**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ПП.09.02.07-1.24.202.07

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

ПМ.07 Соадминистрирование баз данных и серверов

|  |  |  |
| --- | --- | --- |
| Руководитель от предприятия:  М.П. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.А. Кличенко) |
| Руководитель от техникума: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (С.Н. Касьяненко) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.С. Замарацкий) |

Иркутск 2024

Содержание

[ВВЕДЕНИЕ 3](#_Toc164479518)

[1. Техническое задание на распределение доступа к БД 4](#_Toc164479519)

[2. Настройка мониторинга для запросов 6](#_Toc164479520)

[3. Настройка конфигурационного файла 8](#_Toc164479521)

[4. Создание базы данных 10](#_Toc164479522)

[5. Настройка PgBouncer 11](#_Toc164479523)

[6. Проверка открытых портов 13](#_Toc164479524)

[ЗАКЛЮЧЕНИЕ 14](#_Toc164479525)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 15](#_Toc164479526)

# ВВЕДЕНИЕ

Целью прохождения производственной практики в области информационных систем и программирования является практическое применение полученных в ходе обучения теоретических знаний, а также более глубокое понимание будущей работы и своих профессиональных обязанностей.

Цели практики:

* расширение и систематизация знаний на основе изучений деятельности;
* приобретение практических навыков в сфере информационных технологий;
* развитие навыков работы в команде и коммуникации с коллегами;
* укрепление теоретических знаний, полученный во время обучения.

Задачи практики:

* настройка политики безопасности;
* защита данных от несанкционированного доступа;
* мониторинг активности портов;
* администрирование базы данных;
* реализация защиты базы данных.

1. Техническое задание на распределение доступа к БД

**1 Наименование проекта и цель распределения доступа**

Наименование проекта: база данных «Киноклуб».

Цель: обеспечить безопасное и эффективное распределение доступа к таблице базы данных для пяти пользователей с разными уровнями доступа.

Исполнитель: студент Иркутского Авиационного Техникума, группы БД-20-2, Замарацкий Алексей Саидович.

Разработка базы данных проходит в рамках производственной практики для ГБПОУИО «ИАТ».

Сроки разработки с 05.04.2024 по 18.04.2024 года.

**2 Описание задачи**

Пользователи должны иметь возможность просматривать, добавлять, изменять и удалять данные в таблице базы данных в соответствии с определёнными правилами доступа.

**3 Требования к доступу**

Администратор (уровень 1): полный доступ ко всем данным и функциям таблицы.

Менеджер (уровень 2): доступ к просмотру и изменению данных, но без возможности добавления и удаления.

Специалист (уровень 3): доступ к просмотру и изменению данных, но без возможности добавления и удаления.

Эксперт (уровень 4): доступ к просмотру данных, но без возможности изменения или удаления.

Пользователь (уровень 5): только просмотр данных.

**4 Правила доступа**

Администратор имеет полный доступ ко всем данным и функциям таблицы.

Менеджер может просматривать и изменять данные, но не добавлять и не удалять их.

Специалист может просматривать и изменять данные, но не добавлять и не удалять их.

Эксперт может просматривать данные, но не изменять или удалять их.

Пользователь может только просматривать данные.

**5 Безопасность**

Все пользователи должны пройти аутентификацию и авторизацию перед получением доступа к таблице.

Аутентификация должна осуществляться через логин и пароль, а авторизация — через определение уровня доступа пользователя.

Доступ должен быть ограничен по IP-адресу и времени сессии.

**6 Резервное копирование и восстановление данных**

Необходимо предусмотреть возможность резервного копирования данных таблицы и восстановления данных в случае сбоев или ошибок.

**7 Мониторинг и отчётность**

Система должна предоставлять информацию о количестве и характере доступа к таблице, а также о возможных проблемах и ошибках.

**8 Тестирование и отладка**

Перед внедрением системы необходимо провести тестирование на соответствие требованиям и стандартам, а также отладку всех функций и возможностей.

**9 Поддержка и обновление**

После внедрения системы необходимо обеспечить поддержку и обновление программного обеспечения, исправление ошибок и уязвимостей.

1. Настройка мониторинга для запросов

Параметр **logging\_collector** включает сборщик сообщений. Это фоновый процесс, который собирает отправленные в stderr сообщения и перенаправляет их в журнальные файлы. Такой подход зачастую более полезен чем запись в syslog, поскольку некоторые сообщения в syslog могут не попасть.

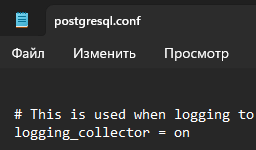


Рисунок 1 — logging\_collector

При включённом logging\_collector задаёт имена журнальных файлов. Значение **log\_filename** трактуется как строка формата в функции strftime, поэтому в ней можно использовать спецификаторы % для включения в имена файлов информации о дате и времени.

Если для задания имени файлов не используются спецификаторы %, то для избегания переполнения диска, следует использовать утилиты для ротации журнальных файлов.

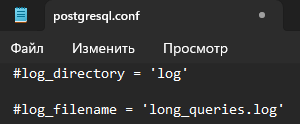


Рисунок 2 — log\_filename

В качестве значения **log\_destination**указывается один или несколько методов протоколирования, разделённых запятыми. По умолчанию используется stderr. Параметр можно задать только в конфигурационных файлах или в командной строке при запуске сервера.

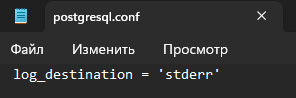


Рисунок 3 — log\_destination

**Log\_min\_duration\_statement** записывает в журнал продолжительность выполнения всех команд, время работы которых равно или превышает указанное количество миллисекунд. Значение 0 (ноль) заставляет записывать продолжительность работы всех команд. Значение -1 (по умолчанию) запрещает регистрировать продолжительность выполнения операторов. Например, при значении 250ms, все команды, которые выполняются за 250 миллисекунд и дольше будут записаны в журнал сервера.

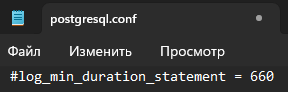


Рисунок 4 — log\_min\_duration\_statement

**Log\_statement** управляет тем, какие SQL-команды записывать в журнал. Допустимые значения: none (отключено), ddl, mod и all (все команды). Ddl записывает все команды определения данных, такие как CREATE, ALTER, DROP. Mod записывает все команды ddl, а также команды изменяющие данные, такие как INSERT, UPDATE, DELETE, TRUNCATE и COPY FROM. PREPARE, EXECUTE и EXPLAIN ANALYZE также записываются, если вызваны для команды соответствующего типа.

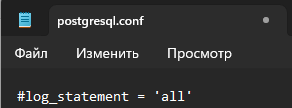


Рисунок 5 — log\_statement

Результат настройки мониторинга для запросов изображён на следующем рисунке.

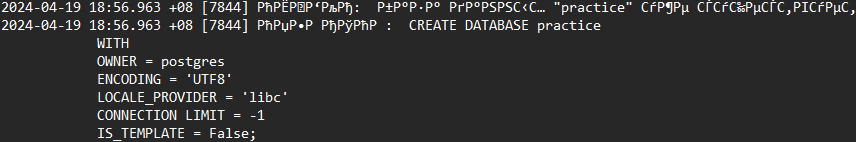


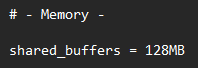
Рисунок 6 — результат запроса

1. Настройка конфигурационного файла

PostgreSQL использует свой собственный буфер, а также использует буферизованный IO ядра. Это означает, что данные хранятся в памяти дважды, сначала в буфере PostgreSQL, а затем в буфере ядра.

Буфер PostgreSQL называется shared\_buffer, который является наиболее эффективным настраиваемым параметром для большинства операционных систем. Этот параметр устанавливает, сколько выделенной памяти будет использоваться PostgreSQL для кеширования. Значение по умолчанию для shared\_buffer установлено очень низким, таким образом пользователь не получает большой выгоды от него.

Замечено, что в производственных средах большое значение для shared\_buffer действительно дает хорошую производительность, хотя для достижения правильного баланса всегда следует проводить тесты.

  
Рисунок 7 — shared\_buffer

Effective\_cache\_size предоставляет оценку памяти, доступной для кэширования диска. Это всего лишь ориентир, а не точный объем выделенной памяти или кэша. Он не выделяет фактическую память, но сообщает оптимизатору объем кэша, доступный в ядре. Если значение этого параметра установлено слишком низким, планировщик запросов может принять решение не использовать некоторые индексы, даже если они будут полезны. Поэтому установка большого значения всегда имеет смысл.



Рисунок 8 — effective\_cache\_size

Настройка work\_mem используется для сложной сортировки. Если необходимо выполнить сложную сортировку, то следует увеличить значение work\_mem для получения хороших результатов. Сортировка в памяти происходит намного быстрее, чем сортировка данных на диске. Установка очень высокого значения может стать причиной узкого места в памяти, поскольку этот параметр относится к операции сортировки пользователя. Поэтому, если много пользователей, пытаются выполнить операции сортировки, тогда система выделит для всех пользователей. Установка этого параметра глобально может привести к очень высокому использованию памяти. Поэтому настоятельно рекомендуется изменить его на уровне сеанса.



Рисунок 9 — work\_mem

Maintenance\_work\_mem — это параметр памяти, используемый для задач обслуживания. Значение по умолчанию составляет 64 МБ. Установка большого значения помогает в таких задачах, как VACUUM, RESTORE, CREATE INDEX, ADD FOREIGN KEY и ALTER TABLE.



Рисунок 10 — maintenance\_work\_mem

Synchronous\_commit используется для обеспечения того, что фиксация транзакции будет ожидать записи WAL на диск, прежде чем вернуть клиенту статус успешного завершения. Это компромисс между производительностью и надежностью. Если приложение разработано таким образом, что производительность важнее надежности, следует отключить synchronous\_commit. В этом случае транзакция фиксируется очень быстро, потому что она не будет ожидать сброса файла WAL, но надежность будет поставлена R03; R03; под угрозу. В случае сбоя сервера данные могут быть потеряны, даже если клиент получил сообщение об успешном завершении фиксации транзакции.



Рисунок 11 — synchronous\_commit

1. Создание базы данных

На рисунке ниже представлена схема базы данных для учёта посещения студентами учебного заведения. В созданной базе данных есть таблица “students”, которая содержит данные о студенте. Также две таблицы “groups” и “speciality”, отвечающие за название группы и специальности. Таблица “visit” содержит в себе id студента, дата входа и дата выхода.

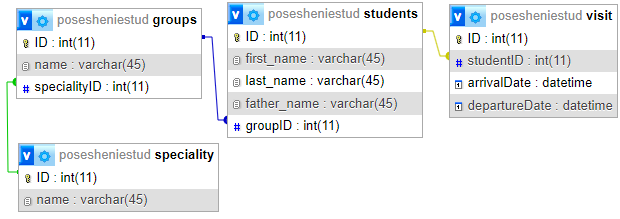


Рисунок 12 — модель БД

Далее на следующих рисунках представлен алгоритм добавления пользователя с выдачей ему необходимых прав для добавления и управления правами пользователей.

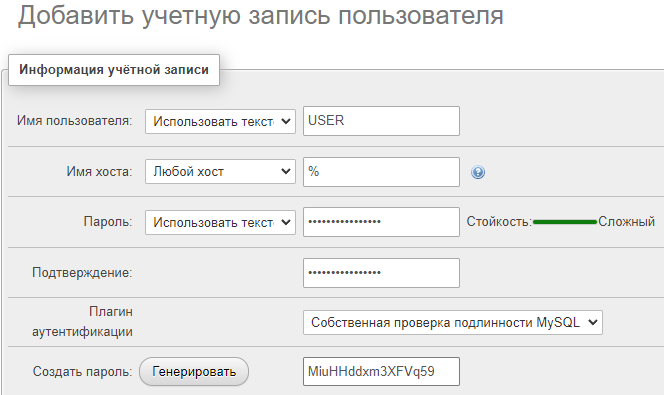


Рисунок 13 — добавление учётной записи

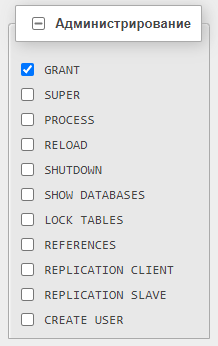


Рисунок 14 — добавление прав

1. Настройка PgBouncer

PgBouncer — это необязательная функция, которую можно включить на каждом сервере базы данных. Она поддерживается на уровнях вычислений общего назначения и оптимизированных для памяти вычислительных ресурсов как в общедоступных, так и в частных сетях доступа.

Ниже представлен рисунок, на котором происходит установка PgBouncer из репозиториев Ubuntu.

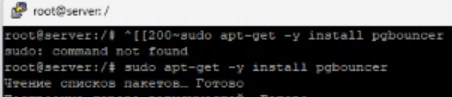


Рисунок 15 — установка PgBouncer

Также, ниже показан процесс редактирования файла, в котором указывается на каком хосту и порту PgBouncer должен искать запущенные сервер PostgreSQL.



Рисунок 16 — редактирование файла

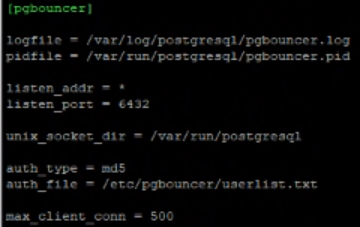


Рисунок 17 — итоговый файл

Ниже на рисунке запрос на создание пользователя с его паролем.



Рисунок 18 — создание пользователя

Далее представлен метод хэширования пароля пользователя с префиксом md5.



Рисунок 19 — хэширование пароля

Ниже показана настройка доступа по портам и ip-адресам к серверу БД.



Рисунок 20 — создание БД



Рисунок 21 — выдача привилегий пользователю к БД

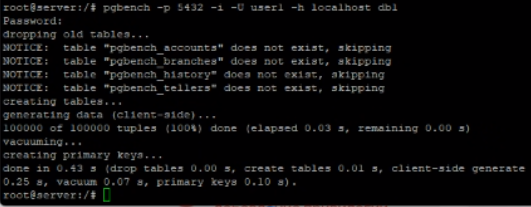


Рисунок 22 — подготовка базы данных

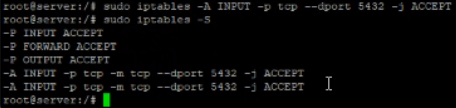


Рисунок 23 — результат

1. Проверка открытых портов

Nmap — свободная утилита, предназначенная для разнообразного настраиваемого сканирования IP-сетей с любым количеством объектов, определения состояния объектов сканируемой сети.



Рисунок 24 — установка Nmap

После установки сканируем порты nmap для localhost. Видно, что открыто несколько портов, которые используются каким-либо сервисом. Каждый из них может быть потенциально уязвимым, поэтому иметь много открытых портов на машине небезопасно.

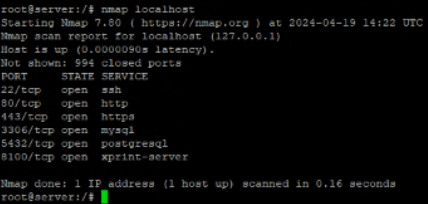


Рисунок 25 — сканирование портов

# ЗАКЛЮЧЕНИЕ

В результате, практика предоставляет множество возможностей для получения новых знаний и навыков, а также для развития профессиональных компетенций. Были закреплены теоретические знания, получены новые профессиональный навыки и умения. Также, были выполнены поставленные задачи и достигнуты цели.

В процессе практики была выполнена политика безопасности, также защита данных от несанкционированного доступа, что является крайне важным в повседневной жизни. Также, была пройдена организация безопасности баз данных, начиная с создания пользователей и выдачи им привилегий до взаимодействия с остальными учётными записями.

В целом, практика является важным этапом в карьере любого специалиста, поскольку она позволяет получить практический опыт и знания, необходимые для успешной работы в этой области.

Я успешно освоил и закрепил новые умения и навыки, применив теоретические знания на практике.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. infostart.ru – Настройка параметров PostgreSQL для оптимизации производительности – URL: https://infostart.ru/1c/articles/1089974/ (Дата обращения: 18.04.2024) Тест: электронный
2. prj-exp.ru – Техническое задание на распределение доступа к БД – URL: https://www.prj-exp.ru/patterns/pattern\_tech\_task.php (Дата обращения: 18.04.2024) Тест: электронный
3. selectel.ru – Как настроить репликацию в PostgreSQL – URL: https://selectel.ru/blog/tutorials/how-to-set-up-replication-in-postgresql/ (Дата обращения: 18.04.2024) Тест: электронный
4. selectel.ru – Установка пулера соединений PgBouncer для PostgreSQL – URL: https://selectel.ru/blog/tutorials/how-to-install-pgbouncer-connection-pooler-for-postgresql/ (Дата обращения: 18.04.2024) Тест: электронный