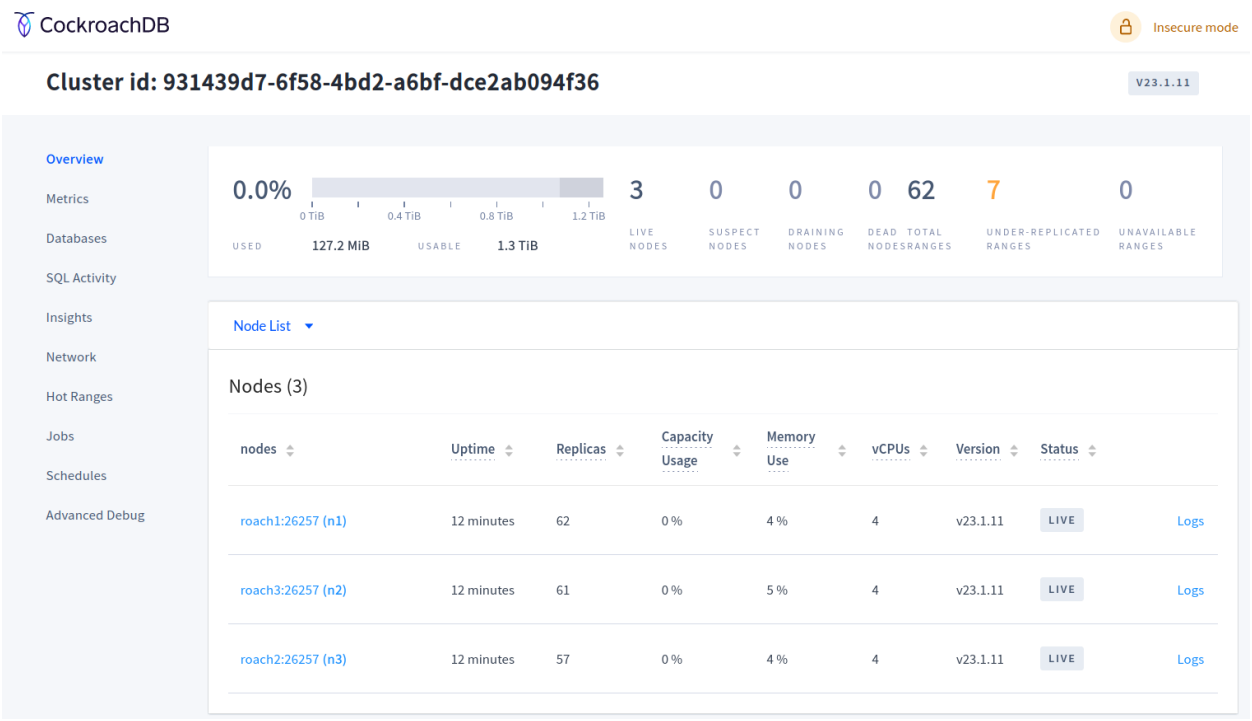

Figura 2: *Administrativo do CockroachDB*

No lado esquerdo da página temos um menu que possui várias opções interessantes, recomendo a exploração dessas e seu entendimento durante todo o nosso trabalho.

2 Criação do Database

Acessar a máquina docker criada:
```
$ docker exec -it roach1 bash
```

Acessar o SQL:
```
# ./cockroach sql --insecure
```

Uma vez conectados podemos criar nosso database:
```
> create database banco;
```

Mostrar os databases criados:
```
> show databases;
```

Utilizar um determinado database:
```
> use banco;
```

Criar uma tabela:
```
> create table conta (id serial PRIMARY KEY, balanco DECIMAL);
```

Mostrar essa tabela:
```
> show columns from conta;
```

Mostrar todas as tabelas criadas:
```
> show tables;
```

Inserir registros na tabela:

```
> insert into conta values (1, 110.10);
> insert into conta values (2, 210.15);
> insert into conta values (3, 400.00);
```

Mostrar os registros criados:

```
> select * from conta;
```

O comando:

```
> help
```

Fornece um auxílio do que é possível fazer neste ambiente. Agora vem a parte mais interessante, vamos sair desse editor de SQL:

```
> quit
```

E do contêiner:

```
# exit
```

Podemos parar quaisquer contêineres e continuarmos trabalhando livremente incluindo, alterando ou eliminando informações, ao reativarmos o contêiner parado os dados estarão sincronizados. Mas atenção, como estamos usando uma simulação, não interrompa o contêiner principal **roach1**, pois ele é master do nosso servidor.

Façamos assim, interrompa o contêiner **roach3**, entre novamente no **roach1** ou **roach2** e proceda modificações no banco ou tabela. Saia do contêiner e ative o **roach3**, agora entre neste conteiner:

```
$ docker exec -it roach3 bash
```

E verifique que os dados foram alterados corretamente.

3 CockroachDB e Apache Cassandra

São ambos sistemas de gerenciamento de banco de dados distribuídos, mas eles têm diferenças significativas em termos de arquitetura, modelo de dados, consistência de dados, e outras características. Aqui está uma comparação entre os dois:

- **Modelo de Dados**: Cassandra é um banco de dados NoSQL baseado em colunas, enquanto CockroachDB é um banco de dados relacional que suporta SQL. Isso significa que se você precisa de um modelo de dados relacional e quer usar SQL, CockroachDB pode ser uma escolha melhor. Por outro lado, se você precisa de um banco de dados baseado em colunas para lidar com grandes volumes de dados, Cassandra pode ser mais adequado.
- **Consistência de Dados**: CockroachDB oferece consistência forte, o que significa que uma vez que uma transação é confirmada, todos os futuros leitores verão o resultado dessa transação. Cassandra, por outro lado, oferece consistência eventual, o que significa que pode haver um atraso antes que todas as cópias de um dado sejam atualizadas.
- **Resiliência**: Ambos os bancos de dados são projetados para serem altamente resilientes e disponíveis. Eles fazem isso replicando dados em vários nós. No entanto, a maneira como eles lidam com falhas de nó é um pouco diferente. Cassandra usa um modelo de replicação baseado em anel, enquanto CockroachDB usa um modelo de replicação baseado em Raft.

- **Escalabilidade**: Ambos os bancos de dados são altamente escaláveis e podem lidar com grandes volumes de dados. No entanto, a maneira como eles escalam é um pouco diferente. Cassandra é conhecida por sua capacidade de escalar linearmente com a adição de mais nós, enquanto CockroachDB também oferece a capacidade de escalar geograficamente, permitindo que você distribua seus dados em diferentes regiões geográficas.
- **Suporte e Comunidade**: Cassandra, sendo um projeto Apache, tem uma grande comunidade e muitos recursos disponíveis online. CockroachDB, embora tenha uma comunidade crescente, é um projeto mais novo e pode não ter tantos recursos disponíveis.

4 Conclusão

CockroachDB é frequentemente utilizado em cenários em que alta disponibilidade, escalabilidade e consistência são requisitos críticos, como em ambientes de nuvem, microsserviços e aplicações empresariais que precisam lidar com grandes volumes de dados. Sua arquitetura distribuída e compatibilidade com SQL tornam-no uma escolha atraente para aqueles que buscam um banco de dados resiliente e fácil de integrar em seus ecossistemas existentes.

Sou um entusiasta do mundo **Open Source** e novas tecnologias. Qual a diferença entre Livre e Open Source? Livre significa que esta apostila é gratuita e pode ser compartilhada a vontade. Open Source além de livre todos os arquivos que permitem a geração desta (chamados de arquivos fontes) devem ser disponibilizados para que qualquer pessoa possa modificar ao seu prazer, gerar novas, complementar ou fazer o que quiser. Os fontes da apostila (que foi produzida com o LaTex) está disponibilizado no GitHub [4]. Veja ainda outros artigos que publico sobre tecnologia através do meu Blog Oficial [2].

Referências

[1] Página do CockroachDB
`https://www.cockroachlabs.com/`

[2] Fernando Anselmo - Blog Oficial de Tecnologia
`http://www.fernandoanselmo.blogspot.com.br/`

[3] Encontre essa e outras publicações em
`https://cetrex.academia.edu/FernandoAnselmo`

[4] Repositório para os fontes da apostila
`https://github.com/fernandoans/publicacoes`