**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ПП.09.02.07-1.24.201.17

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

ПМ.07 Соадминистрирование баз данных и серверов

|  |  |  |
| --- | --- | --- |
| Руководитель от предприятия: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (В.А. Фирсов) |
| М.П. |  |  |
| Руководитель от техникума: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (С.Н. Касьяненко) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Ю.В. Свинина) |

Иркутск 2024

Содержание

[ВВЕДЕНИЕ 3](#_Toc164479614)

[1 Создание пользователей в PostgreSQL 4](#_Toc164479615)

[2 Мониторинг для запросов PostgreSQL 7](#_Toc164479616)

[2.1 Репликация на основе Postgresql Master-Slave 8](#_Toc164479617)

[3 База данных для учета посещения студентами учебного заведения 10](#_Toc164479618)

[4 Настройка PgBouncer 12](#_Toc164479619)

[5 Проверка портов с помощью NMAP 15](#_Toc164479620)

[ЗАКЛЮЧЕНИЕ 16](#_Toc164479621)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 17](#_Toc164479622)

# ВВЕДЕНИЕ

В настоящее время производственная практика является неотъемлемой частью образовательного процесса студентов различных специальностей. Она позволяет студентам приобретать необходимые навыки и опыт работы в своей будущей профессии.

Профессиональный модуль ПМ.07 "Соадминистрирование баз данных и серверов" является неотъемлемой частью подготовки специалистов в области администрирования баз данных.

В целом, профессиональный модуль является основополагающим для подготовки высококвалифицированных специалистов в области администрирования баз данных, которые могут успешно решать задачи по разработке и интеграции сложных программных систем.

# Создание пользователей в PostgreSQL

С помощью pgAdmin необходимо подключится к базе данных и в разделе Login/Group Roles выбрать Create — Login/Group Role.

На рисунке 1 изображено контекстное меню.

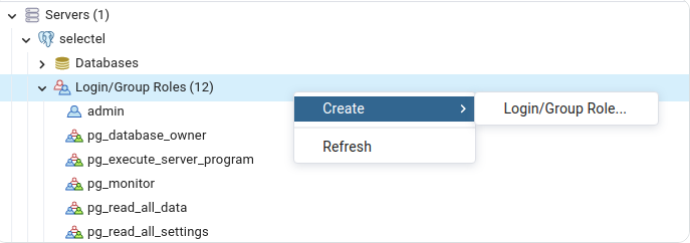


Рисунок 1 – Контекстное меню

Появится диалоговое окно с несколькими вкладками. На первой вкладке указывается имя пользователя.

На рисунке 2 изображено вкладка General.

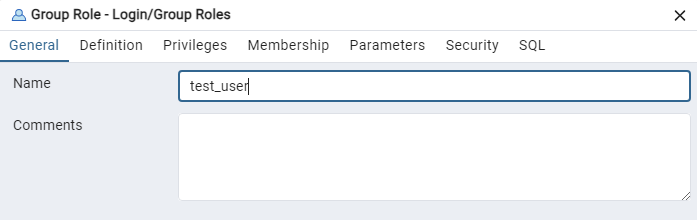


Рисунок 2 – Имя пользователя

На вкладке Definition задаётся пароль пользователя. На рисунке 3 изображена вкладка Definition.

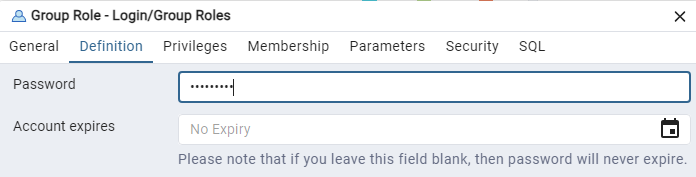


Рисунок 3 – Вкладка Definition

Вкладка Privileges нужна для выдачи привилегий пользователям таким как создание ролей, баз данных, объявление суперпользователя, возможность входить в систему и т.д.

На рисунке 4 изображена вкладка Privileges.

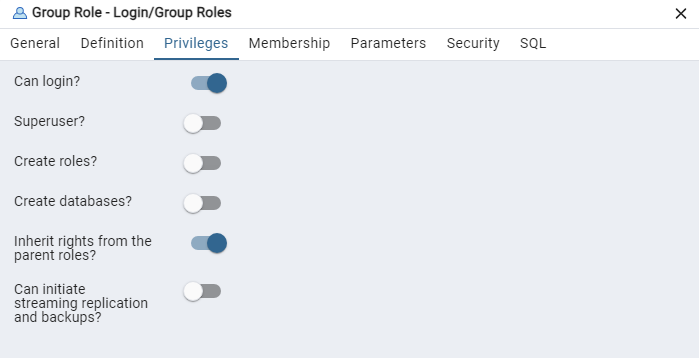


Рисунок 4 – Выдача привелегий

На вкладке Membership определяются роли пользователей (см. рисунок 5).

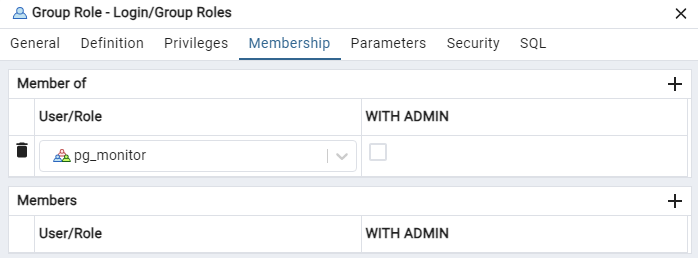


Рисунок 5 – Роли пользователей

На вкладке SQL можно увидеть SQL-команды, с помощью которых будет создан пользователь (рисунок 6).

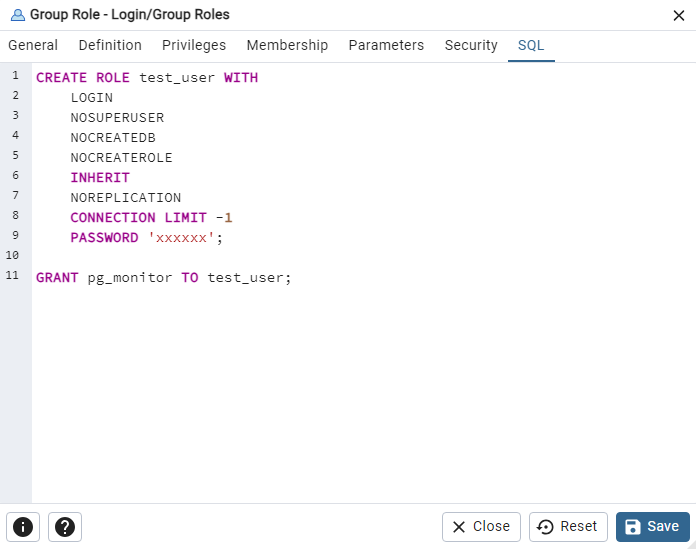


Рисунок 6 – Вкладка SQL

После нажатия на кнопку Save, и пользователь создался. и стал отображаться в разделе Login/Group Roles:

На рисунке 7 изображен добавленный пользователь.

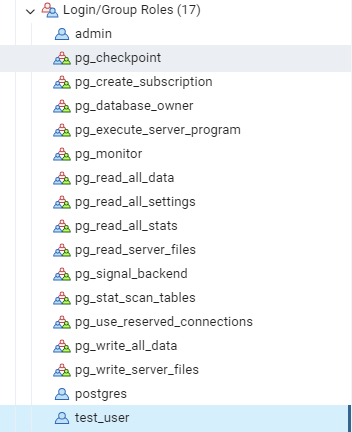


Рисунок 7 – Добавленный пользователь

Техническое задание на создание БД, размещенном на репозитории GitHub https://github.com/yuliaSvinina/PM07.

# Мониторинг для запросов PostgreSQL

Для настройки мониторинга нужно настроить расширение pg\_stat\_statements для PostgreSQL, нужно изменить postgresql.conf по расположению C:\Program Files\PostgreSQL\16\data.

На рисунке 8 изображен файл postgesql.conf.

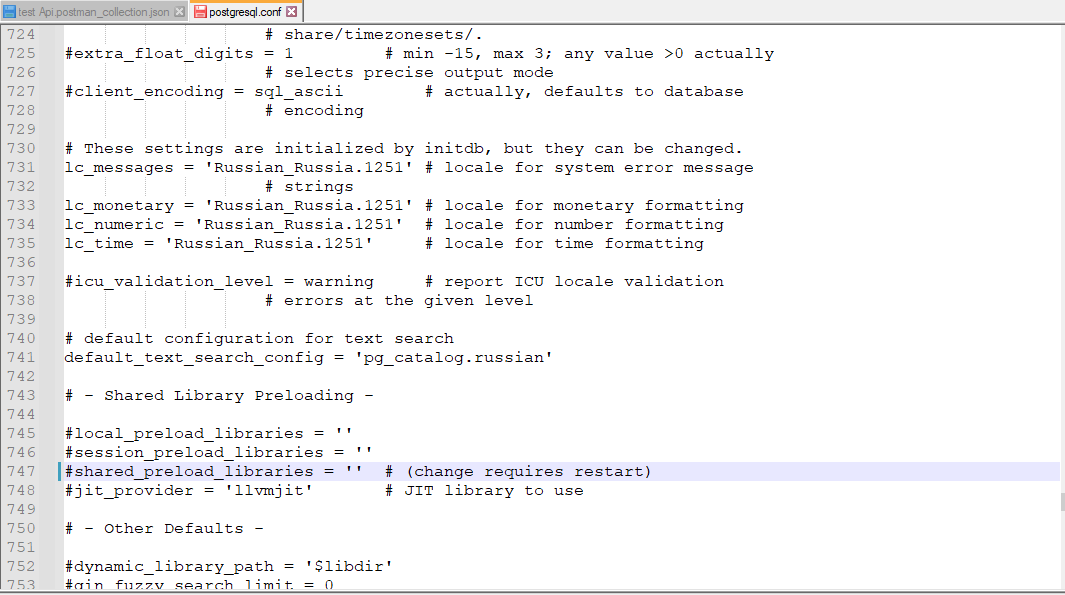


Рисунок 8 – Файл postgesql.conf

Необходимо изменить 747 строку на shared\_preload\_libraries = ‘pg\_stat\_statements’.

На рисунке 9 изображен измененный файл postgresql.conf.

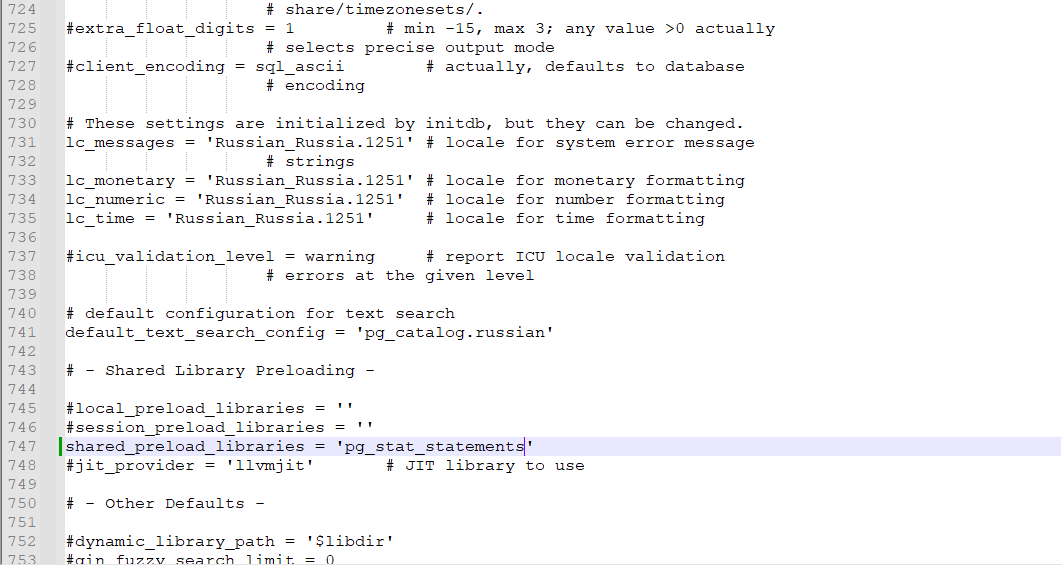


Рисунок 9 – Измененный файл postgresql.conf

Далее необходимо написать запрос на мониторинг в PostgreSQL, запрос будет выглядеть так SELECT \* FROM pg\_stat\_statements.

На рисунке 10 изброжен выполнение запроса.

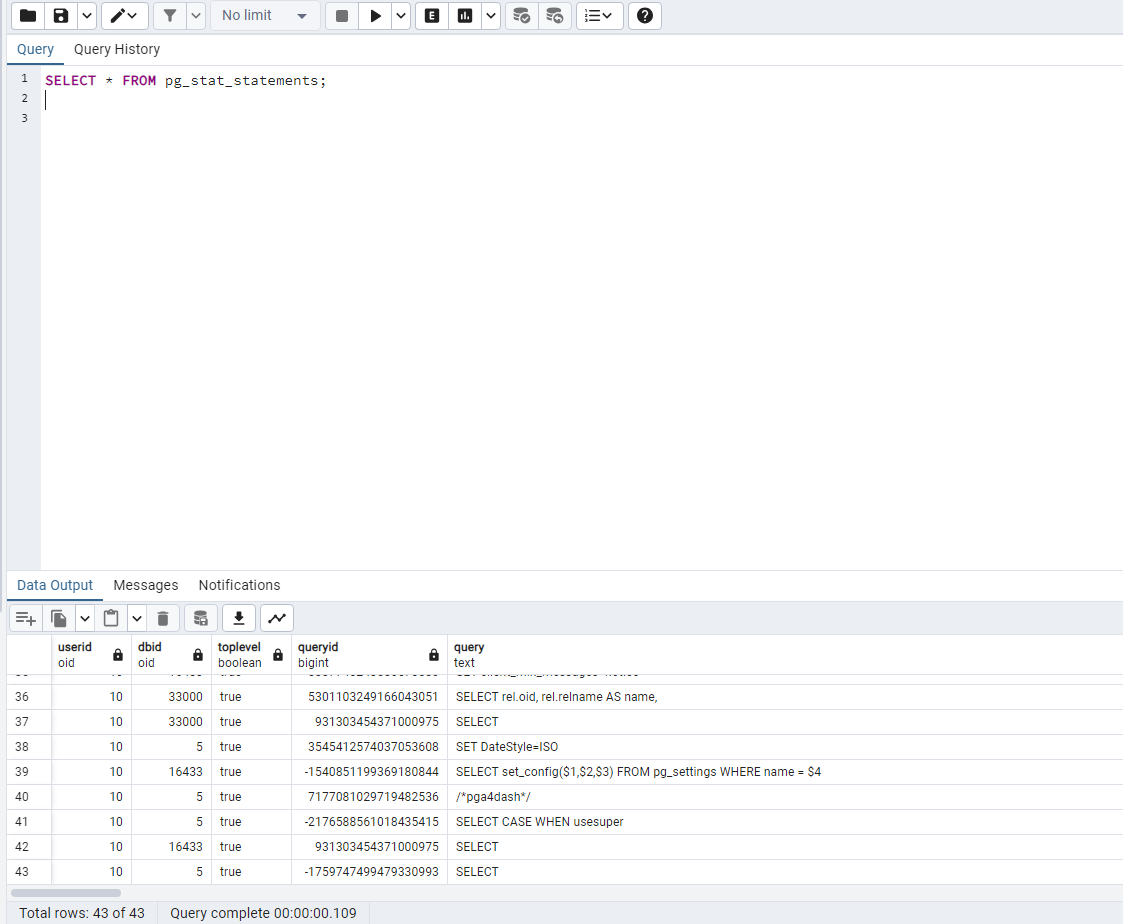


Рисунок 10 – Выполнение запроса

## Репликация на основе Postgresql Master-Slave

Репликация Master-Slave — это широко распространенный и важный архитектурный шаблон, обычно используемый для достижения высокой доступности, масштабируемости и избыточности в различных программных системах. В частности, это относится к процессу поддержания согласованности данных на нескольких узлах в распределенной среде, такой как базы данных, файловые системы или любое приложение, требующее синхронизации данных.

На рисунке 11 изображена установка postgresql.



Рисунок 11 – Установка postgresql в Ubuntu

Далее необходимо работать в консоли сервера Master. Под аккаунтом postgres необходимо создать пользователя для репликации.

На рисунке 12 изображено создание пользователя для репликации.

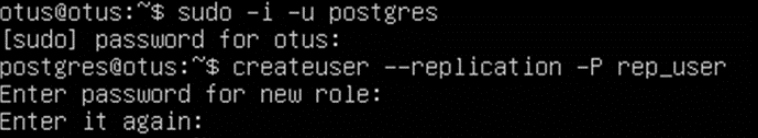


Рисунок 12 – Создание пользователя для репликации

После манипуляций необходимо перезапустить postgresql.

На этом настройка сервера Master завершена. Теперь перейдем к настройке сервера Slave. Необходимо исправить файл postgresql.conf: listen\_addresses = 'localhost, 192.168.1.136'.

Для внесения дальнейших изменений необходимо остановить сервер:

На рисунке 13 изображена остановка сервера postgresql.

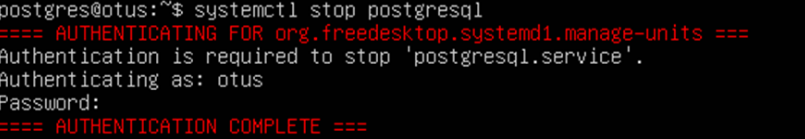


Рисунок 13 – Остановка сервера postgresql

Так как в режиме подчиненного сервера все данные реплицируются с основного, нам необходимо удалить файлы из каталога main.

Теперь необходимо провести проверку работы процесса репликации. Для этого используются команда pg\_basebackup с адресом основного сервера и именем пользователя для репликаций.

На рисунке 14 изображена проверка работы репликации.



Рисунок 14– Проверка работы репликации

Для настройки конфигурационного файла оптимизации работы при различных аппаратных ресурсах необходимо изменить файл конфигурации postgresql.conf.

# База данных для учета посещения студентами учебного заведения

Для создания базы данных были определены 3 сущности: Студенты, Занятие, Посещение занятий, а также атрибуты данных сущностей будут описаны ниже в запросах.

На рисунке 15 изображен запрос на создание базы данных.

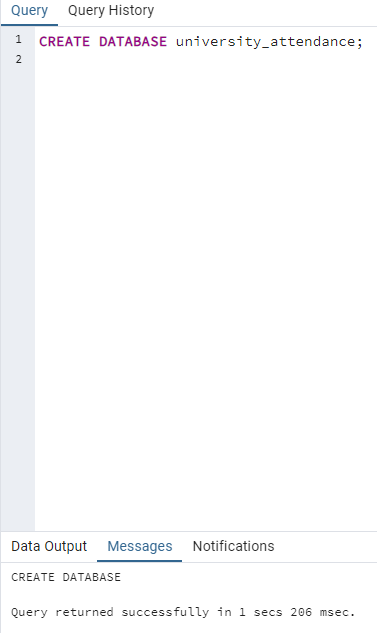


Рисунок 15 – Запрос на создание базы данных

На рисунке 16 изображен запрос на создание таблицы студенты.

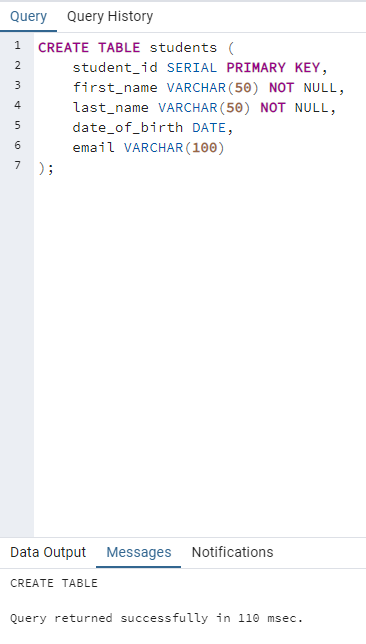


Рисунок 16 – Запрос на создание таблицы студенты

На рисунке 17 изображен запрос на создание таблицы посещение занятий.

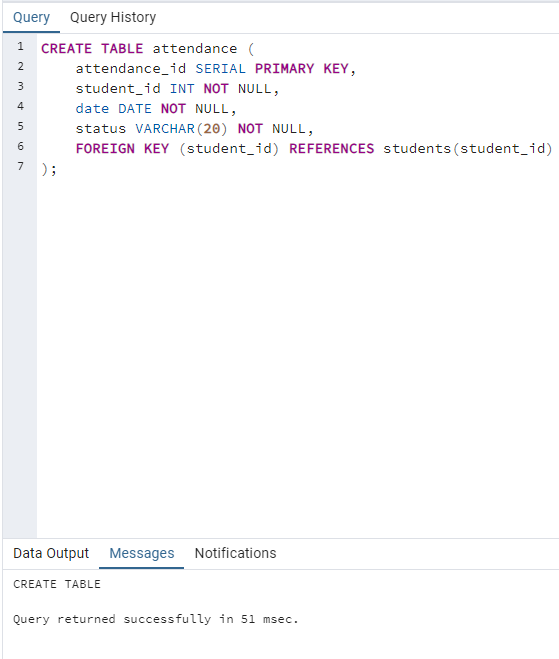


Рисунок 17 – Запрос на создание таблицы посещение занятий

На рисунке 18 изображен запрос на создание таблицы занятия.

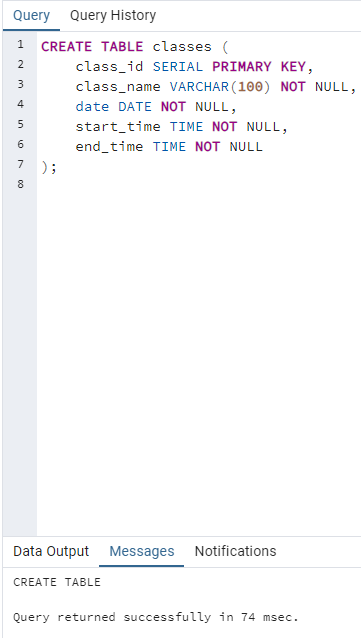


Рисунок 18 – Запрос на создание таблицы занятия

# Настройка PgBouncer

Необходимо установить PgBouncer из стандартных репозиториев Ubuntu.

На рисунке 19 изображена установка PgBouncer.

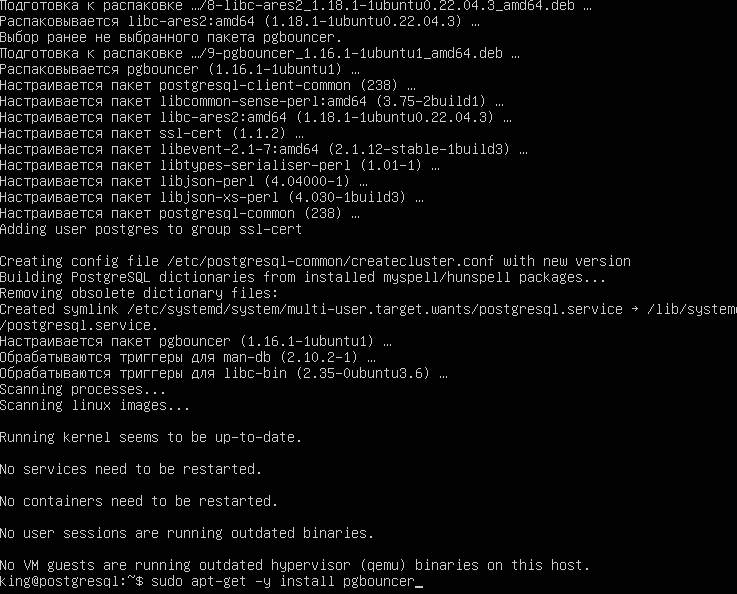


Рисунок 19 – Установка PgBouncer

После завершения установки отредактируем файл /etc/pgbouncer/pgbouncer.ini. Укажем, на каком хосту и порту PgBouncer должен искать запущенный сервер PostgreSQL. Для этого в раздел [databases] добавим.

На рисунке 20 изображен измененный файл pgbouncer.ini.

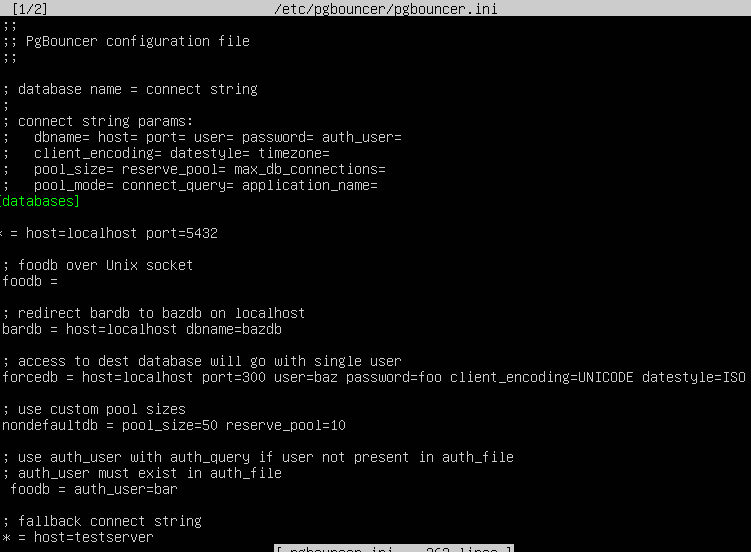


Рисунок 20 – Изменённый файл pgbouncer.ini

На рисунке 21 итоговый файл конфигурации.

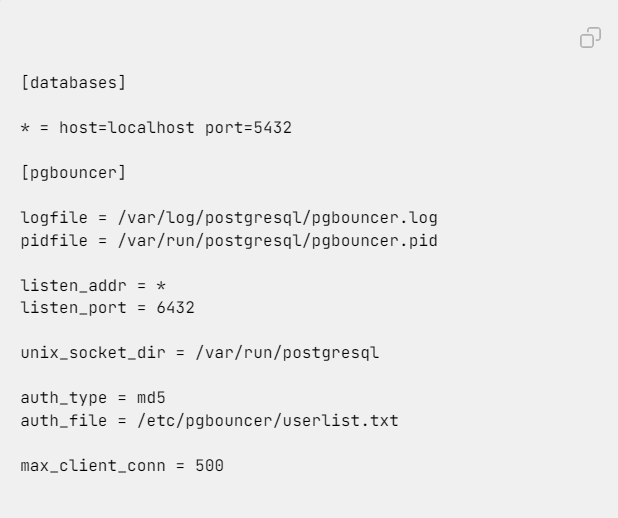


Рисунок 21 – Итоговый файл конфигурации

Необходимо создать пользователей для проверки шифровании и для подключения к пулу PgBouncer.

На рисунке 22 изображен созданный пользователь в userlist.txt.

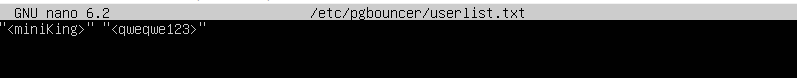


Рисунок 22 – Созданный пользователь

Нужно зашифровать пароль для это пропишите команду echo –n ‘<пароль созданного пользователя>’ | md5sum.

На рисунке 23 изображен зашифрованный пароль.



Рисунок 23 – Зашифрованный пароль

Проверяем с помощью утилиты pgbench. Сначала нужно подготовить БД к бенчмарку, для этого запустим команду.

На рисунке 24 изображена подготовка БД к бенчмарку.



Рисунок 24 – Подготовка БД к бенчмарку

Запуск по очереди двух тестов: на стандартном порту 5432, и на порту 6432 с пулером. Сымитируем запросы 50 клиентов в 16 потоков, тест будет продолжаться 60 секунд.

На рисунке 25 изображены результаты бенчмарка.

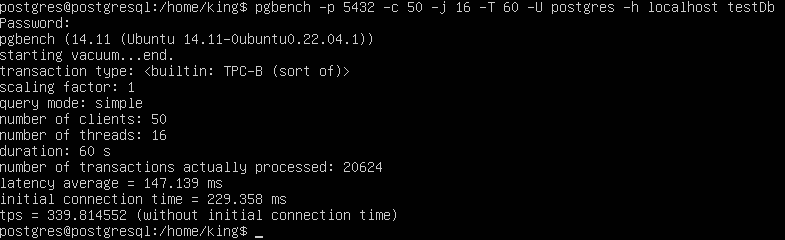


Рисунок 25 – Результаты бенчмарка

# Проверка портов с помощью NMAP

NMAP – это свободная утилита, предназначенная для разнообразного настраиваемого сканирования IP-сетей с любым количеством объектов, а также для определения состояния объектов сканируемой сети (портов и соответствующих им служб).

На рисунке 26 изображён процесс установки NMAP на Ubuntu Server

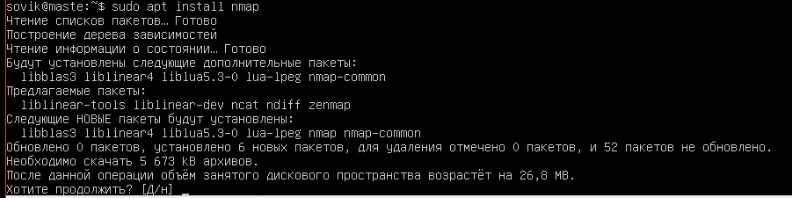


Рисунок 26 – Установка NMPA на Ubuntu Server

# ЗАКЛЮЧЕНИЕ

В результате производственной практики по модулю ПМ.07 "Соадминистрирование баз данных и серверов" были:

* развернуты и настроены сервера Ubuntu,
* настроена PostgreSQL,
* создана база данных,
* установлен PgBouncer,
* проведено сканирование портов с помощью NMAP.

Эти навыки и знания, полученные в процессе работы, будут полезны для будущей профессиональной деятельности.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Habr.com – Мониторинг Postgresql: запросы / Хабр – URL: https://habr.com/ru/companies/okmeter/articles/311028/ (дата обращения 15.04.2024). – Текст: электронный.
2. Habr.com – Отказоустойчивый кластер Master-Slave на PostgreSQL –URL: https://habr.com/ru/articles/188096/ (дата обращения 16.04.2024). – Текст: электронный.
3. selectel.ru – Установка пулера соединений PgBouncer для PostgreSQL – URL: https://selectel.ru/blog/tutorials/how-to-install-pgbouncer-connection-pooler-for-postgresql/?ysclid=lv4wuk22d2634508927 (дата обращения: 16.04.2024). – Текст: электронный.
4. Learnubuntu.com – Install nmap on Ubuntu – URL: <https://learnubuntu.com/install-nmap/> (дата обращения 17.04.2024) – Текст: электронный.
5. Habr.com – Nmap: сканирование портов – URL: <https://habr.com/ru/articles/767590/> (дата обращения 17.04.2924) – Текст: электронный.
6. Translated.turbopages.org – Как проверить (сканировать) наличие открытых портов в Linux – URL: <https://translated.turbopages.org/proxy_u/en-ru.ru.5e5ace12-6621c625-0f307623-74722d776562/https/linuxize.com/post/check-open-ports-linux/?__ya_mt_enable_static_translations=1> (дата обращения: 18.04.2024) – Текст: электронный.