

# УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Распределенные системы хранения данных»

## **Лабораторная работа №2**

*Вариант 65*

Студент

*Ибадуллаев А. Э.*

*P33131*

Преподаватель

*Шешуков Д. М.*

Санкт-Петербург, 2023 г.

## Описание задания

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) ssh [s666666@se.ifmo.ru](mailto:s666666@se.ifmo.ru) -p 2222
- 2) ssh пользователь@узел

Персональный пароль для работы с узлом выдается преподавателем.

Обратите внимание, что домашняя директория пользователя  
/var/postgres/\$LOGNAME

Этапы выполнения работы:

Инициализация кластера БД

- Имя узла — pg101.
- Имя пользователя — postgres3.
- Директория кластера БД — \$HOME/u01/dci13.
- Кодировка, локаль — ANSI1251, русская
- Перечисленные параметры задать через переменные окружения.

Конфигурация и запуск сервера БД

- Способ подключения к БД — TCP/IP socket, номер порта 9005.
- Остальные способы подключений запретить.
- Способ аутентификации клиентов — по паролю SHA-256.
- 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 OLTP: 500 транзакций/сек. с записью размером по 4 КБ, акцент на высокую доступность данных;

- Директория WAL файлов — \$HOME/u02/dci13.
- Формат лог-файлов — log.
- Уровень сообщений лога — INFO.
- Дополнительно логировать — контрольные точки.

Дополнительные табличные пространства и наполнение

- Создать новое табличное пространство для индексов:  
? \$HOME/u03/dci13.
- На основе template0 создать новую базу — whitebunn.
- 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.

## Выполнение

### Инициализируем кластер базы данных

```
PGDATA=$HOME/u01/dci13
PGENCOD=WIN1251
PGLOCALE=ru_RU.CP1251
PGSUSERSNAME=postgres3
export PGDATA PGLOCALE PGENCOD PGSUSERSNAME

mkdir -p $PGDATA
chown postgres3 $PGDATA
initdb --encoding=$PGENCOD --locale=$PGLOCALE --
username=$PGSUSERSNAME
```

Создал .profile файл для экспорта переменных окружения при запуске

```
[postgres3@pg101 ~]$ initdb --encoding=$PGENCOD --locale=$PGLOCALE --username=$PGSUSERSNAME
Файлы, относящиеся к этой СУБД, будут принадлежать пользователю "postgres3".
От его имени также будет запускаться процесс сервера.

Кластер баз данных будет инициализирован с локалью "ru_RU.CP1251".
Выбрана конфигурация текстового поиска по умолчанию "russian".

Контроль целостности страниц данных отключён.

исправление прав для существующего каталога /var/db/postgres3/u01/dci13... 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/postgres3/u01/dci13 -l файл_журнала start
```

### pg\_hba.conf

| # TYPE   | DATABASE    | USER | ADDRESS      | METHOD        |
|--|-------------|------|--------------|---------------|
| host   | all         | all  | all          | scram-sha-256 |
| # "local" is for Unix domain socket connections only               |             |      |              |               |
| local  | all         | all  |              | reject        |
| # IPv4 local connections:  |             |      |              |               |
| host   | all         | all  | 127.0.0.1/32 | reject        |
| # IPv6 local connections:  |             |      |              |               |
| host   | all         | all  | ::1/128      | reject        |
| # Allow replication connections from localhost, by a user with the |             |      |              |               |
| # replication privilege.   |             |      |              |               |
| local  | replication | all  |              | reject        |
| host   | replication | all  | 127.0.0.1/32 | reject        |
| host   | replication | all  | ::1/128      | reject        |

### postgresql.conf

```

# - Connection Settings -

#listen_addresses = 'localhost'          # what IP address(es) to listen on;
                                          # comma-separated list of addresses;
                                          # defaults to 'localhost'; use '*' for all
                                          # (change requires restart)
port = 9005                              # (change requires restart)
max_connections = 500                    # (change requires restart)

# - Authentication -

#authentication_timeout = 1min           # 1s-600s
password_encryption = scram-sha-256     # scram-sha-256 or md5
#db_user_namespace = off

log_min_messages = info                 # values in order of decreasing detail:
                                          # _debug_

# - Archiving -

archive_mode = on                       # enables archiving; off, on, or always
                                          # (change requires restart)
archive_command = 'cp %p $HOME/u02/dci13/%f' # command to use to archive a logfile

# - Settings -

#wal_level = replica                   # minimal, replica, or logical
                                          # (change requires restart)
fsync = on                             # flush data to disk for crash safety
effective_cache_size = 2GB             # (change requires restart)
                                          # CSVLOGS.
                                          # (change requires restart)

# These are only used if logging_collector is on:
log_directory = 'log'                  # directory where log files are written,
                                          # can be absolute or relative to PGDATA

# - Memory -

shared_buffers = 2GB                   # min 128kB
                                          # (change requires restart)
#huge_pages = try                       # on, off, or try
                                          # (change requires restart)
#huge_page_size = 0                     # zero for system default
                                          # (change requires restart)
temp_buffers = 1MB                     # min 800kB
max_prepared_transactions = 1000       # zero disables the feature
                                          # (change requires restart)
# Caution: it is not advisable to set max_prepared_transactions nonzero unless
# you actively intend to use prepared transactions.
work_mem = 1MB                         # min 64kB

log_checkpoints = on

# - Checkpoints -

#checkpoint_timeout = 5min              # range 30s-1d
#checkpoint_completion_target = 0.9     # checkpoint target duration, 0.0 - 1.0
#checkpoint_flush_after = 0             # measured in pages, 0 disables
#checkpoint_warning = 30s               # 0 disables
max_wal_size = 2GB
min_wal_size = 80MB

```

### 3anyck Postgresql

```
pg_ctl start -l logfile
```

```
psql -p 9005 -d postgres -h localhost
```

```
CREATE TABLESPACE indexspace LOCATION
'/var/db/postgres3/u03/dci13';
```

```
DROP DATABASE postgres;
```

```
CREATE DATABASE whitebunn WITH TEMPLATE = template0 TABLESPACE  
=indexspace;
```

```
CREATE ROLE alibaba LOGIN PASSWORD 'alibaba';
```

```
CREATE TABLE bunn ( id bigserial primary key, name text );
```

```
CREATE INDEX ON bunn (name) TABLESPACE indexspace;
```

```
GRANT INSERT ON bunn TO alibaba;
```

```
GRANT USAGE, SELECT ON SEQUENCE bunn_id_seq TO alibaba;
```

```
psql -U alibaba -h localhost -d whitebunn -p 9005 -f insert.sql
```

[insert.sql](#):

```
INSERT INTO bunn (name) VALUES('Postgre');
```

```
INSERT INTO bunn (name) VALUES('Postgre');
```

```
INSERT INTO bunn (name) VALUES('Postgre');
```

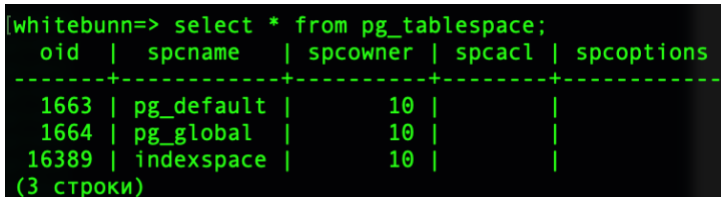
```
INSERT INTO bunn (name) VALUES('Postgre');
```

```
INSERT INTO bunn (name) VALUES('Postgre');
```

```
INSERT INTO bunn (name) VALUES('Postgre');
```

## Выводим табличные пространства и их содержимое

```
select * from pg_tablespace;
```



```
whitebunn=> select * from pg_tablespace;  
oid | spcname | spcowner | spcacl | spcoptions  
-----+-----+-----+-----+-----  
1663 | pg_default | 10 | |  
1664 | pg_global | 10 | |  
16389 | indexspace | 10 | |  
(3 строки)
```

```
SELECT c.relname, t.spcname FROM pg_class c JOIN pg_tablespace t  
ON c.reltablespace = t.oid;
```

```

whitebunn=> SELECT c.relname, t.spcname FROM pg_class c JOIN pg_tablespace t ON c.reltablespace =
t.oid;

```

| relname                                 | spcname   |
|---|-----------|
| 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_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_role_member_index       | pg_global |
| pg_auth_members_member_role_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_roident_index     | pg_global |
| pg_replication_origin_roname_index      | pg_global |
| pg_shseclabel_object_index              | pg_global |
| pg_subscription_oid_index               | pg_global |
| pg_subscription_subname_index           | pg_global |
| pg_authid                               | 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 |

(43 строки)

```

SELECT d.datname, t.spcname FROM pg_tablespace t JOIN pg_database
d ON d.dattablespace = t.oid; SELECT d.datname, t.spcname FROM
pg_tablespace t JOIN pg_database d ON d.dattablespace = t.oid;

```

```

whitebunn=> SELECT d.datname, t.spcname FROM pg_tablespace t JOIN pg_database d ON d.dattablespace =
t.oid;

```

| datname   | spcname    |
|-----------|------------|
| postgres  | pg_default |
| template1 | pg_default |
| template0 | pg_default |
| whitebunn | pg_default |

(4 строки)

## Вывод

Во время выполнения лабораторной работы я научился создавать и конфигурировать кластер БД PostgreSQL. Я познакомился с созданием и работой табличных пространств и ролей, заполнил БД тестовыми данными.