МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное образовательное учреждение высшего образования “Национальный исследовательский университет ИТМО”

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине

“Распределённые системы хранения данных”

вариант 234562

Выполнили:

Иванов Матвей Сергеевич

Шульга Артём Игоревич

группа P33111

Преподаватель:

Николаев Владимир Вячеславович

г. Санкт-Петербург, 2023

Задание

На выделенном узле создать и сконфигурировать новый кластер БД, саму БД, табличные пространства и новую роль в соответствии с заданием.
Произвести наполнение базы.

Отчёт должен содержать все команды по настройке, а также измененные строки конфигурационных файлов.

Подключение к узлу через helios:

1) подключаетесь к гелиосу
2) ssh пользователь@узел

Персональный пароль для работы с узлом выдается преподавателем.
Обратите внимание, что домашняя директория пользователя /var/postgres/$LOGNAME

Инициализация кластера БД

- Имя узла — pg165.
- Имя пользователя — postgres0.
- Директория кластера БД — $HOME/u22/pg1651.
- Кодировка, локаль — UTF8, английская
- Перечисленные параметры задать через аргументы команды.

Конфигурация и запуск сервера БД

- Способ подключения к БД — TCP/IP socket, номер порта 9165.
- Остальные способы подключений запретить.
- Способ аутентификации клиентов — по имени пользователя.
- Настроить следующие параметры сервера БД:
  max_connections,
  shared_buffers,
  temp_buffers,
  work_mem,
  checkpoint_timeout,
  effective_cache_size,
  fsync,
  commit_delay.
  Параметры должны быть подобраны в соответствии со сценарием OLAP: 10 пользователей, пакетная запись данных в среднем по 80 МБ.
- Директория WAL файлов — поддиректория в PGDATA.
- Формат лог-файлов — csv.
- Уровень сообщений лога — INFO.
- Дополнительно логировать — контрольные точки.

Дополнительные табличные пространства и наполнение

- Пересоздать шаблон template0 в новом табличном пространстве:
  - $HOME/u02/zne78.
- На основе template0 создать новую базу — theovermind8.
- От имени новой роли (не администратора) произвести наполнение существующих баз тестовыми наборами данных. Предоставить права по необходимости. Табличные пространства должны использоваться по назначению.
- Вывести список всех табличных пространств кластера и содержащиеся в них объекты.

Ход выполнения

Инициализация кластера БД

Подключение к узлу:
ssh postgres0@pg165

Инициализация кластера БД:
initdb --encoding UTF8 --locale en_US.UTF-8 -U postgres0 -D $HOME/u22/pg1651

Конфигурация и запуск сервера БД

Путь до pg_hba.conf:
psql: SHOW hba_file;

Путь до postgresql.conf:
psql: SHOW config_files;

pg_hba.conf:
TYPE DATABASE USER ADDRESS METHOD
host all postgres0 all md5
Все остальные - метод в reject
Для локального подключения:
local all all peer

Перезагружаем конфигурацию:

SELECT pg_reload_conf();

postgresql.conf:

port = 9165
max_connections=10
# Объём кэшированных данных. Поскольку у нас система OLAP, размер кэша должен быть велик
shared_buffers=1GB # 80 MiB * 10 users + extra;
# Объём для временных объектов (временные таблицы например). Для OLAP нет необходимости в большом количестве
temp_buffers=80MB # 8 MiB * 10 users

# Объём памяти для каждой операции хэширования и сортировки. Для OLAP необходимо много памяти под эти операции
work_mem=80 MB
# Таймаут для автоматической контрольной точки. У OLAP длительные операции, поэтому можно увеличить таймаут
checkpoint_timeout=1h
# Ожидаемый объем кэшированных данных. Для OLAP необходимы большие кэши
effective_cache_size=1 GB
# Отключаем запись с использованием fsync для повышения скорости
fsync=off
# Задержка перед фактическим сохранением на диск. В OLAP мало транзакций, поэтому можно увеличить это значение
commit_delay=100000 (100 ms)

log_destination=csvlog
logging_collector=on # для csv логов
log_min_messages=info
log_checkpoints=on

Директория WAL файлов — поддиректория в PGDATA. - нельзя поменять с помощью конфигов, только останавливать постгрю, перемещать руками + делать ссылки

Запуск сервера БД

pg_ctl -D $HOME/u22/pg1651 start

Дополнительные табличные пространства и наполнение

Пересоздание template0:

CREATE TABLESPACE new_tablespace0 LOCATION '/var/db/postgres0/u02/zne78';
CREATE DATABASE new_template0 TEMPLATE template0 TABLESPACE new_tablespace0;
UPDATE pg_database SET datistemplate=true WHERE datname='new_template0';

Создаём базу theovermind8:

CREATE DATABASE theovermind8
WITH TEMPLATE = new_template0 TABLESPACE = new_tablespace0;

Наполнение баз данных:

```
// Создание новой роли
CREATE ROLE sample WITH LOGIN PASSWORD ‘password’;
GRANT CONNECT ON DATABASE postgres, postgres0, theovermind8 TO sample;
GRANT CREATE ON SCHEMA public TO sample;
GRANT SELECT, INSERT ON ALL TABLES IN SCHEMA public TO sample;

// В postgres
CREATE TABLE test_table(id integer primary key, name varchar(255), age integer);

// В postgres0
CREATE TABLE inf (id int primary key, another_id int, some_other_id int);
CREATE TABLE dobro (id int primary key, dobro_amount int, dobro_reciever_id int);


// В theovermind8
CREATE TABLE secure_paymnet_information(
        id int primary key, payment_id int, amount int, reciever_id int
);
```

Получение всех объектах во всех табличных пространствах в текущей БД:

Притом мы не можем получить объекты из других баз данных, поскольку pg_class содержит данные только по текущей базе данных.

```
WITH deafult_tsp as (
        SELECT tsp.spcname as default_tsp
        FROM pg_tablespace tsp
        JOIN pg_database db ON tsp.oid = db.dattablespace
        WHERE db.datname = current_database()
)
SELECT
        COALESCE(tsp.spcname, d.default_tsp) as tablespace,
        c.relname AS object_name,
        CASE c.relkind
            WHEN 'r' THEN 'table'
            WHEN 'i' THEN 'index'
            WHEN 'S' THEN 'sequence'
            WHEN 'v' THEN 'view'
            WHEN 'm' THEN 'materialized view'
            WHEN 'c' THEN 'composite type'
            WHEN 't' THEN 'TOAST table'
            WHEN 'f' THEN 'foreign table'
        END AS object_type
FROM pg_catalog.pg_class c
JOIN pg_catalog.pg_namespace n ON n.oid = c.relnamespace
CROSS JOIN deafult_tsp d
LEFT JOIN pg_catalog.pg_tablespace tsp ON  tsp.oid = c.reltablespace
WHERE n.nspname NOT LIKE 'pg_%'  AND n.nspname != 'information_schema';
```

Вывод в БД postgres на выделенной ноде:

```
 tablespace |   object_name    | object_type
------------+------------------+-------------
 pg_default | test_table       | table
 pg_default | test_table_pkey  | index
(2 строки)
```

Для того, чтобы получить списки объектов во всех табличных пространствах во всех базах данных можно написать BASH скрипт, который будет поочередно подключаться ко всем БД и исполнять скрипт на тот, что привердён выше. Либо можно написать функцию на языке plpgsql с использованием расширения dblink (для подключения к БДшкам);

Рассмотрим каждый из этих способов подробнее:

1) Исполняемый BASH скрипт

SQL файл для получения всех баз данных в табличных пространствах (dbs.sql):

```
SELECT
        tsp.spcname as tablespace,
        db.datname as database_name
FROM pg_tablespace tsp
LEFT JOIN pg_database db ON tsp.oid = db.dattablespace
WHERE db.datistemplate IS DISTINCT FROM true;
```

SQL файл для получения всех объектов в базе данных (в дефолтном для БД табличном пространстве)(obj_default.sql):

```
SELECT
        n.nspname AS schema_name,
        c.relname AS object_name,
        CASE c.relkind
            WHEN 'r' THEN 'table'
            WHEN 'i' THEN 'index'
            WHEN 'S' THEN 'sequence'
            WHEN 'v' THEN 'view'
            WHEN 'm' THEN 'materialized view'
            WHEN 'c' THEN 'composite type'
            WHEN 't' THEN 'TOAST table'
            WHEN 'f' THEN 'foreign table'
        END AS object_type
FROM pg_catalog.pg_class c
JOIN pg_catalog.pg_namespace n ON n.oid = c.relnamespace
```

```
WHERE c.reltablespace <> 0 AND n.nspname NOT LIKE 'pg_%'  AND n.nspname !=
'information_schema';
```

SQL файл для получения всех объектов в базе данных (в заданных табличных пространствах)(obj_tsp.sql):

```
SELECT
        tsp.spcname AS tablespace,
        current_database() AS db_name,
        n.nspname AS schema_name,
        c.relname AS object_name,
        CASE c.relkind
            WHEN 'r' THEN 'table'
            WHEN 'i' THEN 'index'
            WHEN 'S' THEN 'sequence'
            WHEN 'v' THEN 'view'
            WHEN 'm' THEN 'materialized view'
            WHEN 'c' THEN 'composite type'
            WHEN 't' THEN 'TOAST table'
            WHEN 'f' THEN 'foreign table'
        END AS object_type
FROM pg_catalog.pg_class c
JOIN pg_catalog.pg_namespace n ON n.oid = c.relnamespace
JOIN pg_catalog.pg_tablespace tsp ON tsp.oid = c.reltablespace
WHERE n.nspname NOT LIKE 'pg_%'  AND n.nspname != 'information_schema';
```

BASH скрипт (script.sh):

```
#!/bin/bash
if [ $# -ne 2 ]; then
  echo "Usage: $0 <login> <password>"
  exit 1
fi
export PGPASSWORD=$2
spaces_and_dbs=$(psql -U "$1" -h 127.0.0.1 -p 9165 -d postgres -f dbs.sql -t)
while IFS=' | ' read -r space_name db_name; do
  if [ -z "$db_name" ]
  then
```

```
    echo "$space_name | no objects found"
    continue
  else
    echo "$space_name | $db_name"
  fi
  obj_output=$(psql -U "$1" -h 127.0.0.1 -p 9165 -d $db_name -f obj.sql -t)
  while IFS=" | " read -r schema_name object_name object_type; do
     echo "   $space_name | $db_name | $schema_name | $object_name |
$object_type"
  done <<< "$obj_output"
done <<< "$spaces_and_dbs"
```

Получение всех объектов во всех табличных пространствах:
bash script.sh <superuser username> <password>

Вывод команды на выделенной ноде:

```
[postgres0@pg165 ~]$ bash script.sh postgres0 123456
pg_default | postgres
    pg_default | postgres | public | test_table | table
    pg_default | postgres | public | test_table_pkey | index
pg_default | postgres0
    pg_default | postgres0 | public | inf | table
    pg_default | postgres0 | public | inf_pkey | index
    pg_default | postgres0 | public | dobro | table
    pg_default | postgres0 | public | dobro_pkey | index
pg_global | no objects found
new_tablespace0 | theovermind8
    new_tablespace0 | theovermind8 | public | secure_paymnet_information2 | table
    new_tablespace0 | theovermind8 | public | secure_paymnet_information2_pkey | index
```

2) Функция с использованием dblink:

Также можно, создав функцию с использованием dblink:

```
CREATE EXTENSION dblink;
CREATE OR replace FUNCTION get_tablespace_objects (username TEXT, password
TEXT) returns SETOF RECORD
AS
  $$
  DECLARE
    connection_info TEXT;
    i RECORD;
    r RECORD;
  BEGIN
```

```
    FOR r IN (
        SELECT
                cast(db.datname AS TEXT) AS db_name,
                cast(tsp.spcname AS TEXT) as tablespace
        FROM  pg_database  db
        JOIN pg_tablespace tsp ON tsp.oid = db.dattablespace
        WHERE  db.datistemplate = FALSE
     )  LOOP

    SELECT format ('dbname=%s user=%s password=%s', r.db_name, username,
password)
        INTO connection_info;

    perform dblink_connect(connection_info);
    FOR i IN (
        SELECT
                r.tablespace,
                r.db_name,
                t.nspname AS schema_name,
                t.relname AS object_name,
                CASE t.relkind
                    WHEN 'r' THEN 'table'
                    WHEN 'i' THEN 'index'
                    WHEN 'S' THEN 'sequence'
                    WHEN 'v' THEN 'view'
                    WHEN 'm' THEN 'materialized view'
                    WHEN 'c' THEN 'composite type'
                    WHEN 't' THEN 'TOAST table'
                    WHEN 'f' THEN 'foreign table'
                END AS object_type
        FROM  dblink(
          'SELECT n.nspname::TEXT, c.relname::TEXT, c.relkind::TEXT FROM
        pg_catalog.pg_class c JOIN pg_catalog.pg_namespace n  ON n.oid =
        c.relnamespace;'
        ) AS t(nspname TEXT, relname TEXT, relkind TEXT)
        WHERE t.nspname NOT LIKE 'pg_%'  AND t.nspname != 'information_schema'
    ) LOOP
        RETURN NEXT i;
```

```
    END LOOP;
      perform dblink_disconnect();
END LOOP;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

SELECT * FROM get_tablespace_objects(<superuser username>, <password>) AS (tablespace TEXT, db_name TEXT, schema_name TEXT, object_name TEXT, object_type TEXT);

Но, к несчастью, на выделенной ноде нельзя поставить EXTENSION dblink;

Доп задание №1:

Создать таблицу с текстовым полем, добавить длинную строку. Посмотреть как хранится

После добавления длинной рандомной строки, можем посмотреть название TOAST таблицы:

```
SELECT relname FROM pg_class WHERE oid = (
        SELECT reltoastrelid FROM pg_class WHERE relname = 'long_text_field'
);
```

И Содержание этой TOAST таблицы (не выводя сами блоки данных):

```
SELECT chunk_id, chunk_seq FROM pg_toast.pg_toast_16434;
```

Выведем размеры таблиц:

1) total, включая размер самой таблицы, индексы и TOAST
2) table_size, включая размер самой таблицы и TOAST
3) А также отдельно размер TOAST

```
select
  table_name,
  pg_size_pretty(pg_total_relation_size(quote_ident(table_name))) as
total_relation_size,
  pg_total_relation_size(quote_ident(table_name)) as total_relation_size_byte,
  pg_size_pretty(pg_table_size(quote_ident(table_name))) as table_size,
  pg_table_size(quote_ident(table_name)) as table_size_bytes,
  pg_size_pretty(pg_table_size(quote_ident(table_name)) -
pg_relation_size(quote_ident(table_name))) as toast_size,
  (pg_table_size(quote_ident(table_name)) -
pg_relation_size(quote_ident(table_name))) as toast_size_bytes
from information_schema.tables
where table_schema = 'public'
order by 3 desc;
```

Доп задание №2:

Информация обо всех работающих процессах:

```
SELECT * from pg_stat_activity ;
```

Статистика по всем таблицам в базе данных:

```
SELECT * FROM pg_stat_user_tables;
```

Логирование:

```
SHOW logging_collector;
SELECT pg_current_logfile();
```

WAL файлы:

```
SELECT pg_current_wal_lsn();
SELECT * FROM pg_stat_wal \gx
```

Background writer:

```
SELECT * FROM pg_stat_bgwriter \gx
```

Информация по logical replication

```
SELECT * FROM pg_stat_replication \gx
```

По AutoVacuum:

```
SELECT * FROM pg_stat_progress_vacuum;
select * from pg_settings where name like '%autovacuum%'
```

Вывод

В ходе выполнения данной лабораторной работы мы научились создавать и настраивать кластер Postgres по заданным требованиям, а также организовывать доступ к нему. Помимо этого мы научились работать с шаблонами баз данных и с табличными пространствами.