Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Администрирование СУБД

Лабораторная работа №2

Вариант №456555

Группа: P3324

Выполнил: Маликов Глеб Игоревич

Преподаватель:

Николаев Владимир Вячеславович

Санкт-Петербург

2024г.

Задание

Цель работы - на выделенном узле создать и сконфигурировать новый кластер БД Postgres, саму БД, табличные пространства и новую роль, а также произвести наполнение базы в соответствии с заданием. Отчёт по работе должен содержать все команды по настройке, скрипты, а также измененные строки конфигурационных файлов.

Способ подключения к узлу из сети Интернет через helios:

ssh -J sXXXXXX@helios.cs.ifmo.ru:2222 postgresY@pgZZZ

Способ подключения к узлу из сети факультета:

ssh postgresY@pgZZZ

Номер выделенного узла pgZZZ, а также логин и пароль для подключения Вам выдаст преподаватель.

Этап 1. Инициализация кластера БД

- Директория кластера: $HOME/khk43
- Кодировка: ISO_8859_5
- Локаль: русская
- Параметры инициализации задать через аргументы команды

Этап 2. Конфигурация и запуск сервера БД

- Способы подключения: 1) Unix-domain сокет в режиме peer; 2) сокет TCP/IP, принимать подключения к любому IP-адресу узла
- Номер порта: 9555
- Способ аутентификации TCP/IP клиентов: по имени пользователя
- Остальные способы подключений запретить.
- 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 с аппаратной конфигурацией:
оперативная память 4ГБ, хранение на жёстком диске (HDD).

- Директория WAL файлов: $HOME/oka84

- Формат лог-файлов: .csv
- Уровень сообщений лога: ERROR
- Дополнительно логировать: завершение сессий и продолжительность выполнения команд

Этап 3. Дополнительные табличные пространства и наполнение базы

- Создать новые табличные пространства для временных объектов: $HOME/mqb89, $HOME/utr38
- На основе template0 создать новую базу: uglyredbird
- Создать новую роль, предоставить необходимые права, разрешить подключение к базе.
- От имени новой роли (не администратора) произвести наполнение ВСЕХ созданных баз тестовыми наборами данных. ВСЕ табличные пространства должны использоваться по назначению.
- Вывести список всех табличных пространств кластера и содержащиеся в них объекты.

Реализация

Этап 1. Инициализация кластера БД

Создаем директорию кластера и инициализируем базу данных:

```
mkdir -p $PGDATA
chown $PGUSERNAME $PGDATA
initdb -D $PGDATA --encoding=$PGENCODE --locale=$PGLOCALE --
username=$PGUSERNAME
```

Результат:

```
[postgres1@pg167 ~]$ initdb -D $PGDATA --encoding=$PGENCODE --locale=$PGLOCALE --username=$PGUSERNAME
Файлы, относящиеся к этой СУБД, будут принадлежать пользователю "postgres1".
От его имени также будет запускаться процесс сервера.

Кластер баз данных будет инициализирован с локалью "ru_RU.ISO8859-5".
Выбрана конфигурация текстового поиска по умолчанию "russian".

Контроль целостности страниц данных отключён.

исправление прав для существующего каталога /var/db/postgres1/khk43... ок
создание подкаталогов... ок
выбирается реализация динамической разделяемой памяти... posix
выбирается значение max_connections по умолчанию... 100
выбирается значение shared_buffers по умолчанию... 128MB
выбирается часовой пояс по умолчанию... Europe/Moscow
создание конфигурационных файлов... ок
выполняется подготовительный скрипт... ок
выполняется заключительная инициализация... ок
сохранение данных на диске... ок

initdb: предупреждение: включение метода аутентификации "trust" для локальных подключений
initdb: подсказка: Другой метод можно выбрать, отредактировав pg_hba.conf или ещё раз запустив initdb с ключом -A, --auth-local ил
и --auth-host.

Готово. Теперь вы можете запустить сервер баз данных:

    pg_ctl -D /var/db/postgres1/khk43 -l файл_журнала start

[postgres1@pg167 ~]$
```

Этап 2. Конфигурация и запуск сервера БД

Скачиваем конфигурационные файлы:

```
scp postgres1@pg167:khk43/postgresql.conf ~
scp postgres1@pg167:khk43/pg_hba.conf ~
```

Настройка способов подключения

Редактируем файл postgresql.conf:

- сокет TCP/IP, принимать подключения к любому IP-адресу узла
- Номер порта: 9555

```
# - Connection Settings -

listen_addresses = '*'          # what IP address(es) to listen on;
                    # comma-separated list of addresses;
                    # defaults to 'localhost'; use '*' for all
                    # (change requires restart)
port = 9555                 # (change requires restart)
password_encryption = md5   # scram-sha-256 or md5
```

Редактируем файл pg_hba.conf:

- Unix-domain сокет в режиме peer

- Способ аутентификации TCP/IP клиентов: по имени пользователя
- Остальные способы подключений запретить. Из-за проблем ident в Гелиос, меняем на md5.

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD

# Разрешить локальные подключения через Unix-domain сокет с
аутентификацией peer
local   all             all                                     peer
# Разрешить TCP/IP подключения со всех IP-адресов с аутентификацией по
имени пользователя (ident)
host    all             all             0.0.0.0/0               md5
host    all             all             ::/0                    md5
# Запретить все остальные подключения
local   replication     all                                     reject
host    replication     all             127.0.0.1/32            reject
host    replication     all             ::1/128                 reject
```

Параметры сервера БД

max_connections:

```
max_connections = 100
```

shared_buffers: Ставим 1/4 от оперативной памяти согласно документации postgresql, т.е. 1ГБ.

```
shared_buffers = 1GB
```

temp_buffers: Количество памяти, выделенной для временных таблиц на одну сессию. Учитывая максимальное количество соединений, temp_buffers займут 100 * 16MB = 1600MB.

```
temp_buffers = 16MB
```

work_mem: Количество памяти, выделенной для операций сортировки и хеширования на одно соединение. Не зная какого вида операции будут производиться, (сложные соединения и сортировки или простые запросы) то оставляем значение по умолчанию 4MB. Work_mem максимально может занимать 100 * 4MB = 400MB.

```
work_mem = 4MB
```

checkpoint_timeout: Интервал времени между контрольными точками (checkpoints). Контрольные точки обеспечивают согласованность данных на диске. Учитывая, что у нас HDD, более длинный интервал времени между контрольными точками уменьшит нагрузку на диск.

```
checkpoint_timeout = 15min
```

effective_cache_size: Этот параметр представляет собой оценку для планировщика о количестве дискового кэша, доступного для PostgreSQL. Это значение должно быть больше shared_buffers. Учитывая, что у нас HDD,

то операции ввода-вывода будут медленными, поэтому считывание из кэша будет предпочтительнее. Так, ставим 75% от оперативной памяти, т.е. 3ГБ.

```
effective_cache_size = 3GB
```

fsync: Этот параметр должен быть включен для обеспечения безопасности данных в случае сбоя системы. Отключение этого параметра может улучшить производительность, но риск потери данных в случае сбоя неприемлем для большинства производственных систем.

```
fsync = on
```

commit_delay: Этот параметр задает задержку в миллисекундах перед сохранением WAL. Без тестирования сложно подобрать оптимальное значение. По умолчанию 0.

```
commit_delay = 0
```

WAL файлы и логирование:

Директория WAL файлов:

Создадим директорию для WAL файлов:

```
mkdir -p $HOME/oka84
chown $PGUSERNAME $HOME/oka84
```

- archive_mode - включает архивирование WAL файлов.
- archive_command - команда, которая будет выполняться для архивирования WAL файлов. В данном случае, копируем файл в директорию $HOME/oka84.

```
archive_mode = on
archive_command = 'cp %p $HOME/oka84/%f'
```

Формат лог-файлов:

- log_destination - куда писать логи. В данном случае, в файл csv.
- logging_collector - включает сборщик логов и позволяет перенаправлять в файлы.
- log_directory - директория для логов. Оставляем по умолчанию.
- log_filename - формат имени файла лога. Ставим формат csv.

```
log_destination = 'csvlog'
logging_collector = on
log_directory = 'log'
log_filename = 'postgresql-%Y-%m-%d_%H%M%S.csv'
```

Уровень сообщений лога:

- log_min_messages - минимальный уровень сообщений, которые будут записаны в лог. В данном случае, только ошибки и выше.

```
log_min_messages = error
```

Дополнительно логировать:

- log_connections - логировать подключения.
- log_disconnections - логировать отключения. Оба параметра используем для отслеживания завершения сессий.
- log_duration - логировать продолжительность выполнения

команд.

- log_min_duration_statement - минимальная продолжительность выполнения команды, которая будет логироваться. В данном случае 0 - логировать все команды.

```
log_connections = on
log_disconnections = on
log_duration = on
log_min_duration_statement = 0
```

Запуск сервера БД

Загрузим обратно конфигурационные файлы:

```
scp ~/postgresql.conf postgres1@pg167:khk43
scp ~/pg_hba.conf postgres1@pg167:khk43
```

Запускаем сервер:

```
pg_ctl -D /var/db/postgres1/khk43 -l файл_журнала start
```

Проверка всех параметров

Статус сервера:

```
[postgres1@pg167 ~]$ pg_ctl -D ~/khk43 status
pg_ctl: сервер работает (PID: 63080)
/usr/local/bin/postgres "-D" "/var/db/postgres1/khk43"
```

Подключение локально:

```
[postgres1@pg167 ~]$ psql -p 9555 -d postgres
psql (16.4)
Введите "help", чтобы получить справку.

postgres=#
```

Подключение удаленно

Создадим нового пользователя PostgreSQL с паролем:

```
CREATE ROLE testuser WITH LOGIN PASSWORD 'testpassword';
```

Попробуем подключиться удаленно:

```
[s372819@helios ~]$ psql -h pg167 -p 9555 -U testuser -d postgres
Пароль пользователя testuser:
psql (16.4)
Введите "help", чтобы получить справку.

postgres=>
```

Проверка параметров:

```
postgres=# SHOW max_connections;
SHOW shared_buffers;
SHOW temp_buffers;
SHOW work_mem;
SHOW checkpoint_timeout;
SHOW effective_cache_size;
SHOW fsync;
SHOW commit_delay;
 max_connections
-----------------
```

```
 100
(1 строка)

 shared_buffers
----------------
 1GB
(1 строка)

 temp_buffers
--------------
 16MB
(1 строка)

 work_mem
----------
 4MB
(1 строка)

 checkpoint_timeout
--------------------
 15min
(1 строка)

 effective_cache_size
----------------------
 3GB
(1 строка)

 fsync
-------
 on
(1 строка)

 commit_delay
--------------
 0
(1 строка)

postgres=#
```

Этап 3. Дополнительные табличные пространства и наполнение базы

Создание табличных пространств

```
mkdir -p /var/db/postgres1/mqb89
mkdir -p /var/db/postgres1/utr38
```

```
CREATE TABLESPACE mqb89 LOCATION '/var/db/postgres1/mqb89';
CREATE TABLESPACE utr38 LOCATION '/var/db/postgres1/utr38';
```

Проверка:

```
postgres=# \db
           Список табличных пространств
    Имя      | Владелец  |       Расположение
-------------+-----------+--------------------------
 mqb89       | postgres1 | /var/db/postgres1/mqb89
 pg_default  | postgres1 |
 pg_global   | postgres1 |
 utr38       | postgres1 | /var/db/postgres1/utr38
(4 строки)
```

Создание базы данных

```
CREATE DATABASE uglyredbird TEMPLATE template0;
```

```
postgres=# \l
                                                  Список баз данных
    Имя     | Владелец  | Кодировка  | Провайдер локали |   LC COLLATE    |    LC CTYPE     | локаль ICU | Правила ICU |      Права доступа
------------+-----------+------------+------------------+-----------------+-----------------+------------+-------------+-----------------------
 postgres   | postgres1 | ISO 8859 5 | libc             | ru_RU.ISO8859-5 | ru_RU.ISO8859-5 |            |             |
 template0  | postgres1 | ISO_8859_5 | libc             | ru_RU.ISO8859-5 | ru_RU.ISO8859-5 |            |             | =c/postgres1          +
            |           |            |                  |                 |                 |            |             | postgres1=CTc/postgres1
 template1  | postgres1 | ISO_8859_5 | libc             | ru_RU.ISO8859-5 | ru_RU.ISO8859-5 |            |             | =c/postgres1          +
            |           |            |                  |                 |                 |            |             | postgres1=CTc/postgres1
 uglyredbird | postgres1 | ISO_8859_5 | libc             | ru_RU.ISO8859-5 | ru_RU.ISO8859-5 |            |             |
(4 строки)
```

Создание роли

```
CREATE ROLE newuser WITH LOGIN;  --Пароль не нужен так как используем
подключение peer
-- Предоставить необходимые права
GRANT CONNECT, CREATE ON DATABASE uglyredbird TO newuser;
GRANT CREATE ON TABLESPACE mqb89 TO newuser;
GRANT CREATE ON TABLESPACE utr38 TO newuser;
```

Наполнение созданных баз тестовыми наборами данных.

Запускаем скрипт наполнения базы от имени нового пользователя:

```
psql -p 9555 -d uglyredbird -U newuser -f $HOME/newuser.sql
```

Проверка:

```
uglyredbird=> SELECT * FROM pg_catalog.pg_tables WHERE tableowner = 'newuser';
 schemaname |       tablename        | tableowner | tablespace | hasindexes | hasrules | hastriggers |
rowsecurity
------------+------------------------+------------+------------+------------+----------+-------------+-----
--------
 main       | students               | newuser    |            | t          | f        | f           | f
 main       | courses                | newuser    |            | t          | f        | f           | f
 pg_temp_3  | temp_enrollments       | newuser    | mqb89      | t          | f        | f           | f
 pg_temp_3  | temp_course_statistics | newuser    | utr38      | f          | f        | f           | f
(4 строки)
```

Вывести список всех табличных пространств кластера и содержащиеся в них объекты

Выведем все табличные пространства

```
uglyredbird=> SELECT * FROM pg_tablespace;
  oid  |  spcname   | spcowner |                    spcacl                    | spcoptions
-------+------------+----------+----------------------------------------------+------------
  1663 | pg_default |       10 |                                              |
  1664 | pg_global  |       10 |                                              |
 16389 | mqb89      |       10 | {postgres1=C/postgres1,newuser=C/postgres1} |
 16390 | utr38      |       10 | {postgres1=C/postgres1,newuser=C/postgres1} |
(4 строки)
```

Выведем все объекты в табличных пространствах

```
SELECT
    spcname AS tablespace,
    relname
FROM
    pg_class
    JOIN pg_tablespace ON pg_tablespace.oid = reltablespace;
```

```
 tablespace |                 relname
------------+----------------------------------------
 pg_global  | pg_toast_1262
 pg_global  | pg_toast_1262_index
 pg_global  | pg_toast_2964
 pg_global  | pg_toast_2964_index
 pg_global  | pg_toast_1213
 pg_global  | pg_toast_1213_index
 pg_global  | pg_toast_1260
 pg_global  | pg_toast_1260_index
 pg_global  | pg_toast_2396
 pg_global  | pg_toast_2396_index
 pg_global  | pg_toast_6000
 pg_global  | pg_toast_6000_index
 pg_global  | pg_toast_3592
 pg_global  | pg_toast_3592_index
 pg_global  | pg_toast_6243
 pg_global  | pg_toast_6243_index
 pg_global  | pg_toast_6100
 pg_global  | pg_toast_6100_index
 pg_global  | pg_database_datname_index
 pg_global  | pg_database_oid_index
 pg_global  | pg_db_role_setting_databaseid_rol_index
 pg_global  | pg_tablespace_oid_index
 pg_global  | pg_tablespace_spcname_index
 pg_global  | pg_authid_rolname_index
 pg_global  | pg_authid_oid_index
 pg_global  | pg_auth_members_oid_index
 pg_global  | pg_auth_members_role_member_index
 pg_global  | pg_auth_members_member_role_index
 pg_global  | pg_auth_members_grantor_index
```

```
 pg_global  | pg_shdepend_depender_index
 pg_global  | pg_shdepend_reference_index
 pg_global  | pg_shdescription_o_c_index
 pg_global  | pg_replication_origin_roiident_index
 pg_global  | pg_replication_origin_roname_index
 pg_global  | pg_shseclabel_object_index
 pg_global  | pg_parameter_acl_parname_index
 pg_global  | pg_parameter_acl_oid_index
 pg_global  | pg_subscription_oid_index
 pg_global  | pg_subscription_subname_index
 pg_global  | pg_authid
 mqb89      | temp_enrollments
 utr38      | temp_course_statistics
 pg_global  | pg_subscription
 pg_global  | pg_database
 pg_global  | pg_db_role_setting
 pg_global  | pg_tablespace
 pg_global  | pg_auth_members
 pg_global  | pg_shdepend
 pg_global  | pg_shdescription
 pg_global  | pg_replication_origin
 pg_global  | pg_shseclabel
 pg_global  | pg_parameter_acl
(52 строки)
```

Выведем все объекты, созданные новым пользователем:

```
SELECT
    relname, spcname AS tablespace
FROM
    pg_class LEFT JOIN pg_tablespace ON pg_tablespace.oid =
reltablespace
WHERE
    relowner = (SELECT oid FROM pg_roles WHERE rolname = 'newuser');
```

```
           relname            | tablespace
------------------------------+------------
 students_student_id_seq      |
 students                     |
 students_pkey                |
 courses_course_id_seq        |
 courses                      |
 courses_pkey                 |
 temp_enrollments_temp_id_seq |
 temp_enrollments             | mqb89
 temp_enrollments_pkey        |
 temp_course_statistics       | utr38
(10 строк)
```

Вывод

В результате выполнения лабораторной работы были успешно достигнуты все поставленные цели, связанные с созданием и конфигурированием нового кластера базы данных PostgreSQL на выделенном узле. Работа включала три основных этапа: инициализацию кластера БД, конфигурацию и запуск сервера БД, а также создание дополнительных табличных пространств и наполнение базы тестовыми данными. Лабораторная работа позволила углубить знания и приобрести практические навыки в настройке и управлении кластером базы данных PostgreSQL.