Citus

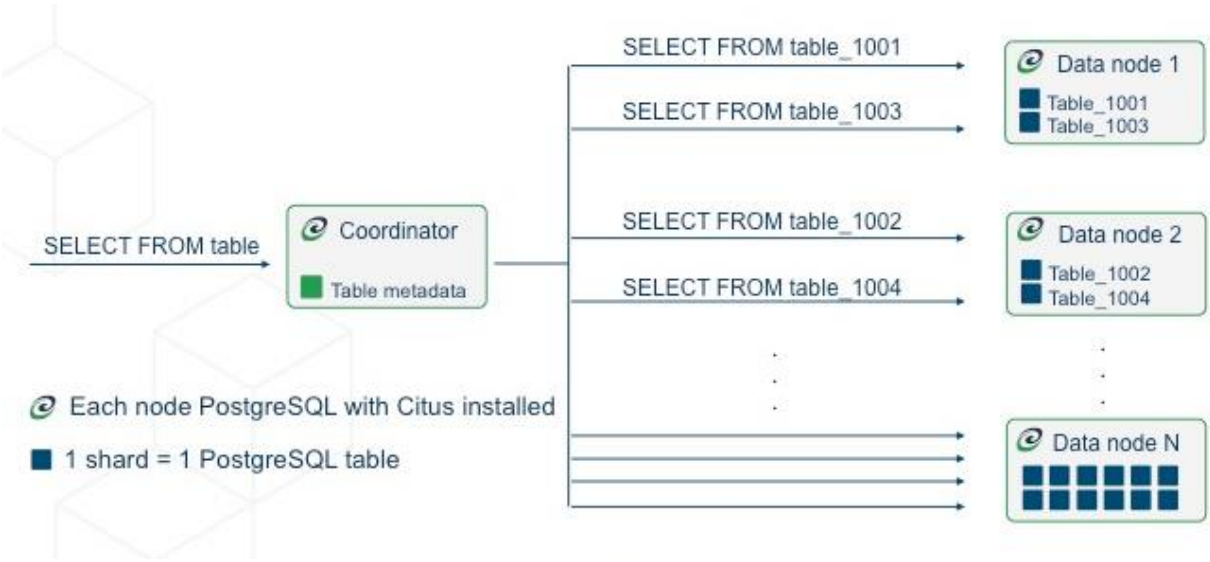
What?

Citus is an open source extension of Postgres that distributes data and queries among multiple nodes in a cluster.

It can be installed in two ways;

- **Single-Node Citus:**It is used for initial development and testing as well as to get a head start. (Installation link)
- **Multi-Node Citus**(Installation link)

Architectural



Concepts

nodes

Nodes hold more data than is possible on a single computer, and accordingly use more CPU cores.

This architecture also adds more nodes to the cluster, making the database scalable.

coordinator

Each cluster has a special node called a coordinator. Applications send their queries to the coordinator node, which forwards their queries to the respective workers and accumulates the results.

The coordinator forwards it to the single worker node for each query. If the data to be returned in the query is shared between more than one node, it runs the query by parallelizing all of them. The coordinator understands which tables to run queries on by referring to the metadata tables. These tables track the DNS names of the worker nodes, their state, and the distribution of data among the nodes.

In other words, the queries come to the coordinators, the coordinators divide this query into small parts and send it to different worker nodes, then these results are reassembled in the coordinators.

Shards

Each node has more than one table in it, these tables are called shards. These shards are actually smaller pieces of posgresql tables, no different than a regular postgresql table.

Citus data table types

There are 3 different types of tables used for different purposes;

- **Distributed Tables:**They are like a regular table, they just keep the table's data between nodes divided into many parts. It is the most used table type.

- **Reference Tables**
- **Local Tables**

Installation on Docker

one.From here We download version 10.0.3 and run the following command in the relevant folder.

a. `docker-compose -p citus up`

2. If you get the following error during this process

a. `worker_1 | error: exec: "/docker-entrypoint.sh": stat /docker-entrypoint.sh: no such file or directory`

b. in the relevant folder**wait-for-manager.sh**open the file named**I.** instead of command**ii.**Paste the command.

I.**exec**gosu postgres"/docker-entrypoint.sh" "postgres"

ii.**exec**gosu postgres"/usr/local/bin/docker-entrypoint.sh" "postgres"

After these procedures, your citus**postgres**installed in the database**master**and **worker**will occur.

- We run the following command to connect to the database.
  he is `docker exec -it citus_master psql -U postgres`
- We run the following command to connect to Worker.
  he is `docker exec -it {worker-name} psql -U postgres`

Adding Workers (Scaling)

If we want to work with more than one worker, we run the following command in the relevant folder by entering the number of workers.

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

Important commands

List current employee workers

```
SELECT master_get_active_worker_nodes();
```

Limiting the number of shards

```
set citus.shard_count={shard-count};
```
(must be used before making the table distributed)

Distribute the table

```
SELECT create_distributed_table('table-name','primary-key-col-name');
```

List nodes

```
SELECT * from pg_dist_node;
```

List shards

```
SELECT * FROM citus_shards;
```