Отчет по лабораторной работе №6

Дисциплина: Администрирование сетевых подсистем

Иванов Сергей Владимирович

Содержание

# 1 Цель работы

Целью этой работы является приобретение практических навыков по установке и конфигурированию системы управления базами данных на примере программного обеспечения MariaDB.

# 2 Задание

1. Установите необходимые для работы MariaDB пакеты (см. раздел 6.4.1).
2. Настройте в качестве кодировки символов по умолчанию utf8 в базах данных.
3. В базе данных MariaDB создайте тестовую базу addressbook, содержащую таблицу city с полями name и city, т.е., например, для некоторого сотрудника указан город, в котором он работает (см. раздел 6.4.1).
4. Создайте резервную копию базы данных addressbook и восстановите из неё данные (см. раздел 6.4.1).
5. Напишите скрипт для Vagrant, фиксирующий действия по установке и настройке базы данных MariaDB во внутреннем окружении виртуальной машины server. Соответствующим образом следует внести изменения в Vagrantfile (см. раздел 6.4.5).

# 3 Выполнение лабораторной работы

## 3.1 Установка MariaDB

Загрузим операционную систему и перейдем в рабочий каталог с проектом: cd /var/tmp/user\_name/vagrant . Запустим виртуальную машину server: vagrant up server. (рис. 1).

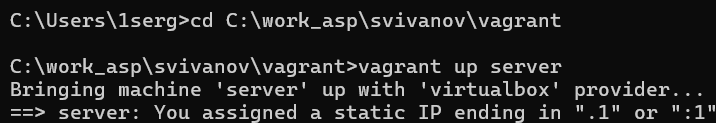


Рис. 1: Запуск server

На виртуальной машине server войдем под пользователем и откроем терминал. Перейдем в режим суперпользователя. Установим необходимые для работы с базами данных пакеты: dnf -y install mariadb mariadb-server (рис. 2).

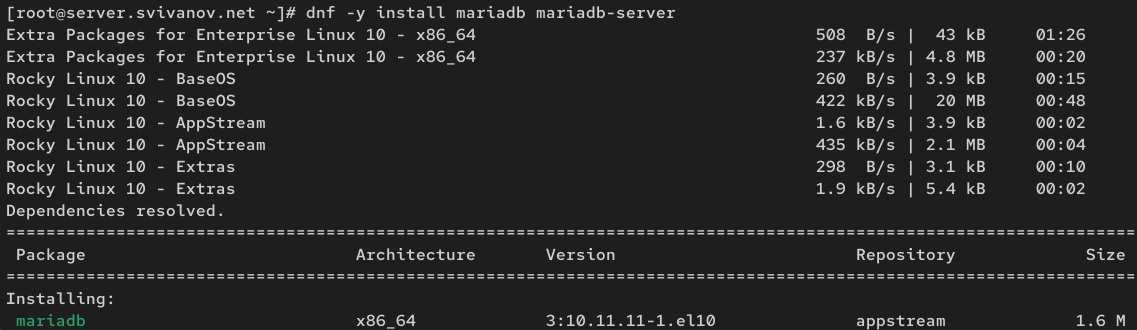


Рис. 2: Установка MariaDB

Просмотрим конфигурационные файлы mariadb в каталоге /etc/my.cnf.d и в файле /etc/my.cnf.

Файл auth\_gssapi.conf. В данном конфигурационном файле auth\_gssapi.conf директива, отвечающая за загрузку плагина аутентификации через Kerberos (auth\_gssapi.so), закомментирована. Это означает, что на данном сервере MariaDB механизм аутентификации GSSAPI отключен и не используется. Сервер будет работать со стандартными методами аутентификации (рис. 3)



Рис. 3: Файл auth\_gssapi.conf

Файл client.cnf. Данный конфигурационный файл является заготовкой для настройки клиентских подключений. В его текущем состоянии он не содержит активных параметров (таких как хост, пользователь, пароль и т.д.), а лишь определяет логическую структуру:

Секция client предназначена для универсальных настроек, которые применяются ко всем клиентским программам (как MySQL, так и MariaDB).

Секция client-mariadb предназначена для специфичных настроек, которые будут применены только к клиентским утилитам MariaDB, что обеспечивает гибкость и совместимость.

Файл подготовлен для добавления конкретных параметров подключения, но в настоящее время он пуст от активных директив. Его функция — организовать будущие настройки клиентской части СУБД. (рис. 4)

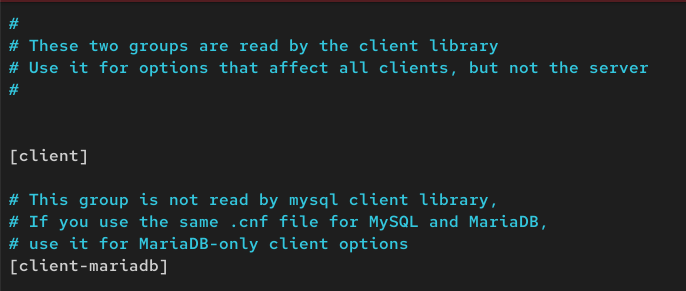


Рис. 4: Файл client.cnf

Файл mariadb-server.cnf. Конфигурационный файл содержит базовые, но критически важные настройки сервера MariaDB. Он определяет “рабочее пространство” сервера: Расположение данных (datadir), Точку для локальных подключений (socket), Файл для диагностики ошибок (log-error), Файл идентификации процесса (pid-file).

Файл задает стандартную конфигурацию, которая обеспечивает базовую функциональность сервера, определяя правильные и общепринятые пути для его ключевых компонентов. (рис. 5)



Рис. 5: Файл mariadb-server.cnf

Файл mysql-clients.cnf. Конфигурационный файл представляет собой структурированную заготовку для тонкой настройки клиентских утилит MariaDB. В его текущем состоянии он не содержит активных параметров, а лишь определяет логические секции для различных программ.

Файл демонстрирует модульный подход к конфигурации MariaDB, где разные аспекты системы настраиваются в отдельных специализированных файлах. В данном случае этот файл отвечает исключительно за настройку поведения клиентских утилит командной строки. (рис. 6)

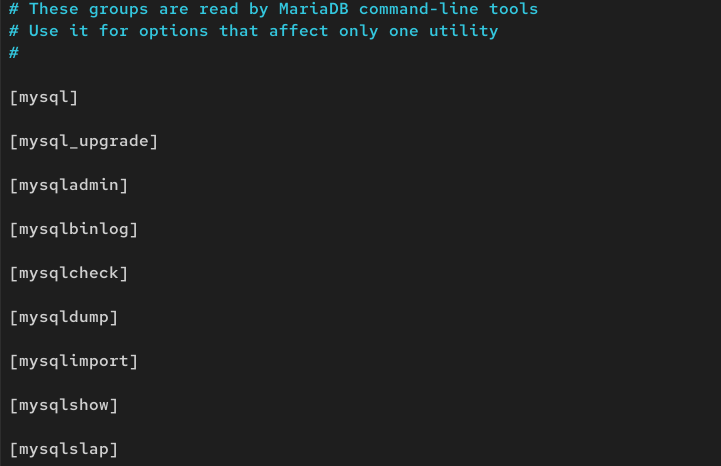


Рис. 6: Файл mysql-clients.cnf

Файл provider\_bzip2.cnf. Конфигурационный файл отвечает за подключение и настройку плагина сжатия bzip2 в сервере MariaDB.

Загрузка плагина: Директива plugin\_load\_add гарантирует, что плагин provider\_bzip2 будет загружен при каждом запуске сервера.

Параметр force\_plus\_permanent обеспечивает отказоустойчивость (сервер не “упадет” из-за отсутствия плагина) и сохраняет настройку постоянной.

После загрузки этого плагина MariaDB получает возможность использовать алгоритм bzip2 для операций сжатия. Это может быть полезно в различных сценариях, например: Сжатие данных при использовании функций типа COMPRESS(), Настройка сжатия для механизма хранения InnoDB (если плагин это поддерживает). (рис. 7)

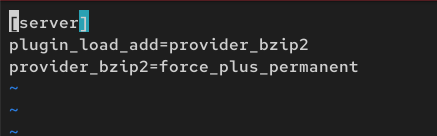


Рис. 7: Файл provider\_bzip2.cnf

Файл provider\_lz4.cnf. Конфигурационный файл предназначен для подключения и настройки плагина сжатия LZ4 в сервере MariaDB.

Загрузка плагина: Директива plugin\_load\_add должна гарантировать, что плагин provider\_lz4 будет загружен при каждом запуске сервера.

Параметр force\_plus\_permanent обеспечивает отказоустойчивость (сервер не “упадет” из-за отсутствия плагина) и сохраняет настройку постоянной.

Практическое применение: После загрузки этого плагина MariaDB получает возможность использовать алгоритм LZ4 для операций сжатия. LZ4 известен своим очень высоким скоростью сжатия и распаковки, что делает его особенно полезным для: Сжатия данных в реальном времени, в приложениях, где важна низкая задержка (low latency, использования в функциях сжатия MariaDB и различных механизмах хранения (рис. 8)

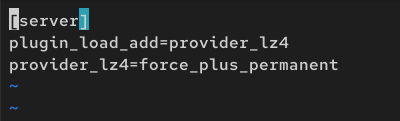


Рис. 8: Файл provider\_lz4.cnf

Файл provider\_lzo.cnf. Конфигурационный файл отвечает за подключение и настройку плагина сжатия LZO в сервере MariaDB.

Загрузка плагина: Директива plugin\_load\_add гарантирует, что плагин provider\_lzo будет загружен при каждом запуске сервера.

Параметр force\_plus\_permanent обеспечивает отказоустойчивость (сервер не “упадет” из-за отсутствия плагина) и сохраняет настройку постоянной.

Практическое применение: После загрузки этого плагина MariaDB получает возможность использовать алгоритм LZO для операций сжатия. LZO — это алгоритм сжатия, который обеспечивает хороший баланс между скоростью и степенью сжатия. Он может использоваться для: Сжатия данных в функциях типа COMPRESS(), настройки сжатия для различных механизмов хранения, операций бэкапа и восстановления данных, любых других сценариев, где требуется эффективное сжатие данных (рис. 9)

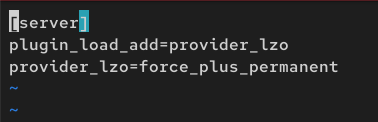


Рис. 9: Файл provider\_lzo.cnf

Файл provider\_snappy.cnf. Конфигурационный файл предназначен для подключения и настройки плагина сжатия Snappy в сервере MariaDB.

Загрузка плагина: Директива plugin\_load\_add должна гарантировать, что плагин provider\_snappy будет загружен при каждом запуске сервера.

Параметр force\_plus\_permanent обеспечивает отказоустойчивость (сервер не “упадет” из-за отсутствия плагина) и сохраняет настройку постоянной.

Практическое применение: после загрузки этого плагина MariaDB получает возможность использовать алгоритм Snappy для операций сжатия. Snappy — это алгоритм сжатия, разработанный Google, который характеризуется:

* Очень высокой скоростью сжатия и распаковки
* Умеренной степенью сжатия
* Оптимизацией для 64-битных систем

Snappy особенно полезен для:

* Сжатия данных в реальном времени
* Приложений, где критична скорость обработки данных
* Распределенных систем и баз данных
* Использования в различных механизмах хранения MariaDB (рис. 10)

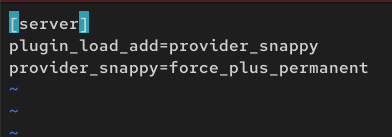


Рис. 10: Файл provider\_snappy.cnf

Файл spider.cnf. Конфигурационный файл содержит закомментированную директиву для загрузки плагина распределенного механизма хранения Spider. В текущем состоянии плагин отключен и не загружается сервером MariaDB. Для его активации необходимо раскомментировать строку plugin-load-add = ha\_spider и перезапустить сервер. Spider предоставляет возможности для создания шардированных и распределенных баз данных, что полезно для масштабирования систем (рис. 11)

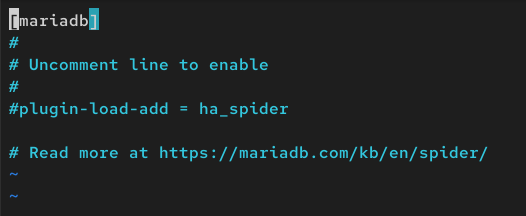


Рис. 11: Файл spider.cnf

Файл my.cnf. Главный конфигурационный файл my.cnf в его текущем состоянии не функционален. Критически важная директива includedir /etc/my.cnf.d закомментирована, что предотвращает загрузку всех модульных конфигурационных файлов из каталога /etc/my.cnf.d. Это означает, что настройки сервера, клиентов и плагинов, определенные в отдельных файлах, не будут применены. Для нормальной работы MariaDB необходимо раскомментировать строку includedir /etc/my.cnf.d. Файл демонстрирует структуру для централизованных настроек через секцию client-server, но в текущем виде не содержит активных конфигурационных параметров (рис. 12)

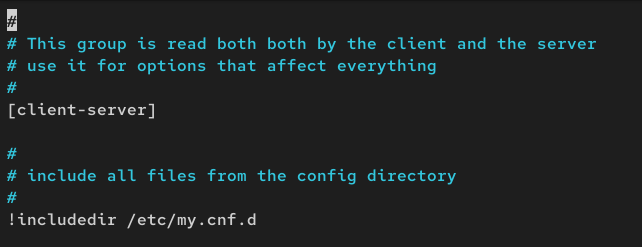


Рис. 12: Файл my.cnf

Для запуска и включения программного обеспечения mariadb используем:

systemctl start mariadb

systemctl enable mariadb (рис. 13)

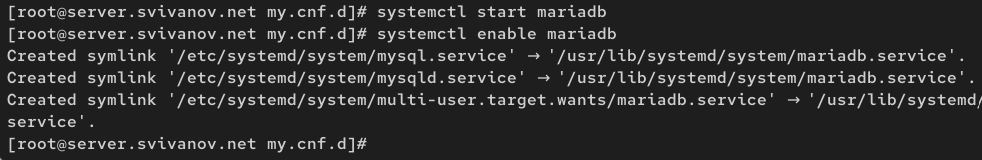


Рис. 13: Запуск mariaDB

Убедимся, что mariadb прослушивает порт, используя ss -tulpen | grep 3306 (рис. 14)

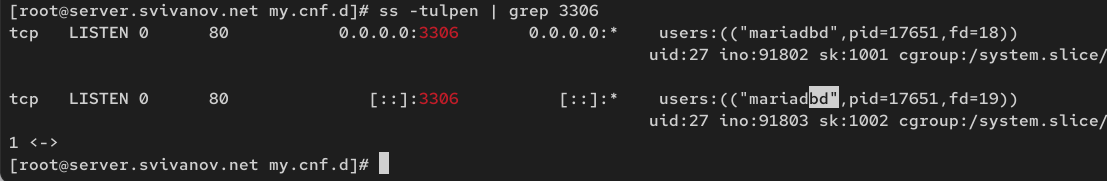


Рис. 14: Порт mariaDB

Запустим скрипт конфигурации безопасности mariadb, используя: mysql\_secure\_installation. С помощью запустившегося диалога и путём выбора Y/n установим пароль для пользователя root базы данных (обратите внимание, что это не пользователь root операционной системы), отключим удалённый корневой доступ и удалим тестовую базу данных и любых анонимных пользователей. (рис. 15)

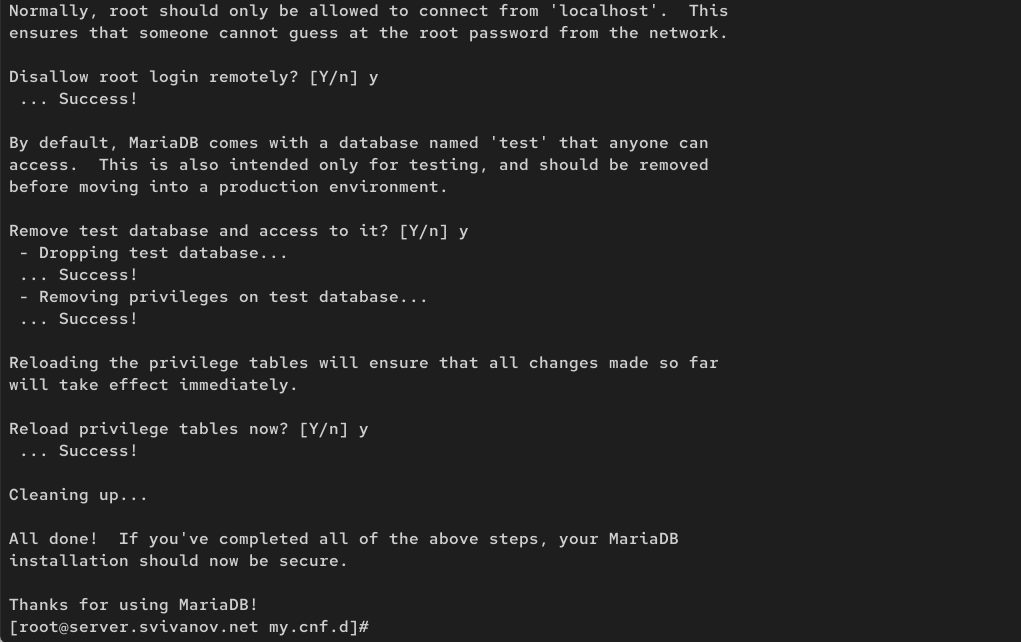


Рис. 15: Запуск скрипта безопасности

Для входа в базу данных с правами администратора базы данных введем mysql -u root -p. (рис. 16)



Рис. 16: Вход в базу данных

Просмотрим список команд MySQL, введя h. (рис. 17)

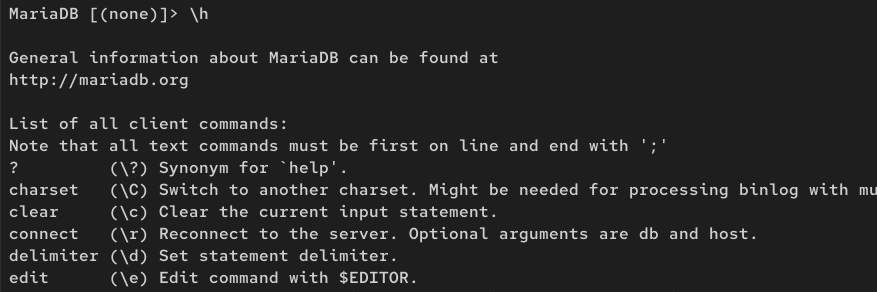


Рис. 17: Список команд

Из приглашения интерактивной оболочки MariaDB для отображения доступных в настоящее время баз данных введем MySQL-запрос SHOW DATABASES;

Были обнаружены четыре системные базы данных:

* information\_schema - содержит метаданные всех объектов СУБД
* mysql - хранит информацию о пользователях, привилегиях и системных настройках
* performance\_schema - предоставляет инструменты для мониторинга производительности
* sys - содержит упрощенные представления для анализа метрик производительности

Все перечисленные базы данных являются служебными и создаются автоматически при установке MariaDB. Пользовательских баз данных в системе не существует. (рис. 18)

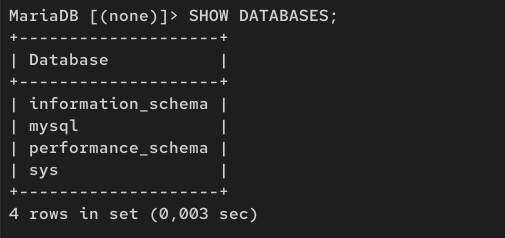


Рис. 18: Отображение доступных баз

## 3.2 Конфигурация кодировки символов

Войдем в базу данных с правами администратора: mysql -u root -p. Для отображения статуса MariaDB введем из приглашения интерактивной оболочки MariaDB: status

Команда status предоставляет информацию о текущем состоянии соединения и сервера MariaDB. Из вывода видно, что:

* Используется MariaDB версии 10.11.11
* Подключение выполнено локально через UNIX-сокет
* Сервер работает 10 минут 38 секунд
* Наблюдается несоответствие кодировок (сервер использует latin1, а клиент - utf8mb3)
* Нагрузка на сервер минимальная (в среднем 0.045 запросов в секунду) (рис. 19)

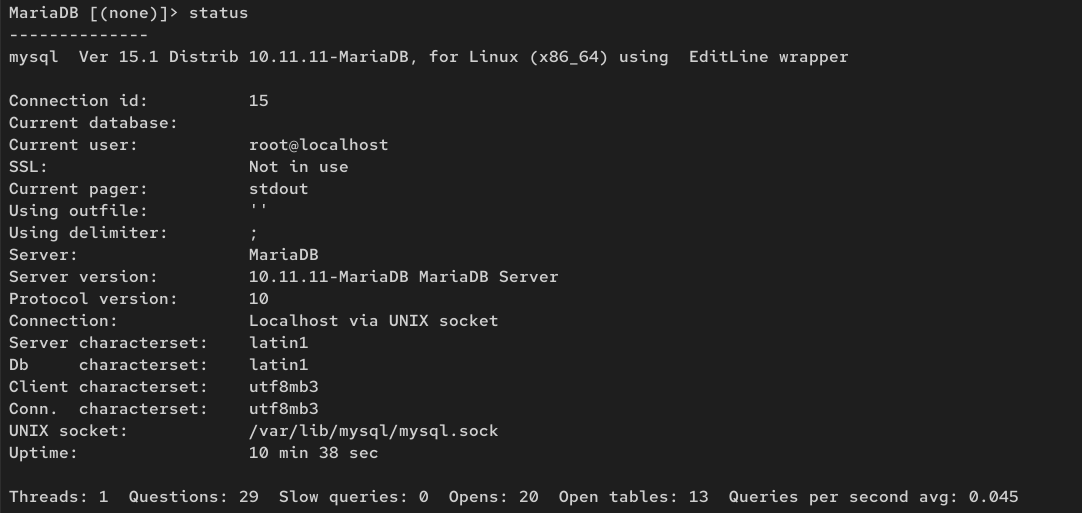


Рис. 19: Статус MariaDB

В каталоге /etc/my.cnf.d создадим файл utf8.cnf:

cd /etc/my.cnf.d

touch utf8.cnf (рис. 20)

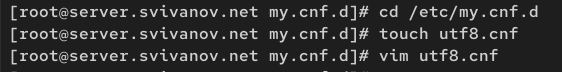


Рис. 20: Создание utf8.cnf

Откроем его на редактирование и укажем в нём следующую конфигурацию: (рис. 21)

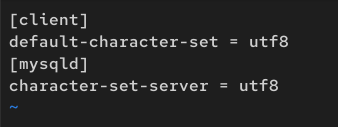


Рис. 21: Редактирование utf8.cnf

Перезапустим MariaDB: systemctl restart mariadb. (рис. 22)

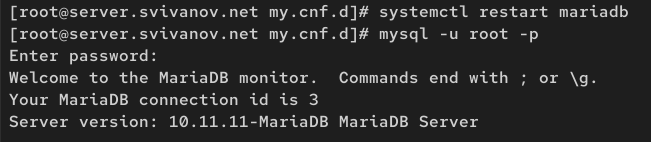


Рис. 22: Перезапуск базы данных

Войдем в базу данных с правами администратора и посмотрим статус MariaDB.

Изменение конфигурации:

* Кодировка сервера изменилась с latin1 на utf8mb3 - это указывает на то, что после перезапуска сервер начал использовать корректные настройки кодировки из конфигурационных файлов
* Сервер работает стабильно, время работы - 23 секунды
* Активно 1 подключение
* Медленных запросов не зафиксировано
* Нагрузка на сервер минимальная
* Перезапуск службы успешно применен и сервер MariaDB работает с обновленными настройками, включая корректную UTF-8 кодировку. (рис. 23)

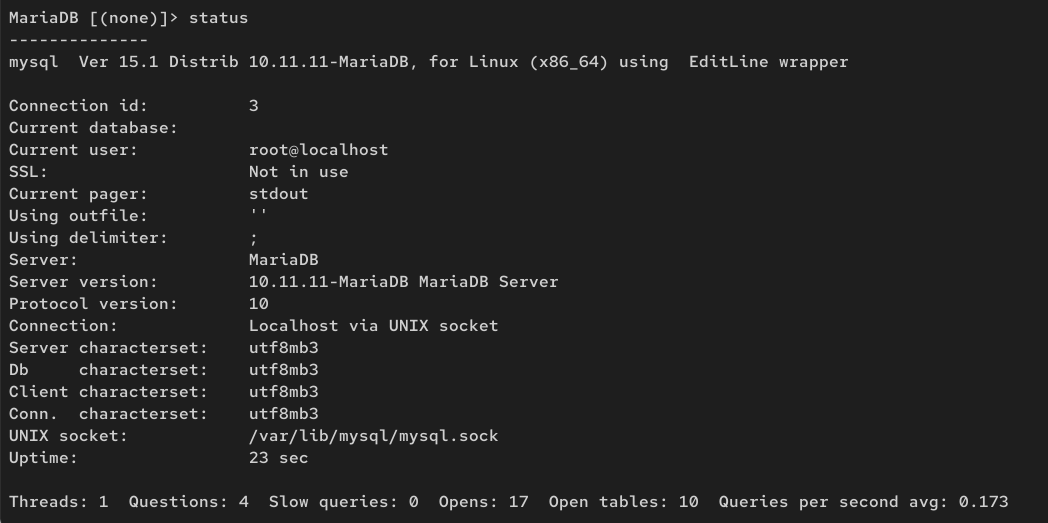


Рис. 23: Просмотр статуса

## 3.3 Создание базы данных

Войдем в базу данных с правами администратора, Создадим базу данных с именем addressbook:

CREATE DATABASE addressbook CHARACTER SET utf8 COLLATE utf8\_general\_ci;

Перейдем к базе данных addressbook

USE addressbook;

Отобразим имеющиеся в базе данных addressbook таблицы:

SHOW TABLES;(рис. 24)

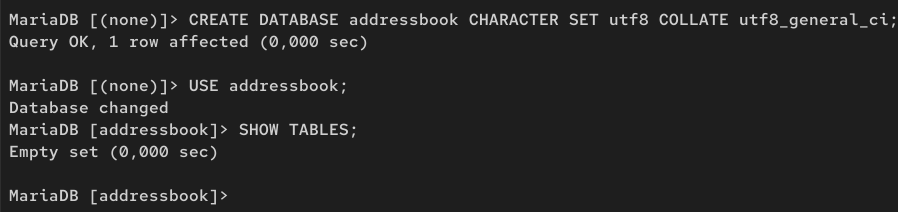


Рис. 24: Создание базы данных

Создадим таблицу city с полями name и city:

CREATE TABLE city(name VARCHAR(40), city VARCHAR(40));

Заполним несколько строк таблицы некоторыми данными по аналогии в соответствии с синтаксисом MySQL:

INSERT INTO city(name,city) VALUES (‘Иванов’,‘Москва’);

В частности, добавим в базу сведения о Петрове и Сидорове: (Петров, Сочи), (Сидоров, Дубна) (рис. 25)

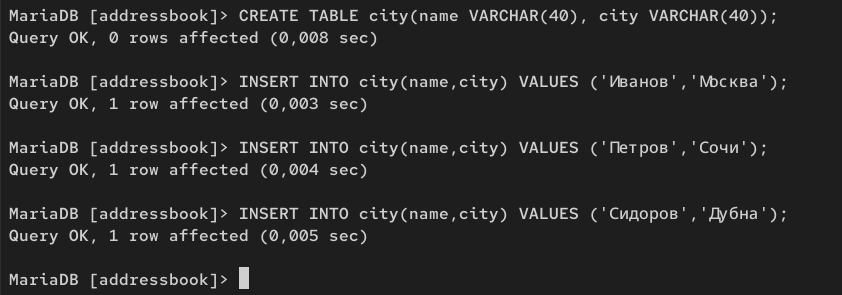


Рис. 25: Создание и наполнение таблицы

Сделаем следующий MySQL-запрос: SELECT \* FROM city;

Запрос SELECT \* FROM city успешно выполнен и показал содержимое таблицы city из базы данных addressbook.

Результат демонстрирует:

* Таблица содержит справочник “Фамилия - Город проживания”
* В текущий момент в таблице зарегистрировано 3 человека
* Все данные отображаются корректно, включая кириллические символы (что подтверждает правильную настройку кодировки UTF-8 после перезапуска сервера) (рис. 26)

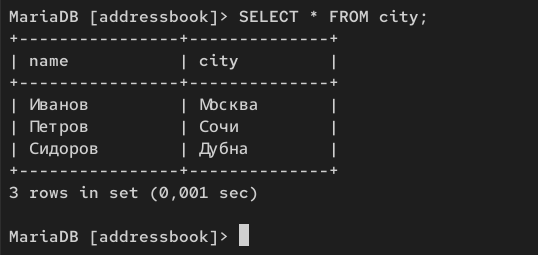


Рис. 26: Запрос к базе

Создадим пользователя для работы с базой данных addressbook и зададим для него пароль:

CREATE USER user@‘%’ IDENTIFIED BY ‘password’;

Предоставим права доступа созданному пользователю user на действия с базой данных addressbook (просмотр, добавление, обновление, удаление данных):

GRANT SELECT,INSERT,UPDATE,DELETE ON addressbook.\* TO user@‘%’; (рис. 27)

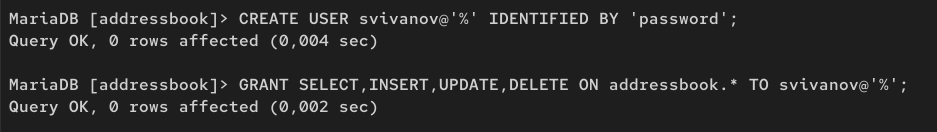


Рис. 27: Создание пользователя

Обновим привилегии (права доступа) базы данных addressbook: FLUSH PRIVILEGES; Посмотрим общую информацию о таблице city базы данных addressbook: DESCRIBE city; (рис. 28)

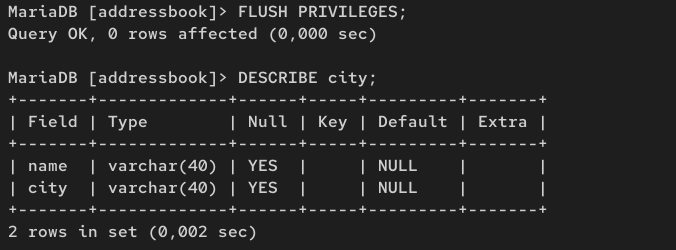


Рис. 28: Обновление привелегий и просмотр базы

Просмотрим список баз данных: mysqlshow -u root -p. Просмотрим список таблиц базы данных addressbook: mysqlshow -u root -p addressbook (рис. 29)

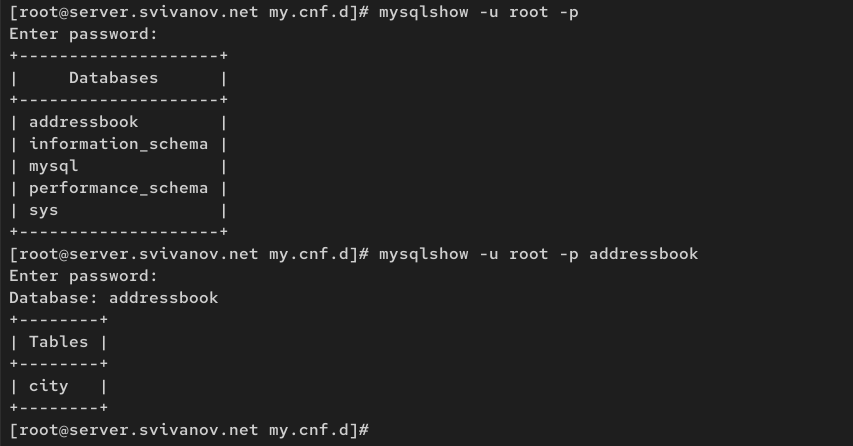


Рис. 29: Список баз и таблиц

## 3.4 Резервные копии

На виртуальной машине server создадим каталог для резервных копий:

mkdir -p /var/backup

Сделаем резервную копию базы данных addressbook:

mysqldump -u root -p addressbook > /var/backup/addressbook.sql

Сделаем сжатую резервную копию базы данных addressbook:

mysqldump -u root -p addressbook | gzip > /var/backup/addressbook.sql.gz

Сделаем сжатую резервную копию базы данных addressbook с указанием даты создания копии:

mysqldump -u root -p addressbook | gzip > $(date+/var/backup/addressbook.%Y%m%d.%H%M%S.sql.gz) (рис. 30)

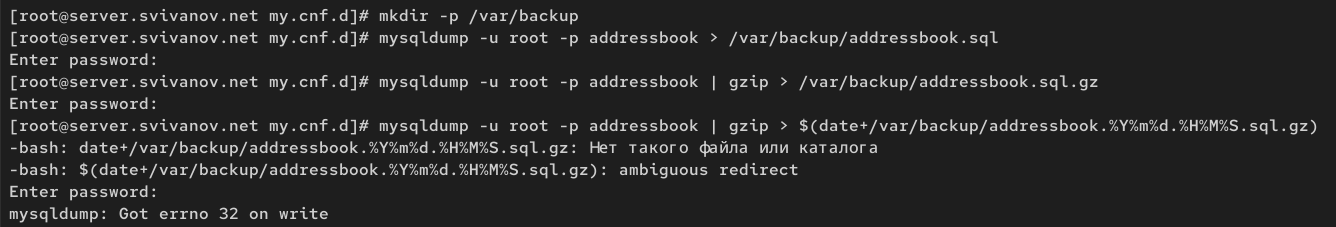


Рис. 30: Резервные копии

Восстановим базу данных addressbook из резервной копии:

mysql -u root -p addressbook < /var/backup/addressbook.sql

Восстановим базу данных addressbook из сжатой резервной копии:

zcat /var/backup/addressbook.sql.gz | mysql -u root -p addressbook (рис. 31)

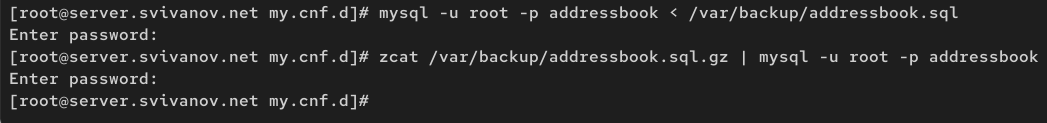


Рис. 31: Восстановление базы данных

На виртуальной машине server перейдем в каталог для внесения изменений в настройки внутреннего окружения /vagrant/provision/server/, создадим в нём каталог mysql, в который поместим в соответствующие подкаталоги конфигурационные файлы MariaDB и резервную копию базы данных addressbook:

cd /vagrant/provision/server

mkdir -p /vagrant/provision/server/mysql/etc/my.cnf.d

mkdir -p /vagrant/provision/server/mysql/var/backup

cp -R /etc/my.cnf.d/utf8.cnf /vagrant/provision/server/mysql/etc/my.cnf.d/

cp -R /var/backup/\* /vagrant/provision/server/mysql/var/backup/ (рис. 32)



Рис. 32: Создание каталогов для внесения изменений

В каталоге /vagrant/provision/server создадим исполняемый файл mysql.sh:

cd /vagrant/provision/server

touch mysql.sh

chmod +x mysql.sh

Открыв его на редактирование, пропишем в нём следующий скрипт: (рис. 33)

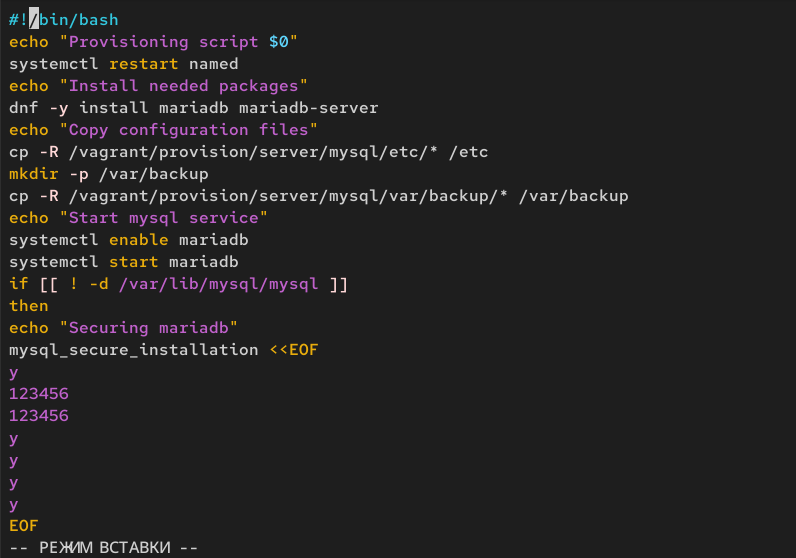


Рис. 33: Скрипт mysql.sh

Для отработки созданного скрипта во время загрузки виртуальных машин в конфигурационном файле Vagrantfile необходимо добавить в конфигурации сервера следующую запись (рис. 34)

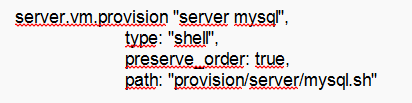


Рис. 34: Запись в Vagrantfile

# 4 Ответы на контрольные вопросы

**1. Какая команда отвечает за настройки безопасности в MariaDB?**

mysql\_secure\_installation

**2. Как настроить MariaDB для доступа через сеть?**

В конфигурационном файле /etc/my.cnf.d/mariadb-server.cnf:

**3. Какая команда позволяет получить обзор доступных баз данных после входа в среду оболочки MariaDB?**

SHOW DATABASES;

**4. Какая команда позволяет узнать, какие таблицы доступны в базе данных?**

USE database\_name;

SHOW TABLES;

**5. Какая команда позволяет узнать, какие поля доступны в таблице?**

DESCRIBE table\_name;

**6. Какая команда позволяет узнать, какие записи доступны в таблице?**

SELECT \* FROM table\_name;

**7. Как удалить запись из таблицы?**

DELETE FROM table\_name WHERE condition;

**8. Где расположены файлы конфигурации MariaDB? Что можно настроить с их помощью?**

/etc/my.cnf - главный конфигурационный файл

/etc/my.cnf.d/ - каталог с модульными конфигурационными файлами

**9. Где располагаются файлы с базами данных MariaDB?**

/var/lib/mysql/

**10. Как сделать резервную копию базы данных и затем её восстановить?**

Создание резервной копии:

mysqldump -u username -p database\_name > backup.sql

mysqldump -u username -p –all-databases > full\_backup.sql

Восстановление из резервной копии:

mysql -u username -p database\_name < backup.sql

mysql -u username -p < full\_backup.sql

# 5 Выводы

В ходе выполнения лабораторной работы мы приобрели практические навыков по установке и конфигурированию системы управления базами данных на примере программного обеспечения MariaDB.