MainPage/DataBase/Lab 2

Университет ИТМО

Факультет ФПИ и КТ

Отчёт по лабораторной работе 2

«Распределенные системы хранения данных»

Вариант: 806

Студент: Чжоу Хунсян

Группа: P33131

Преподаватель:

1 Текст задания.

Внимание! У разных вариантов разный текст задания!

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 s336184@helios.cs.ifmo.ru:2222 postgres2@pg117
```

Способ подключения к узлу из сети факультета:

```
ssh postgres2@pg117
```

Номер выделенного узла pg117, а также логин и пароль для подключения Вам выдаст преподаватель.

> pg117:postgres2:FrMa3dca

Этап 1. Инициализация кластера БД

Директория кластера: `$HOME/ewe49`
Кодировка: KOI8-R
Локаль: русская
Параметры инициализации задать через переменные окружения

```
ssh -J s336184@helios.cs.ifmo.ru:2222 postgres2@pg117
```

```
[postgres2@pg117 ~]$ export PGDATA=$HOME/ewe49
[postgres2@pg117 ~]$ export LANG=ru_RU.KOI8-R
```

```
[postgres2@pg117 ~]$ echo $PGDATA
/var/db/postgres2/ewe49
[postgres2@pg117 ~]$ echo $LANG
ru_RU.KOI8-R
```

```
[postgres2@pg117 ~]$ initdb -D $PGDATA --locale=$LANG
Файлы, относящиеся к этой СУБД, будут принадлежать пользователю "postgres2".
От его имени также будет запускаться процесс сервера.

Кластер баз данных будет инициализирован с локалью "ru_RU.KOI8-R".
Кодировка БД по умолчанию, выбранная в соответствии с настройками: "KOI8R".
Выбрана конфигурация текстового поиска по умолчанию "russian".

Контроль целостности страниц данных отключён.

создание каталога /var/db/postgres2/ewe49... 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... W-SU
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
Другой метод можно выбрать, отредактировав pg_hba.conf или используя ключи -A,
--auth-local или --auth-host при следующем выполнении initdb.

Готово. Теперь вы можете запустить сервер баз данных:

    pg_ctl -D /var/db/postgres2/ewe49 -l файл_журнала start
```

Этап 2. Конфигурация и запуск сервера БД

- Способы подключения:
    i. Unix-domain сокет в режиме peer;
    ii. сокет TCP/IP, принимать подключения к любому IP-адресу узла
- Номер порта: 9806
- 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 12ГБ, хранение на жёстком диске (HDD).
- Директория WAL файлов: $HOME/svq55
- Формат лог-файлов: .csv
- Уровень сообщений лога: ERROR
- Дополнительно логировать: завершение сессий и продолжительность выполнения команд

```
# Connection Settings
listen_addresses = '*'
port = 9806

# Resource Consumption
max_connections = 100
shared_buffers = 3GB
temp_buffers = 16MB
work_mem = 16MB

# Checkpoints
checkpoint_timeout = 10min

# Memory
effective_cache_size = 9GB

# Write-Ahead Logging (WAL)
fsync = on
commit_delay = 0

# WAL Directory
wal_level = replica
archive_mode = on
archive_command = 'cp %p $HOME/svq55/%f'

# Logging
logging_collector = on
log_destination = 'csvlog'
log_directory = 'log'
log_filename = 'postgresql-%Y-%m-%d_%H%M%S.log'
log_truncate_on_rotation = on
log_rotation_age = 1d
log_rotation_size = 0
log_statement = 'none'
log_duration = on
log_connections = on
log_disconnections = on
log_min_messages = error
```

```
# "local" is for Unix domain socket connections only
local   all             all                                     peer

# IPv4 local connections:
host    all             all             127.0.0.1/32            md5
host    all             all             0.0.0.0/0               md5

# IPv6 local connections:
host    all             all             ::1/128                 md5
host    all             all             ::/0                    md5

#
hostnossl all           all             0.0.0.0/0               reject
hostssl  all            all             0.0.0.0/0               md5
```

```
[postgres2@pg117 ~/ewe49]$ pg_ctl -D $PGDATA -l logfile restart
ожидание завершения работы сервера.... готово
сервер остановлен
ожидание запуска сервера.... готово
сервер запущен
```

use UNIX Socket with peer to connect postgres database

```
[postgres2@pg117 ~/ewe49]$ psql -U postgres2 -p 9806 -d postgres
psql (14.2)
Введите "help", чтобы получить справку.

postgres=#
```

```
postgres=# alter user postgres2 with password '123456';
ALTER ROLE
```

Этап 3. Дополнительные табличные пространства и наполнение базы

- Создать новые табличные пространства для партицированной таблицы: `$HOME/gje71`, `$HOME/xca33`
- На основе template1 создать новую базу: fatbluefish
- 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.

```
[postgres2@pg117 ~]$ mkdir $HOME/gje71
[postgres2@pg117 ~]$ mkdir $HOME/xca33
[postgres2@pg117 ~]$ ls
ewe49        gje71        xca33
```

```
[postgres2@pg117 ~/gje71]$ psql -U postgres2 -p 9806 -d postgres
psql (14.2)
Введите "help", чтобы получить справку.

postgres=# CREATE TABLESPACE ts_gje71 LOCATION '/var/db/postgres2/gje71';
CREATE TABLESPACE
postgres=# CREATE TABLESPACE ts_xca33 LOCATION '/var/db/postgres2/xca33';
CREATE TABLESPACE
```

```
postgres=# \l
 postgres  | postgres2 | KOI8R     | ru_RU.KOI8-R | ru_RU.KOI8-R |
 template0 | postgres2 | KOI8R     | ru_RU.KOI8-R | ru_RU.KOI8-R | =c/postgres2
           |           |           |              |              | postgres2=CTc/postgres2
 template1 | postgres2 | KOI8R     | ru_RU.KOI8-R | ru_RU.KOI8-R | =c/postgres2
           |           |           |              |              | postgres2=CTc/postgres2
```

```
postgres=# CREATE DATABASE fatbluefish TEMPLATE template1;
CREATE DATABASE
```

```
postgres=# \l
 fatbluefish | postgres2 | KOI8R     | ru_RU.KOI8-R | ru_RU.KOI8-R |
 postgres    | postgres2 | KOI8R     | ru_RU.KOI8-R | ru_RU.KOI8-R |
 template0   | postgres2 | KOI8R     | ru_RU.KOI8-R | ru_RU.KOI8-R | =c/postgres2
             |           |           |              |              | postgres2=CTc/postgre
 template1   | postgres2 | KOI8R     | ru_RU.KOI8-R | ru_RU.KOI8-R | =c/postgres2
             |           |           |              |              | postgres2=CTc/postgre
```

```
postgres=# CREATE ROLE tolia WITH LOGIN PASSWORD '123456';
CREATE ROLE
postgres=# GRANT CONNECT ON DATABASE fatbluefish TO tolia;
GRANT
postgres=# GRANT ALL PRIVILEGES ON DATABASE fatbluefish TO tolia;
GRANT
postgres=# GRANT CREATE ON TABLESPACE ts_gje71 TO tolia;
GRANT
postgres=# GRANT CREATE ON TABLESPACE ts_xca33 TO tolia;
GRANT
postgres=# \q
```

```
[postgres2@pg117 ~/gje71]$ psql -U tolia -d fatbluefish -p 9806 -h localhost
Пароль пользователя tolia:
psql (14.2)
Введите "help", чтобы получить справку.

fatbluefish=>
```

```
fatbluefish=> CREATE TABLE fish (
    id SERIAL PRIMARY KEY,
    name TEXT NOT NULL
) TABLESPACE ts_gje71;

CREATE TABLE fish_info (
    id SERIAL PRIMARY KEY,
    fish_id INT REFERENCES fish(id),
    info TEXT NOT NULL
) TABLESPACE ts_xca33;
CREATE TABLE
CREATE TABLE
```

```
fatbluefish=> INSERT INTO fish (name) VALUES
('Goldfish'),
('Tuna'),
('Salmon');
INSERT 0 3
fatbluefish=> INSERT INTO fish_info (fish_id, info) VALUES
(1, 'Goldfish are small freshwater fish.'),
(2, 'Tuna are large saltwater fish.'),
(3, 'Salmon are medium-sized fish that swim upstream to spawn.');
INSERT 0 3
fatbluefish=> \q
```

```
[postgres2@pg117 ~/gje71]$ psql -U postgres2 -p 9806 -d fatbluefish
psql (14.2)
Введите "help", чтобы получить справку.

fatbluefish=# SELECT spcname AS "Name", pg_tablespace_location(oid) AS "Location" FROM pg_
    Name    |        Location
------------+------------------------
 pg_default |
 pg_global  |
 ts_gje71   | /var/db/postgres2/gje71
 ts_xca33   | /var/db/postgres2/xca33
(4 строки)

fatbluefish=#
```

```
fatbluefish=# SELECT
    t.spcname 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_class c
JOIN
    pg_tablespace t ON c.reltablespace = t.oid
ORDER BY
    t.spcname, c.relname;
 Tablespace |               Object Name               | Object Type
------------+-----------------------------------------+-------------
 pg_global  | pg_auth_members                         | Table
 pg_global  | pg_auth_members_member_role_index       | Index
 pg_global  | pg_auth_members_role_member_index       | Index
 pg_global  | pg_authid                               | Table
 pg_global  | pg_authid_oid_index                     | Index
 pg_global  | pg_authid_rolname_index                 | Index
 pg_global  | pg_database                             | Table
 pg_global  | pg_database_datname_index               | Index
 pg_global  | pg_database_oid_index                   | Index
 pg_global  | pg_db_role_setting                      | Table
 pg_global  | pg_db_role_setting_databaseid_rol_index | Index
 pg_global  | pg_replication_origin                   | Table
 pg_global  | pg_replication_origin_roiident_index    | Index
 pg_global  | pg_replication_origin_roname_index      | Index
 pg_global  | pg_shdepend                             | Table
 pg_global  | pg_shdepend_depender_index              | Index
 pg_global  | pg_shdepend_reference_index             | Index
 pg_global  | pg_shdescription                        | Table
```

```
 pg_global  | pg_shdescription_o_c_index             | Index
 pg_global  | pg_shseclabel                          | Table
 pg_global  | pg_shseclabel_object_index             | Index
 pg_global  | pg_subscription                        | Table
 pg_global  | pg_subscription_oid_index              | Index
 pg_global  | pg_subscription_subname_index          | Index
 pg_global  | pg_tablespace                          | Table
 pg_global  | pg_tablespace_oid_index                | Index
 pg_global  | pg_tablespace_spcname_index            | Index
 pg_global  | pg_toast_1213                          | TOAST Table
 pg_global  | pg_toast_1213_index                    | Index
 pg_global  | pg_toast_1260                          | TOAST Table
 pg_global  | pg_toast_1260_index                    | Index
 pg_global  | pg_toast_1262                          | TOAST Table
 pg_global  | pg_toast_1262_index                    | Index
 pg_global  | pg_toast_2396                          | TOAST Table
 pg_global  | pg_toast_2396_index                    | Index
 pg_global  | pg_toast_2964                          | TOAST Table
 pg_global  | pg_toast_2964_index                    | Index
 pg_global  | pg_toast_3592                          | TOAST Table
 pg_global  | pg_toast_3592_index                    | Index
 pg_global  | pg_toast_6000                          | TOAST Table
 pg_global  | pg_toast_6000_index                    | Index
 pg_global  | pg_toast_6100                          | TOAST Table
 pg_global  | pg_toast_6100_index                    | Index
 ts_gje71   | fish                                   | Table
 ts_gje71   | pg_toast_16391                         | TOAST Table
 ts_gje71   | pg_toast_16391_index                   | Index
 ts_xca33   | fish_info                              | Table
 ts_xca33   | pg_toast_16400                         | TOAST Table
 ts_xca33   | pg_toast_16400_index                   | Index
(49 строк)
```

Выводы

В ходе выполнения работы научился создавать, инициализировать, настраивать и использовать базы данных с помощью команд.