МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО «СЕВЕРО–КАВКАЗСКАЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНТСТИТУТ ПЕРСПЕКТИВНОЙ ИНЖЕНЕРИИ
ДЕПАРТАМЕНТ ЦИФРОВЫХ, РОБОТОТЕХНИЧЕСКИХ СИСТЕМ И ЭЛЕКТРОНИКИ
МЕЖИНСТИТУТСКАЯ БАЗОВАЯ КАФЕДРА

Дисциплина: Тестирование и отладка программного обеспечения

# ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

**Выполнил:**
студент 4 курса направления подготовки 09.03.04 «Программная инженерия», направленность «Разработка и сопровождение программного обеспечения» группы ПИЖ-б-о-22-1

Джараян Арег Александрович

_____
(Подпись)

**Проверил:**
Ассистент департамента цифровых, робототехнических систем и электроники

Щеголев Алексей Алексеевич

_____
(Подпись)

Работа защищена с оценкой:
_____

Ставрополь, 2025г.

Цель: Изучить базовые компоненты архитектуры PostgreSQL (процессы, память) и получить практические навыки управления конфигурационными параметрами сервера на разных уровнях (экземпляр, сеанс). Освоить работу с основными и дополнительными файлами конфигурации, а также с представлениями pg_settings и pg_file_settings.

Часть 1: Исследование параметров и файлов конфигурации

1. Текущая конфигурация: Подключитесь к серверу с помощью psql. Определите расположение основного файла конфигурации (postgresql.conf) с помощью команды SHOW config_file;.

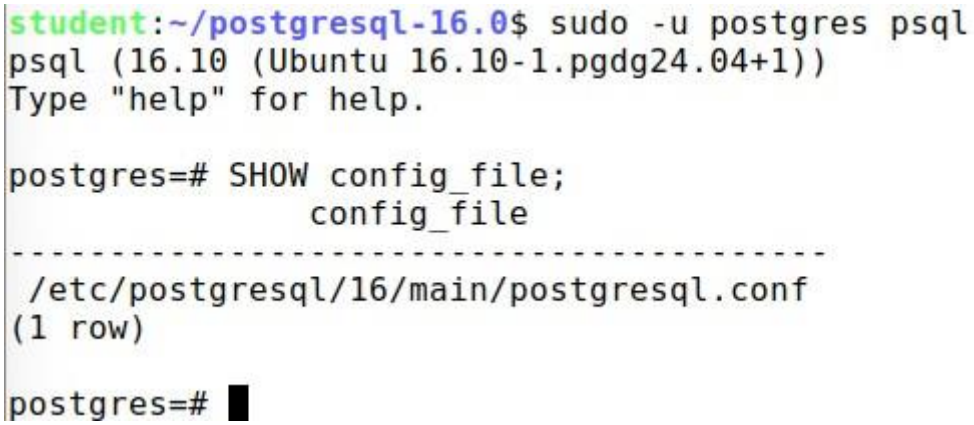


Рисунок 1 – Определение расположения файла

2. Анализ параметров: Изучите представление pg_settings. Найдите параметры, для изменения которых требуется перезагрузка сервера (context = 'postmaster'). Найдите 2-3 параметра с контекстом sighup и user.

```
              name                 |                setting                 |
 context
-----------------------------------+----------------------------------------+
-----------
 archive_mode                      | off                                    |
 postmaster
 autovacuum_freeze_max_age         | 200000000                              |
 postmaster
 autovacuum_max_workers            | 3                                      |
 postmaster
 autovacuum_multixact_freeze_max_age | 400000000                            |
 postmaster
 bonjour                           | off                                    |
 postmaster
 bonjour_name                      |                                        |
 postmaster
 cluster_name                      | 16/main                                |
 postmaster
 config_file                       | /etc/postgresql/16/main/postgresql.conf |
 postmaster
 data_directory                    | /var/lib/postgresql/16/main            |
 postmaster
 data_sync_retry                   | off                                    |
```

Рисунок 2 – Список параметров, требующих перезагрузку

```
postgres=# SELECT name, setting, context
FROM pg_settings
WHERE context = 'sighup'
LIMIT 3;
          name           |  setting   | context
-------------------------+------------+---------
 archive_cleanup_command |            | sighup
 archive_command         | (disabled) | sighup
 archive_library         |            | sighup
(3 rows)
```

Рисунок 3 – Параметры с контекстом sighup

```
postgres=# SELECT name, setting, context
FROM pg_settings
WHERE context = 'user'
LIMIT 3;
        name          | setting | context
----------------------+---------+--------
 application_name     | psql    | user
 array_nulls          | on      | user
 backend_flush_after  | 0       | user
(3 rows)

postgres=#
```

Рисунок 4 – Параметры с контекстом user

3. Анализ файлов: Изучите представление pg_file_settings. Определите, из каких файлов и с какими значениями были считаны текущие настройки параметров shared_buffers и work_mem.

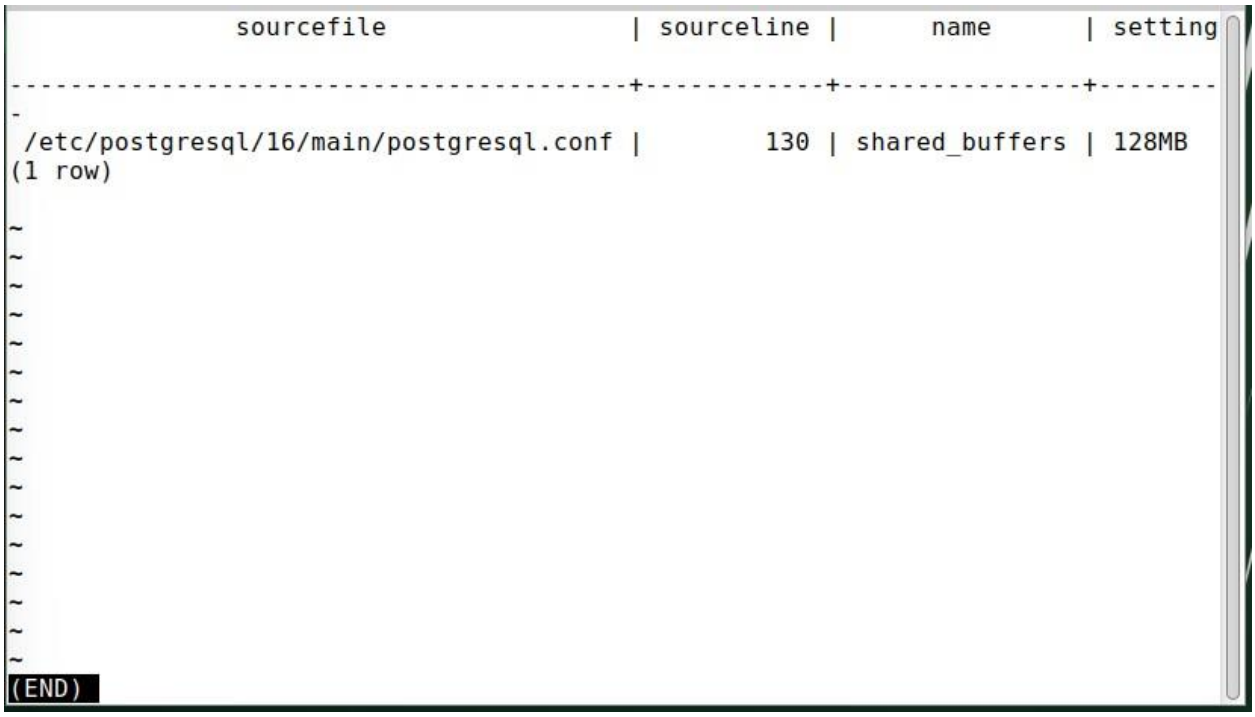


Рисунок 5 – Анализ файлов pg_file_settings

Часть 2: Управление параметрами на уровне экземпляра

1. Изменение через ALTER SYSTEM: Используя команду ALTER SYSTEM, установите для параметра work_mem новое значение. Убедитесь, что изменение записалось в файл postgresql.auto.conf (используйте функцию pg_read_file). Примените изменение, перечитав конфигурацию (SELECT pg_reload_conf();). Проверьте новое значение параметра и его источник в pg_settings.

```
postgres=# ALTER SYSTEM SET work_mem = '64MB';
ALTER SYSTEM
postgres=# SELECT pg_read_file('postgresql.auto.conf');
                      pg_read_file
-----------------------------------------------------------
 # Do not edit this file manually!                        +
 # It will be overwritten by the ALTER SYSTEM command.+
 work_mem = '64MB'                                        +

(1 row)

postgres=#
```

Рисунок 6 – Изменение через ALTER SYSTEM и проверка записи

```
postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)

postgres=#
```

Рисунок 7 – Перечитывание конфигурации

```
postgres=# SELECT name, setting, source
FROM pg_settings
WHERE name = 'work_mem';
   name   | setting |       source
----------+---------+--------------------
 work_mem | 65536   | configuration file
(1 row)

postgres=#
```

Рисунок 8 – Проверка нового значения и его источника

2. Изменение через дополнительный файл: Создайте файл в каталоге, указанном в директиве include_dir основного конфигурационного файла. Установите в этом файле значение для параметра log_min_duration_statement. Примените изменение и проверьте его.

```
postgres=# SHOW config_file;
               config_file
-----------------------------------------
 /etc/postgresql/16/main/postgresql.conf
(1 row)

postgres=# SHOW include_dir;
ERROR:  unrecognized configuration parameter "include_dir"
postgres=#
```

Рисунок 9 – Проверка include_dir

```
student:~/postgresql-16.0$ echo "log_min_duration_statement = 5000" | sudo tee /
etc/postgresql/16/main/conf.d/custom.conf
log_min_duration_statement = 5000
student:~/postgresql-16.0$
```

Рисунок 10 – Создание файла и установка параметра

```
postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)

postgres=# SHOW log_min_duration_statement;
 log_min_duration_statement
----------------------------
 5s
(1 row)

postgres=#
```

Рисунок 11 – Перезагрузка и проверка конфигурации

3. Ошибка в конфигурации: Намеренно внесите синтаксическую ошибку в один из конфигурационных файлов (например, invalid_value вместо числового значения). Попытайтесь перечитать конфигурацию. Изучите представление pg_file_settings, чтобы найти запись об ошибке. Исправьте ошибку и перечитайте конфигурацию.

```
student:~/postgresql-16.0$ echo "work_mem = invalid_value" | sudo tee -a /etc/po
stgresql/16/main/conf.d/custom.conf
work_mem = invalid_value
```

Рисунок 12 – Добавление ошибки в файл

```
postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)

postgres=# SELECT sourcefile, sourceline, name, error
FROM pg_file_settings
WHERE error IS NOT NULL;
 sourcefile | sourceline | name | error
------------+------------+------+-------
(0 rows)

postgres=#
```

Рисунок 13 – Перечитывание конфигурации и проверка ошибки

```
postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)

postgres=# SHOW log_min_duration_statement;
 log_min_duration_statement
----------------------------
 5s
(1 row)

postgres=#
```

Рисунок 14 – Перечитывание и проверка исправного файла

Часть 3: Управление параметрами на уровне сеанса

1. Команда SET: В рамках сеанса измените значение параметра work_mem с помощью SET. Проверьте новое значение. Завершите транзакцию с помощью ROLLBACK и проверьте значение параметра again. Объясните результат.

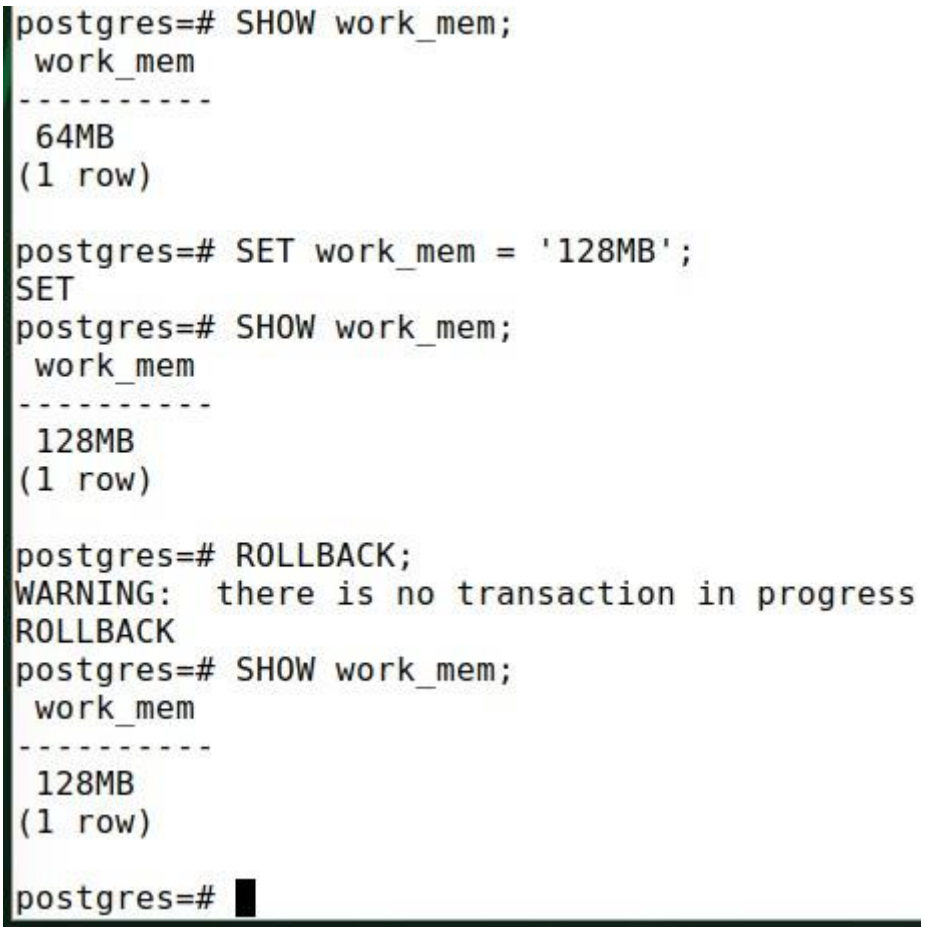


Рисунок 15 – Работа с параметром work_mem с помощью SET

2. Команда SET LOCAL: Откройте транзакцию (BEGIN). Inside the transaction, use SET LOCAL to change the work_mem parameter. Verify the change. After committing the transaction (COMMIT), check the parameter value again. Explain the result.

```
postgres=# BEGIN;
BEGIN
postgres=*# SET LOCAL work_mem = '256MB';
SET
postgres=*# SHOW work_mem;
 work_mem
----------
 256MB
(1 row)

postgres=*# COMMIT;
COMMIT
postgres=# SHOW work_mem;
 work_mem
----------
 128MB
(1 row)

postgres=#
```

Рисунок 16 – Использование SET LOCAL

3. Пользовательский параметр: Создайте и установите значение для пользовательского параметра (имя должно содержать точку, например, app.my_setting). Прочитайте его значение с помощью current_setting.

```
postgres=# SET app.my_setting = 'test_value';
SET
postgres=# SELECT current_setting('app.my_setting');
 current_setting
-----------------
 test_value
(1 row)

postgres=#
```

Рисунок 17 – Создание пользовательского параметра