Citus

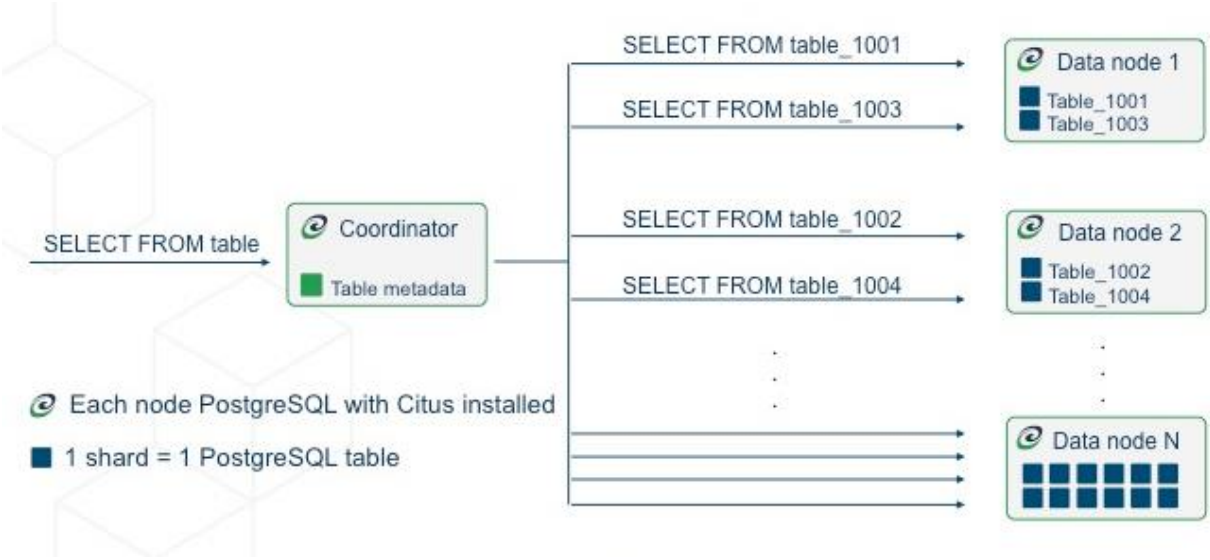
Nedir?

Citus, verileri ve sorguları bir cluster'daki birden fazla node arasında dağıtan Postgresin open source bir uzantısıdır.

İki şekilde kurulabilir;

- **Single-Node Citus:** İlk geliştirme ve testlerin yapılmasının yanı sıra bir başlangıç yapmak için kullanılır. (Kurulum linki)
- **Multi-Node Citus** (Kurulum linki)

Mimari



Kavramlar

Nodes

Node'lar ile tek bir bilgisayarda mümkün olandan daha fazla veri tutulur ve buna bağlı olarak daha fazla CPU çekirdeğini kullanır.

Bu mimari ayrıca cluster'a daha fazla node ekleyerek veri tabanını ölçeklenebilir bir hale getiriyor.

Coordinator

Her cluster'ın bir tane coordinator adında özel bir node'u vardır. Uygulamalar, sorgularını ilgili worker'lara ileten ve sonuçları biriktiren koordinatör node'una gönderir.

Coordinator, her sorgu için onu single worker node'a yönlendirir. Eğer sorguda dönülecek veriler birden fazla node arasında paylaştırılmış ise sorguyu hepsinin arasında paralelleştirerek çalıştırır. Coordinator hangi tablolarda sorguları çalıştıracağını metadata tablolarına başvurarak anlar. Bu tablolar worker node'ların DNS isimlerini, durumlarını ve verilerin düğümler arasındaki dağılımını izler.

Yani sorgular koordinatörlere gelir, koordinatörler bu sorguyu küçük parçalara böler ve farklı worker node'lara gönderir daha sonra bu sonuçlar koordinatörlerde tekrar birleştirilir.

Shards

Her node'un içinde birden fazla tablo vardır bu tablolara shard denir. Bu shard'lar aslında posgresql tablolarının daha küçük parçalara ayrılmış halidir, normal postgresql tablosu ile bir farkı yoktur.

Citus data tablo tipleri

Farklı amaçlar için kullanılan 3 farklı tablo tipi vardır;

- **Distributed Tables:** Normal tablo gibidirler, sadece tablonun verilerini birçok parçaya bölünmüş node'lar arasında tutarlar. En çok kullanılan tablo tipidir.
- **Reference Tables**
- **Local Tables**

Docker üzerinde kurulum

1. Buradan 10.0.3 versiyonunu indiriyoruz ve İlgili klasörde aşağıdaki komutu çalıştırıyoruz.
    a. `docker-compose -p citus up`
2. Bu işlem sırasında aşağıdaki hatayı alırsanız
    a. `worker_1 | error: exec: "/docker-entrypoint.sh": stat /docker-entrypoint.sh: no such file or directory`
    b. İlgili klasörde **wait-for-manager.sh** adlı dosyayı açıp son satırdaki **i.** komut yerine **ii.** Komutu yapıştırıyoruz.
        i. `exec gosu postgres "/docker-entrypoint.sh" "postgres"`
        ii. `exec gosu postgres "/usr/local/bin/docker-entrypoint.sh" "postgres"`

Bu işlemlerden sonra citusun **postgres** veri tabanına kurulu olduğu **master** ve **worker** oluşacak.

- Veri tabanına bağlanmak için aşağıdaki komutu çalıştırıyoruz.
    - `docker exec -it citus_master psql -U postgres`
- Worker'a bağlanmak için aşağıdaki komutu çalıştırıyoruz.
    - `docker exec -it {worker-adı} psql -U postgres`

Worker ekleme(Ölçeklendirme)

Eğer birden fazla worker ile çalışmak istiyorsak aşağıdaki komutu, worker sayısını girerek ilgili klasörde çalıştırıyoruz.

`docker-compose -p citus scale worker={worker-sayısı}`

Önemli komutlar

Mevcut çalışan workerları listeleme

```
SELECT master_get_active_worker_nodes();
```

Shards sayısı sınırlama

```
set citus.shard_count={shard-sayısı};
```
(Tabloyu distributed yapmadan önce kullanılmalı)

Tabloyu distrubute yapmak

```
SELECT create_distributed_table('tablo-adı','primary-key-kolon-adı');
```

Node'ları listeleme

```
SELECT * from pg_dist_node;
```

Shard'ları listeleme

```
SELECT * FROM citus_shards;
```