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 №235644

Группа: P3324

Выполнил:

Круглов Егор Ильич

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/vtz5
* Кодировка: UTF8
* Локаль: английская
* Параметры инициализации задать через переменные окружения

## 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: 9644
* Способ аутентификации TCP/IP клиентов: по паролю SHA-256
* 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 со сценарием OLAP: 5 одновременных пользователей, пакетная запись/чтение данных по 64МБ.
* Директория WAL файлов: $PGDATA/pg\_wal
* Формат лог-файлов: .csv
* Уровень сообщений лога: ERROR
* Дополнительно логировать: завершение сессий и продолжительность выполнения команд

## Этап 3. Дополнительные табличные пространства и наполнение базы

* Пересоздать шаблон template1 в новом табличном пространстве: $HOME/iks88
* На основе template0 создать новую базу: lastredlake
* 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:

export PGDATA=$HOME/vtz5

export ENC=UTF8

export LOC=en\_US.UTF-8

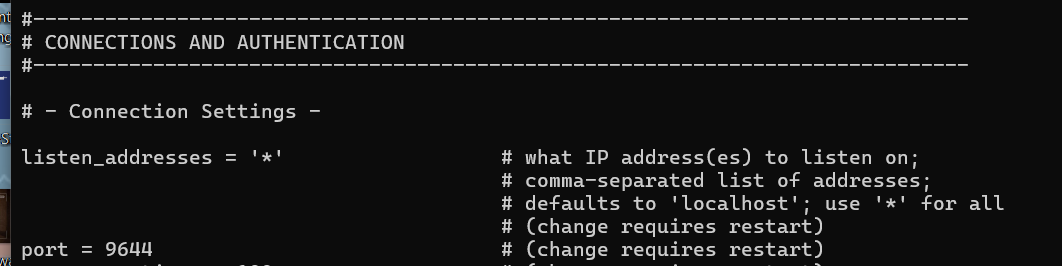
initdb -D $PGDATA --encoding=$ENC --locale=$LOC

## Этап 2. Конфигурация и запуск сервера БД

### Настройка способов подключения

Редактируем файл postgresql.conf:

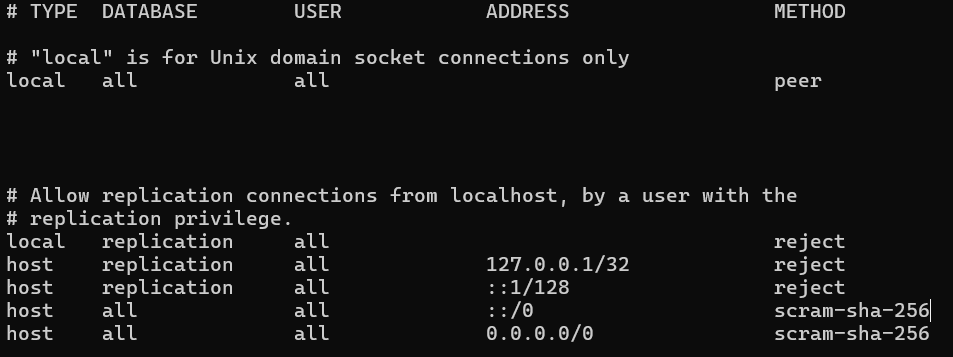
* сокет TCP/IP, принимать подключения к любому IP-адресу узла
* Номер порта: 9644





Редактируем файл pg\_hba.conf:

* Unix-domain сокет в режиме peer
* Способ аутентификации TCP/IP клиентов: по паролю SHA-256
* Остальные способы подключений запретить.



### Параметры сервера БД

Редактируем postgresql.conf:

**max\_connections**: Это максимальное количество одновременных подключений к серверу базы данных. Поскольку у нас 5 одновременных пользователей, можно установить значение чуть больше этого числа для учета системы обслуживания запросов, а также некоторых приложений и процессов, таких как фоновые задачи или администрирование.



**shared\_buffers**: Этот параметр определяет объем памяти, который PostgreSQL использует для кэширования данных, прежде чем они будут записаны на диск. Значение 1GB является разумным начальным значением для систем с ограниченным объемом оперативной памяти. Обычно рекомендуется выделять от 25% до 40% доступной оперативной памяти под shared\_buffers. Однако для тестовых окружений или систем с ограниченными ресурсами это значение может быть уменьшено.

****

**temp\_buffers**: Этот параметр указывает объем временной памяти, выделяемой для каждого сеанса (подключения) для временных таблиц и операций, таких как сортировка. Значение 16MB подходит для небольших операций и временных таблиц. Оно достаточно для большинства запросов без значительного увеличения использования памяти.



**work\_mem**: Этот параметр определяет объем памяти, выделяемой для выполнения операций, таких как сортировка и хеширование. Значение 64MB позволяет выполнять достаточно большие операции без необходимости сбрасывать данные на диск, что улучшает производительность. Это значение подходит для OLAP, где сортировки и агрегации часто требуются.



**checkpoint\_timeout**: Этот параметр указывает интервал времени между контрольными точками, когда PostgreSQL записывает изменения на диск. Увеличение этого значения до 15 минут позволяет уменьшить частоту записи на диск, что может улучшить производительность в сценариях, где производительность записи важнее, чем немедленная согласованность. Это особенно актуально для OLAP, где чтение данных важнее.



**effective\_cache\_size**: Этот параметр указывает PostgreSQL, сколько памяти доступно для кэширования данных в файловой системе, включая кэш ОС и кэш PostgreSQL. Значение 3GB является предположительным значением для систем с ограниченной памятью (75% от оперативной памяти). Указание более высокой величины помогает PostgreSQL оптимизировать планы выполнения запросов, принимая во внимание, что большая часть данных может быть кэширована.

****

**fsync**: Этот параметр управляет тем, нужно ли выполнять синхронизацию всех данных на диск после каждой транзакции. Установка fsync в значение on обеспечивает согласованность данных и защиту от потери данных в случае сбоя системы. Это критически важно для производственных систем, где потеря данных недопустима.



**commit\_delay**: Этот параметр определяет задержку перед подтверждением транзакции, что позволяет пакетировать несколько транзакций вместе, увеличивая производительность записи. Значение 500 мс (миллисекунд) может быть оптимальным для сценариев OLAP, где записи производятся большими партиями, минимизируя накладные расходы на операции подтверждения.



### WAL файлы и логирование:

**Директория WAL файлов:**

Настроим директорию для WAL файлов:

* archive\_mode - включает архивирование WAL файлов.



Нам не нужно отдельно настраивать директорию, так как она у нас дефолтная: $PGDATA\pg\_wal

**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.









**Уровень сообщений лога**:

* log\_min\_messages - минимальный уровень сообщений, которые будут записаны в лог. В данном случае, только ошибки и выше.



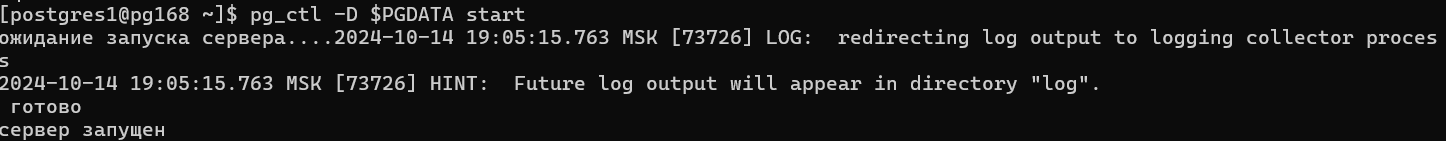
**Дополнительно логировать**:

* log\_disconnections - логировать отключения. Оба параметра используем для отслеживания завершения сессий.
* log\_duration - логировать продолжительность выполнения команд.

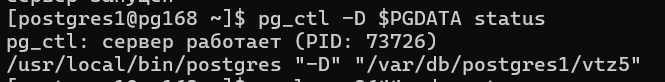


### Запуск сервера БД

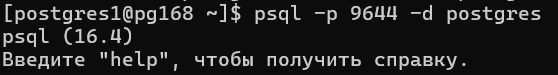
Запускаем сервер:

Проверка всех параметров

Статус сервера:



Подключение локально:

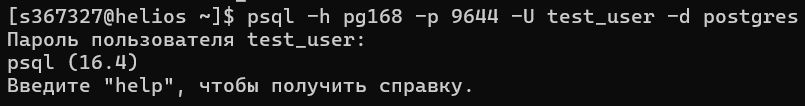


Подключение удаленно:

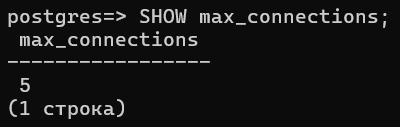
Создадим нового пользователя PostgreSQL с паролем:

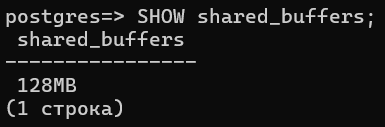


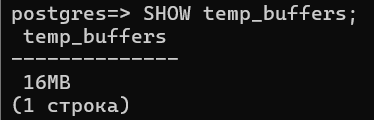
Попробуем подключиться удаленно:

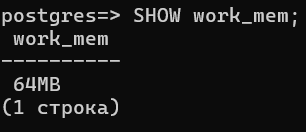


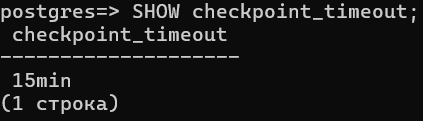
Проверка параметров:

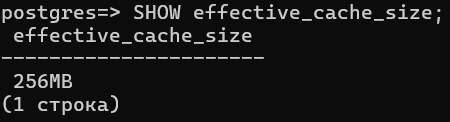


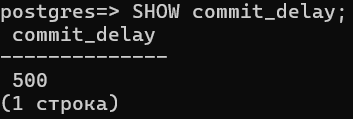
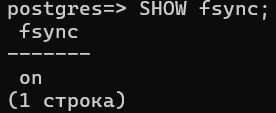






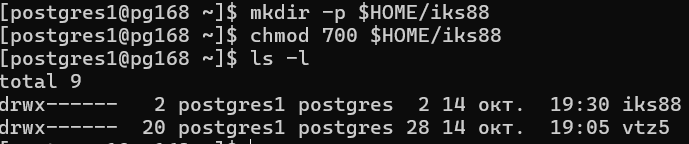






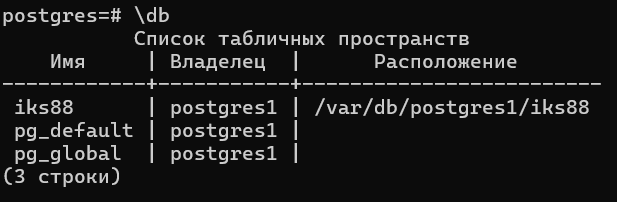
## Этап 3. Дополнительные табличные пространства и наполнение базы

### Создание табличных пространств



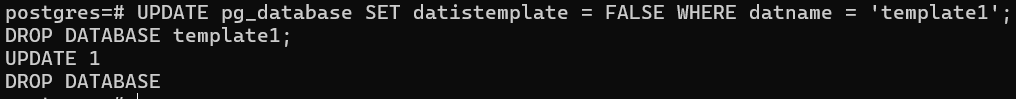


Проверка:



### Пересоздание шаблона template1

Так как template1 является системной базой данных, PostgreSQL требует разрешения на её удаление. Чтобы обойти это ограничение, сначала выполните следующие команды:



Создадим template1 заново, используя в качестве шаблона template0:



### Создание базы данных

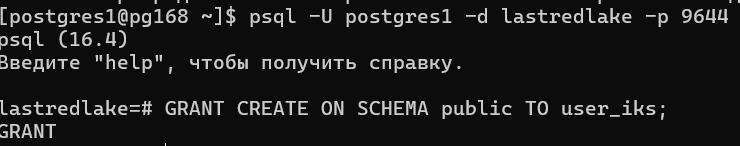


### Создание роли





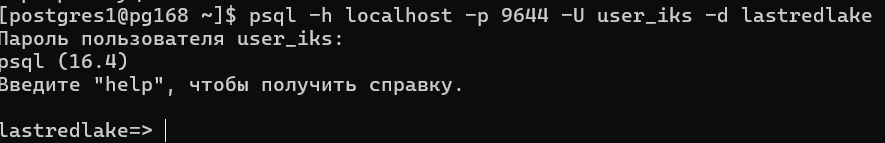




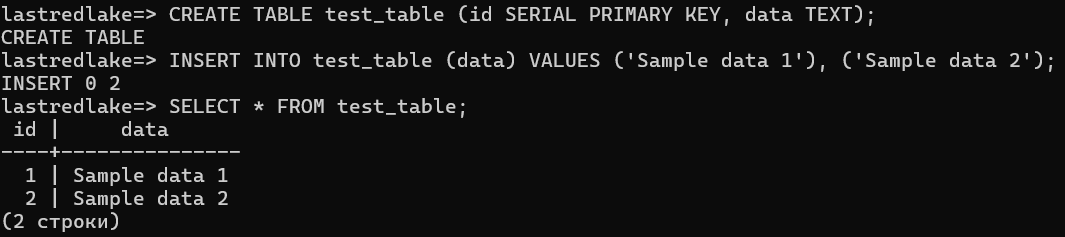


### Наполнение созданных баз тестовыми наборами данных.

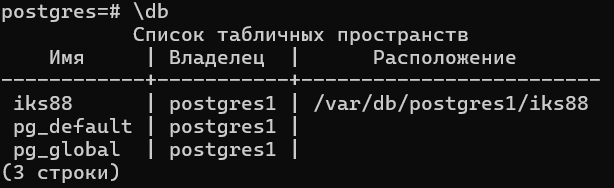
Подключаемся под именем нового пользователя:

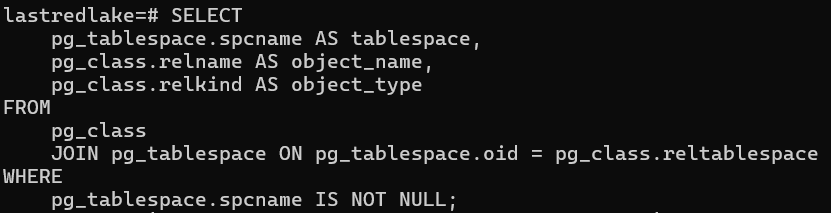


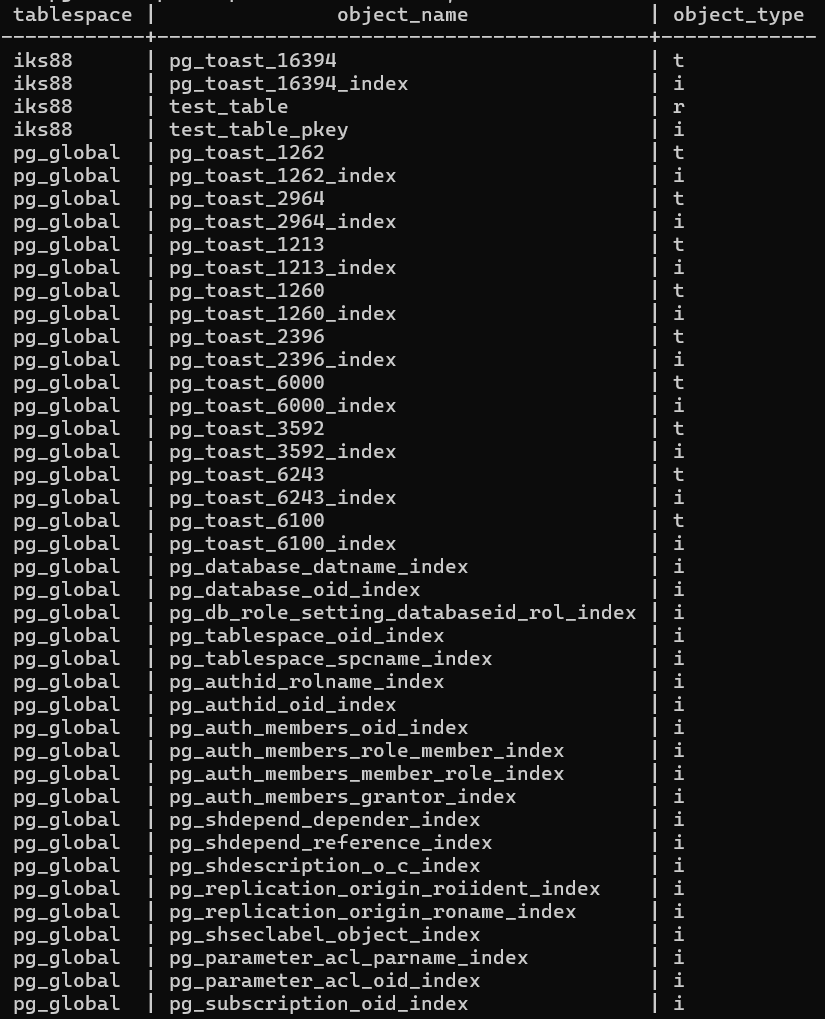
Создадим тестовую таблицу и заполним ее данными:

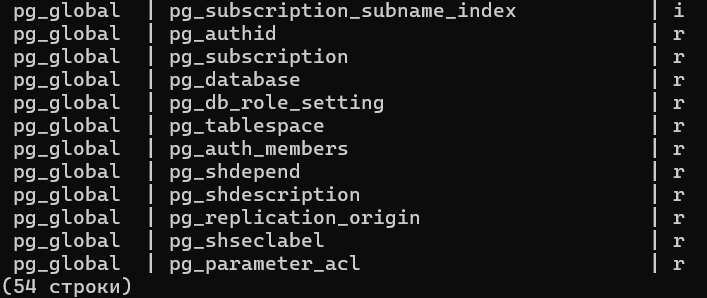


### Вывести список всех табличных пространств кластера и содержащиеся в них объекты









# Вывод

В ходе выполнения лабораторной работы был создан и сконфигурирован новый кластер базы данных PostgreSQL на выделенном узле. Это задание позволило углубить знания и приобрести практические навыки в настройке и управлении кластером базы данных PostgreSQL.