

Сравнение производительности кластера Citus на базе Paton и ArendataDB на большом объеме данных

oTus

PostgreSQL. Advanced



**Меня хорошо видно
& слышно?**



Защита проекта

Тема: Сравнение производительности кластера Citus на базе Patoni и ArenadataDB на большом объеме данных



Воронов Алексей

DevOps инженер ООО «РТК ИТ»



План защиты

Цель и задачи проекта

Какие технологии использовались

Что получилось

Выводы

Вопросы и рекомендации



Цель и задачи проекта

Цель проекта: сравнить производительность аналитических запросов кластеров "**Citus на базе Patoni**" и "**ArenadataDB**"

1. Установить кластер Citus и кластер ArenadataDB
2. Установить базовый мониторинг кластеров
3. Загрузить датасет для тестирования OLAP запросов
4. Выполнить тестовые OLAP запросы и сравнить результаты тестов
5. Сделать результирующие выводы и ответить для себя на вопрос:
В чем разница между Citus и Greenplum ?

Какие технологии использовались

- | | |
|----|---|
| 1. | Citus + Patroni |
| 2. | ADCM + ArenadataDB |
| 3. | Prometheus + etc.exporters + Grafana |
| 4. | extension PostGIS |
| 5. | Ansible для ad-hook операций + Vmware в качестве платформы виртуализации использовалась |



Конфигурация нод:

Версия OS: Ubuntu 22.04
CPU/8, Mem/16GB, SSD/30GB

Распределение нод:

3 для мастер нод
6 для воркер нод

Ограничения:

используется виртуализация vmware
под капотом один SSD диск на все VM

Датасет для тестирования:

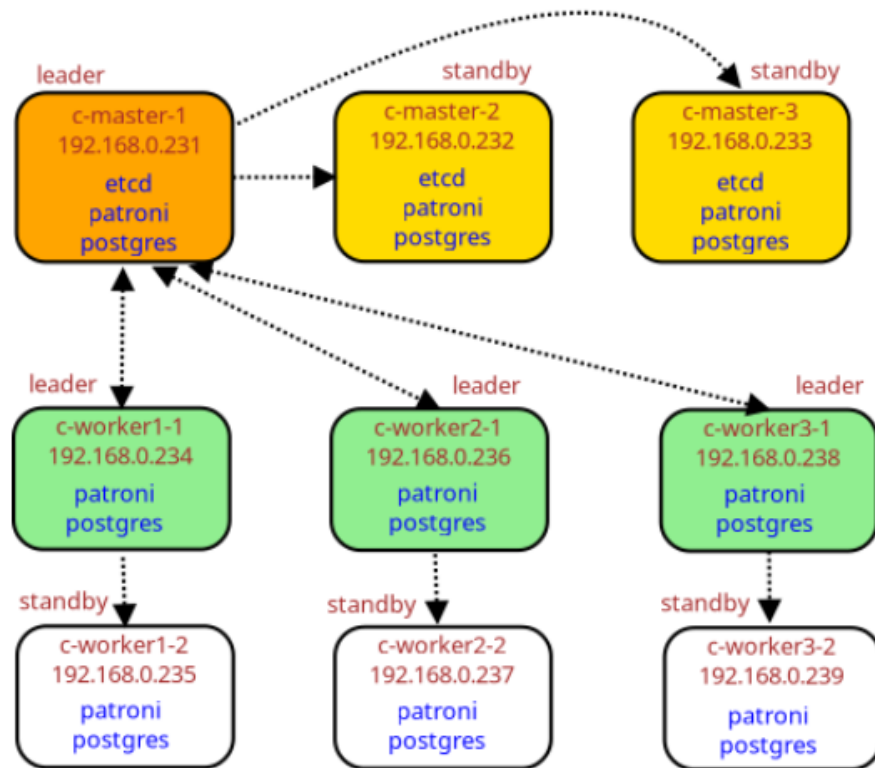
<https://zenodo.org/records/7923702>

(открытые данные авиаперелетов с 2019 по 2022 год)



Схема стенда "кластера Citus на базе Patroni"

Схема тестирования кластера Citus+Patroni

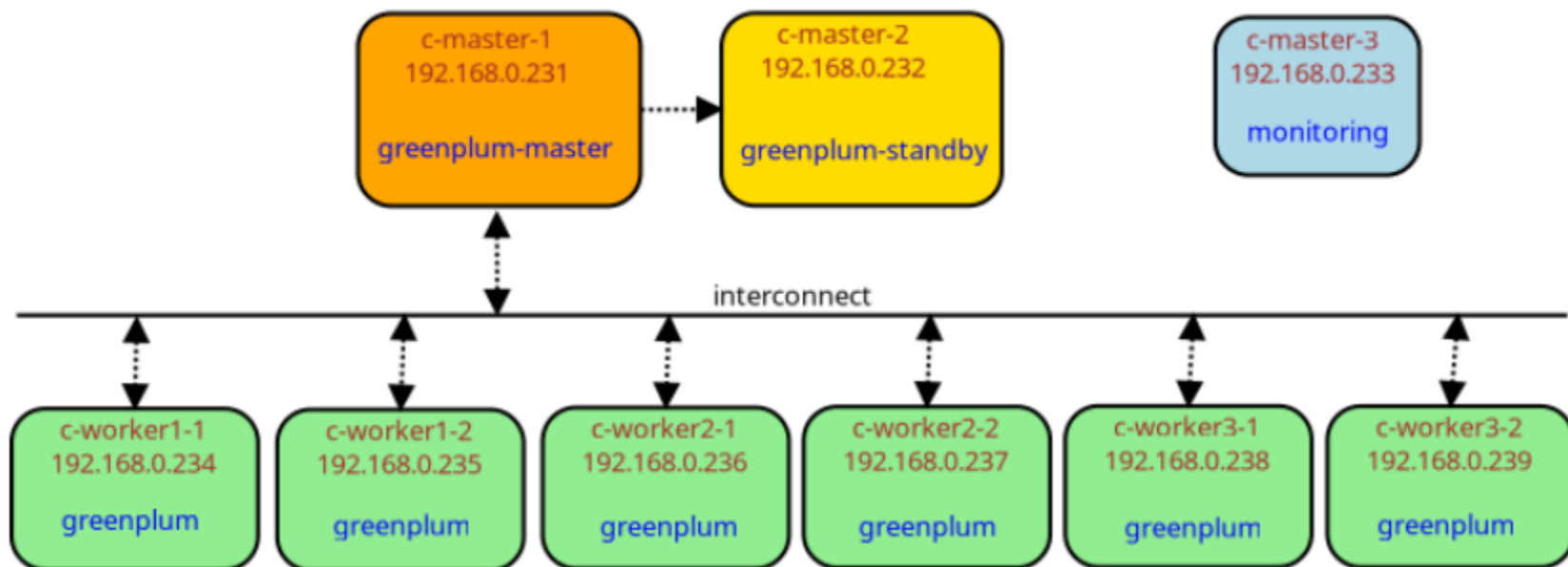


Статус кластера Citus после установки


```
root@c-worker1-1:~# patronictl --config-file=/etc/patroni/config.yml list
+ Citus cluster: patroni +-----+-----+-----+-----+
| Group | Member      | Host      | Role          | State      | TL | Lag in MB |
+-----+-----+-----+-----+-----+-----+-----+
| 0     | c-master1   | c-master1 | Leader        | running    | 1  |          |
| 0     | c-master2   | c-master2 | Quorum Standby | streaming  | 1  | 0         |
| 0     | c-master3   | c-master3 | Quorum Standby | streaming  | 1  | 0         |
| 1     | c-worker1-1 | c-worker1-1 | Leader        | running    | 1  |          |
| 1     | c-worker1-2 | c-worker1-2 | Quorum Standby | streaming  | 1  | 0         |
| 2     | c-worker2-1 | c-worker2-1 | Leader        | running    | 1  |          |
| 2     | c-worker2-2 | c-worker2-2 | Quorum Standby | streaming  | 1  | 0         |
| 3     | c-worker3-1 | c-worker3-1 | Leader        | running    | 1  |          |
| 3     | c-worker3-2 | c-worker3-2 | Quorum Standby | streaming  | 1  | 0         |
+-----+-----+-----+-----+-----+-----+-----+
```

Схема стенда кластера ArenadataDB

Схема тестирования кластера ArenadataDB



Статус кластера ADB:



Clusters

Hostproviders

Hosts

Jobs

Access manager

Audit

Bundles

admin

Settings

Log out

Clusters / otus / Overview

otus • Overview Services Hosts Mapping Configuration Import

Info ▾

The main goal of ADB bundle is an easy and fast installation and managing of Arenadata Database with Arenadata Cluster Manager. ADB bundle consists of ADB, time synchronization and the monitoring client services. These services can be installed either on cloud or bare metal hosts (depending on host bundles you choose). After installation Arenadatabase cluster will be available for psql connect on port 5432 of the master host address.

Services ▾

2/0

All Up Down

ADB •
3/3 components

Monitoring Clients •
1/1 components

Show 10 per page > 1 < >

Hosts ▾

8/0


All Up Down

master1 • master2 •

worker1-1 • worker1-2 •

worker2-1 • worker2-2 •

worker3-1 • worker3-2 •

ARENADATA
CLUSTER MANAGER

Clusters

Hostproviders

Hosts

Jobs

Access manager

Audit

Bundles

admin

Settings

Log out

Clusters

adb-monitoring / Overview

adb-monitoring • Overview Services Hosts Mapping Configuration Import

Info

The main goal of Monitoring bundle is an easy and fast installation and managing of Arenadata Monitoring with Arenadata Cluster Manager. Monitoring bundle consists of [Graphite](#), [Grafana](#) and [Diamond](#). These services can be installed either on cloud or bare metal hosts (depending on host bundles you choose). After installation Grafana web-interface would be available on port 3000 (configurable) of the Grafana host address.

Services

All Up Down

Diamond •
1/1 components

Grafana •
1/1 components

Graphite •
1/1 components

Show 10 per page >

1 < >

Hosts

All Up Down

master3 •

OLAP запросы использованные для сравнительного тестирования

1) Общее кол-во полетов:

```
SELECT COUNT(*) FROM opensky;
```

2) Кол-во полетов "callsign IN ('UUEE', 'UUDD', 'UUWW')":

```
SELECT COUNT(*) FROM opensky WHERE callsign IN ('UUEE', 'UUDD', 'UUWW');
```

3) ТОП 10 аэропортов с максимальным кол-вом полетов:

```
SELECT origin, COUNT(*) FROM opensky WHERE origin != " GROUP BY origin ORDER BY count(*) DESC limit 10;
```

4) ТОП 10 аэропортов с максимальным кол-вом полетов и с расчетом суммарного полетного расстояния:

```
SELECT origin, count(*), round(avg(ST_Distance(ST_MakePoint(longitude_1, latitude_1)::geography,  
ST_MakePoint(longitude_2, latitude_2)::geography))) AS distance FROM opensky WHERE origin != " GROUP BY  
origin ORDER BY count(*) DESC LIMIT 10;
```

5) ТОП 10 аэропортов с максимальным кол-вом полетов и с расчетом суммарного полетного расстояния за 2019-09-01:

```
SELECT origin, count(*), round(avg(ST_Distance(ST_MakePoint(longitude_1, latitude_1)::geography,  
ST_MakePoint(longitude_2, latitude_2)::geography))) AS distance FROM opensky WHERE firstseen >= '2019-09-01'  
AND firstseen < '2019-09-02' and origin != " GROUP BY origin ORDER BY count(*) DESC LIMIT 10;
```








6) Суммарное расстояние за все время наблюдения:

```
SELECT sum(ST_Distance(ST_MakePoint(longitude_1, latitude_1)::geography, ST_MakePoint(longitude_2,  
latitude_2)::geography))/1000 AS distance FROM opensky ;
```



Сравнение результатов тестирования

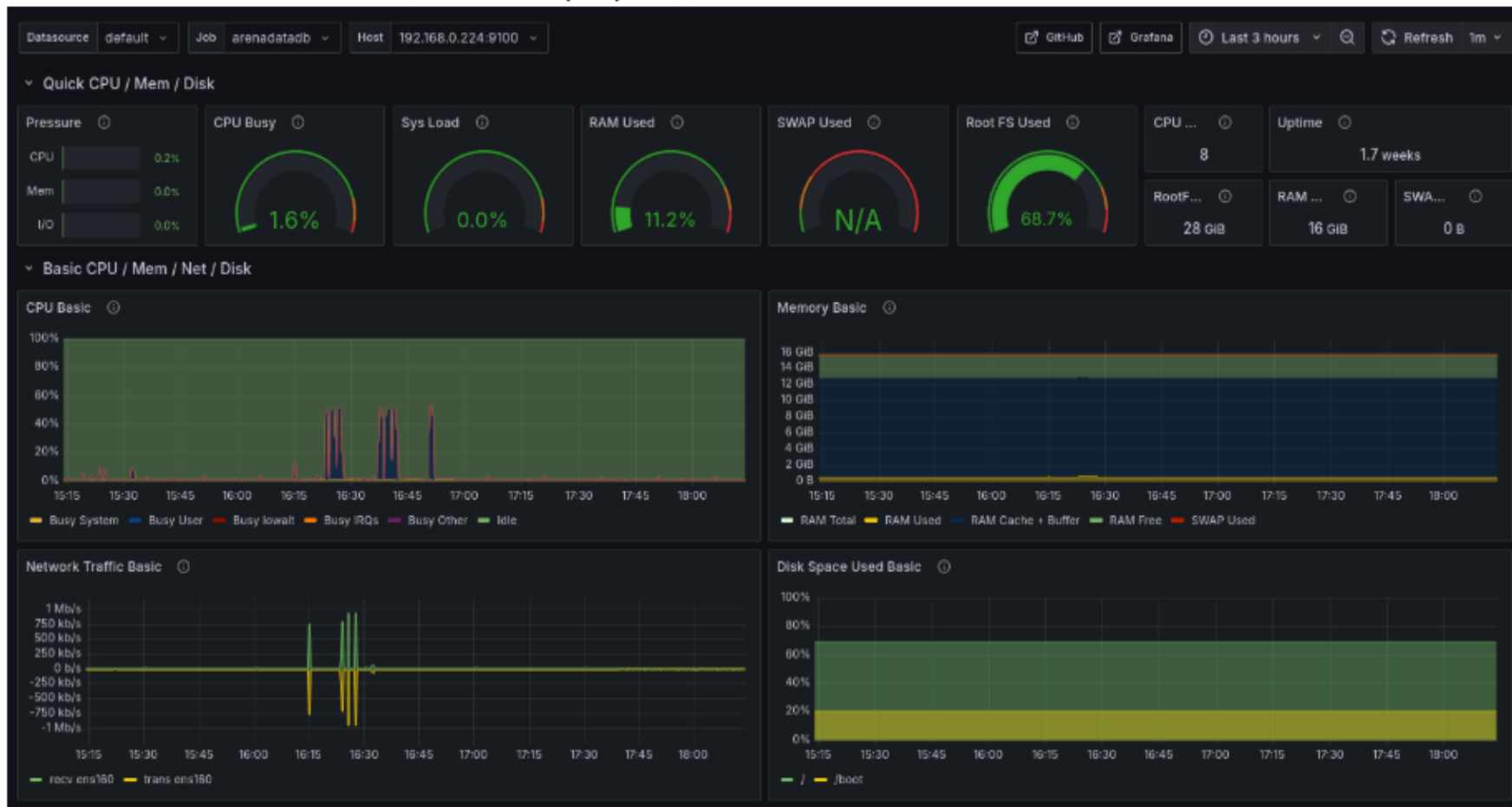
Результаты тестирования сведены в таблицу:

Аналитические запросы	Citus	ArenadataDB
1. Общее кол-во полетов	1925 ms	1919 ms 
2. Кол-во полетов "callsign IN ('UUEE', 'UUDD', 'UUWW')"	899 ms 	1382 ms
3. ТОП 10 аэропортов с максимальным кол-вом полетов	4378 ms	4103 ms 
4. ТОП 10 аэропортов с макс/кол-вом полетов и с расчетом суммарного полетного расстояния	110239 ms	78939 ms 
5. ТОП 10 аэропортов с макс/кол-вом полетов и с расчетом суммарного полетного расстояния за 1 мес.	875 ms	513 ms 
6. Суммарное расстояние за все время наблюдения	110386 ms	98788 ms 
7. Скорость загрузки датасета	30 минут	20 минут 

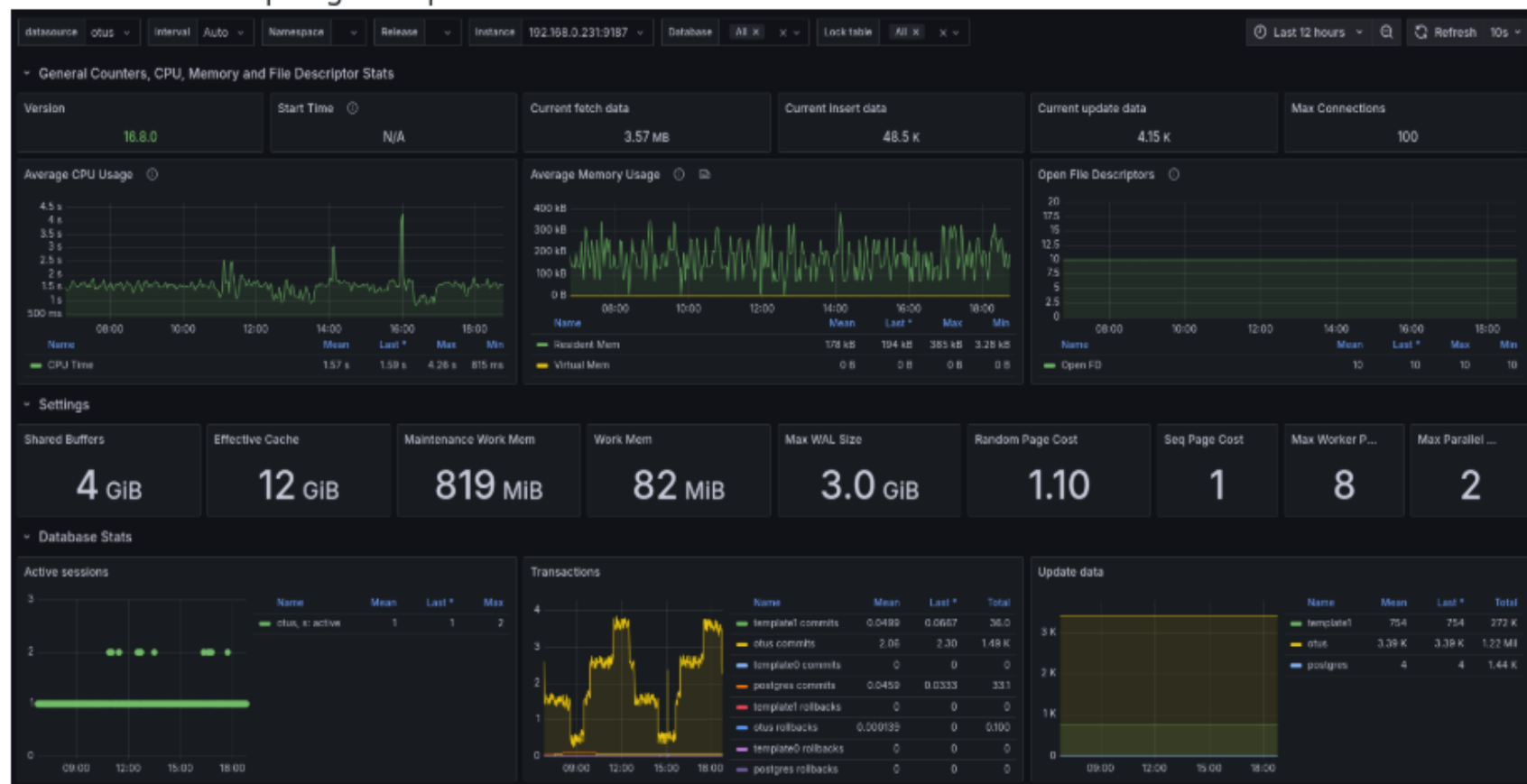
Citus - статистика с одной из воркер нод



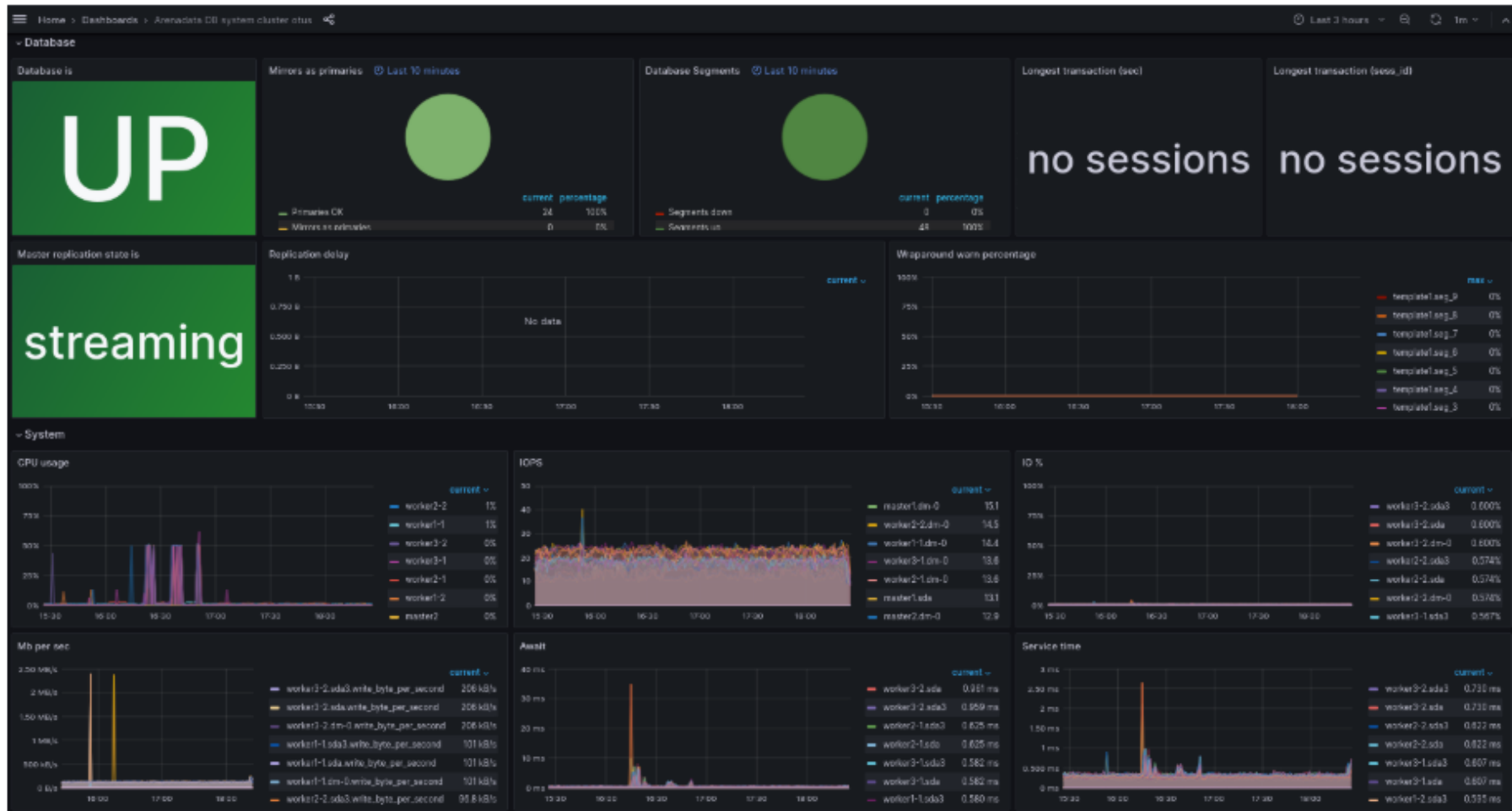
ArenadataDB - статистика с одной из воркер нод



Citus - статистика postgres-exporter



ArenadataDB Monitoring



Выводы (по результатам тестирования):

ArenadataDB (Greenplum) представляет собой массив отдельных баз данных PostgreSQL, работающих вместе для представления единого образа базы данных.

Citus преобразует отдельные экземпляры PostgreSQL в распределенный кластер, который будет размещать части табличных данных (шарды), где данные распределены по ключу шардирования.

Оба решения разработаны с учетом **горизонтального масштабирования**, позволяя распределять данные и запросы по множеству узлов

Скорость выполнения простых OLAP запросов примерно одинакова у обоих решений, но сложные OLAP запросы **ArenadataDB** выполняет значительно быстрее

Загрузка данных в ArenadataDB происходит значительно быстрее

При тестовых OLAP запросах **ArenadataDB** значительно меньше нагружает CPU на воркер нодах

Оба решения позволяют использовать **колоночный тип хранилища с сжатием данных**.

Оба решения имеют удобные **инструменты для контроля состояния**



Выводы (по внутреннему устройству):

Citus для управления распределенными транзакциями использует **протокол двухфазной фиксации (2PC)**, он необходим чтобы гарантировать, что транзакция будет зафиксирована либо на всех узлах, либо не зафиксирована нигде.

Тем не менее, в **Citus** ранее подготовленные транзакции на отдельных узлах применяются в разное время.

Поэтому последующая транзакция сможет прочитать данные, которые были применены на одном узле, но еще не были применены на другом.

Фактически, это нарушает принцип атомарности, позволяя увидеть часть данных о зафиксированной транзакции.

В **ArenadataDB(Greenplum)** это невозможно благодаря работе со снапшотами.

Ядро Greenplum отслеживает информацию об активных распределенных транзакциях, храня идентификатор распределенной транзакции в общей памяти процесса каждого бэкенда, аналогично стандартным локальным идентификаторам транзакций PostgreSQL.

Координатор запросов отвечает за генерацию и назначение значения идентификатор распределенной транзакции.



Выводы:

Таким образом, хотя установка **Citus** расширяет возможности PostgreSQL, она не делает его эквивалентом **Greenplum**, поскольку оба решения имеют разные архитектурные подходы и оптимизированы для различных типов рабочих нагрузок.

Citus может повысить производительность PostgreSQL в **распределенных OLTP-нагрузках**.

Greenplum имеет **специфические функции аналитической обработки больших данных**, включая **поддержку сложных аналитических запросов**, а также **интеграцию с различными инструментами анализа и ETL/ELT-процессов**.



Вопросы и рекомендации

+ если есть вопросы

— если вопросов нет

**Спасибо за
внимание!**

