

ГУАП

Кафедра № 33

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ  
ПРЕПОДАВАТЕЛЬ

Доцент, дтн

должность, уч. степень, звание

подпись, дата

Н.Н. Мошак

инициалы, фамилия

## ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

### АДМИНИСТРИРОВАНИЕ И НАСТРОЙКА ПОЛИТИКИ БЕЗОПАСНОСТИ СЕРВЕРА РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ MySQL

по курсу: БЕЗОПАСНОСТЬ ИНФОРМАЦИОННЫХ СИСТЕМ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

3931

подпись, дата

К.В. Жук

инициалы, фамилия

Санкт-Петербург 2022

### **Цель работы:**

Изучить команды MySQL и научиться устанавливать, администрировать SQL-сервер на примере сервера MySQL и настраивать его параметры безопасности.

Используемое программное обеспечение: операционная система Windows 10.

### **Основные сведения:**

Задача длительного хранения и обработки информации появилась практически сразу с появлением первых компьютеров. Для решения этой задачи в конце 60-х годов были разработаны специализированные программы, получившие название **систем управления базами данных (СУБД)**. СУБД проделали длительный путь эволюции от системы управления файлами, через иерархические и сетевые базы данных. В конце 80-х годов доминирующей стала **система управления реляционными базами данных (СУРБД)**. С этого времени такие СУБД стали стандартом де-факто, и для того, чтобы унифицировать работу с ними, был разработан **структурированный язык запросов (SQL)**, который представляет собой язык управления именно реляционными базами данных.

Существуют следующие разновидности баз данных:

- иерархические;
- реляционные;
- объектно-ориентированные;
- гибридные.

**Иерархическая** база данных основана на древовидной структуре хранения информации. В этом смысле иерархические базы данных очень напоминают файловую систему компьютера.

В **реляционных** базах данных данные собраны в таблицы, которые в свою очередь состоят из столбцов и строк, на пересечении которых расположены ячейки. Запросы к таким базам данных возвращает таблицу, которая повторно может участвовать в следующем запросе. Данные в одних таблицах, как правило, связаны с данными других таблиц, откуда и произошло название "реляционные".

В **объектно-ориентированных** базах данных данные хранятся в виде объектов. С объектно-ориентированными базами данных удобно работать, применяя объектно-ориентированное программирование. Однако, на сегодняшний день такие базы данных еще не достигли популярности реляционных, поскольку пока значительно уступают им в производительности.

**Гибридные** СУБД совмещают в себе возможности реляционных и объектно-ориентированных баз данных.

Эти модели характеризуются простотой структуры данных, удобным для пользователя табличным представлением и возможностью использования

формального аппарата алгебры отношений и реляционного исчисления для обработки данных.

Понятие реляционный (англ. *relation* — отношение) связано с разработками известного английского специалиста в области систем баз данных Эдгара Кодда (Edgar Codd). Модель реляционной базы данных представляет данные в виде таблиц, разбитых на строки и столбцы, на пересечении которых находятся данные. Кратко особенности реляционной базы данных можно описать следующим образом:

- Данные хранятся в таблицах, состоящих из столбцов и строк;
- На пересечении каждого столбца и строчки стоит в точности одно значение;
- У каждого столбца есть своё имя, которое служит его названием, и все значения в одном столбце имеют один тип. Например, в столбце `id_forum` все значения имеют целочисленный тип, а в строке `name` - текстовый;
- Столбцы располагаются в определённом порядке, который определяется при создании таблицы, в отличие от строк, которые располагаются в произвольном порядке. В таблице может не быть не одной строчки, но обязательно должен быть хотя бы один столбец;
- Запросы к базе данных возвращают результат в виде таблиц, которые тоже могут выступать как объект запросов.

Для работы с базами данных используется язык SQL. Стандарт SQL определен ANSI (American National Standard Institute). Однако SQL не является изобретением ANSI, он – продукт исследования фирмы IBM, проводимого в начале 70-х годов 20 века. Другие компании и учебные заведения также внесли вклад в создание этого языка, например компания Oracle или Калифорнийский университет Беркли. После появления на рынке нескольких конкурирующих продуктов, ANSI определил стандарт, которому они должны следовать. Однако введение стандарта *post factum* породило ряд проблем. В итоге стандарт SQL оказался в некотором смысле ограничен: то, что определено ANSI, не всегда является наиболее полезным с точки зрения практического применения, поэтому разработчики SQL-продуктов стремятся разрабатывать их таким образом, чтобы они соответствовали стандарту ANSI, но не были им слишком жестко ограниченным. Что опять же приводит к использованию отдельных команд языка SQL, специфичных у каждого из разработчиков. При этом наиболее удачные решения нередко заимствуются другими разработчиками и, в свою очередь, также со временем становятся нормой. Поэтому периодически производится уточнение стандарта SQL. Первые попытки стандартизировать язык SQL были неудачными: стандарты SQL/86 и SQL/89 (принятые соответственно в 1986 и 1989 годах) недостаточно четко прописывали требования и ограничения, что приводило к значительным расхождениям в реализации SQL различными производителями. Первый реально действующий стандарт был принят в 1992г и известен как SQL/92. В дальнейшем были разработаны SQL:1999, SQL:2003, SQL:2006 и SQL:2008.

## 2.2. Общие сведения о базе данных MySQL

Разработчиком MySQL, популярной SQL-базы данных с открытым кодом, является компания MySQL AB. В настоящее время компания куплена корпорацией Oracle, которой и принадлежит теперь продукт. Однако MySQL по-прежнему остается базой данных с открытым кодом. Свое происхождение MySQL ведет от продукта mSQL, разработанного в конце 1970-х гг. компанией ТсХ и использовавшемуся для доступа к таблицам, для которых использовались собственные быстрые подпрограммы низкого уровня. Однако после тестирования был сделан вывод, что скорость и гибкость mSQL недостаточны. В результате для базы данных был разработан новый SQL-интерфейс. Новый продукт получил название MySQL. Массовое же признание MySQL получила, начиная с линейки продуктов версии 3, которые стали широко использоваться на серверах в сети Интернет. В настоящее время используется 5 версия продукта.

Ниже приведено описание важных характеристик программного обеспечения MySQL:

- Внутренние характеристики и переносимость
  - Написан на С и С++. Протестирован на множестве различных компиляторов.
  - Работает на различных аппаратных платформах и разных операционных системах.
  - Высокая производительность за счет максимально оптимизированного кода, эффективной системы распределения памяти и продуманной системы дисковых таблиц.
- Безопасность
  - Система, основанная на привилегиях и паролях, за счет чего обеспечивается гибкость и безопасность, и с возможностью верификации с удаленного компьютера. Пароли защищены, т.к. они при передаче по сети при соединении с сервером шифруются.
- Масштабируемость
  - Способность работать с очень большими базами данных (десятки и сотни миллионов записей).
  - Возможность кластеризации серверов и распределения обработки информации между серверами

Тип ИС закрытого контура в соответствии с вариантом – **2Б**

### **Требования к классу защищенности 2Б:**

#### Подсистема управления доступом:

Должны осуществляться идентификация и проверка подлинности субъектов доступа при входе в систему по идентификатору (коду) и паролю условно-постоянного действия длиной не менее шести буквенно-цифровых символов.

#### Подсистема регистрации и учета:

Должна осуществляться регистрация входа (выхода) субъектов доступа в систему (из системы), либо регистрация загрузки и инициализации операционной системы и ее программного останова. Регистрация выхода из системы или останова не проводится в моменты аппаратурного отключения АС.

#### В параметрах регистрации указываются:

- дата и время входа (выхода) субъекта доступа в систему (из системы) или загрузки (останова) системы;
- результат попытки входа: успешная или неуспешная (при НСД);
- должен проводиться учет всех защищаемых носителей информации с помощью их маркировки и с занесением учетных данных в журнал (учетную карточку).

#### Подсистема обеспечения целостности:

Должна быть обеспечена целостность программных средств СЗИ НСД, обрабатываемой информации, а также неизменность программной среды.

При этом:

- целостность СЗИ НСД проверяется при загрузке системы по наличию имен (идентификаторов) компонент СЗИ;
- целостность программной среды обеспечивается отсутствием в АС средств разработки и отладки программ во время обработки и (или) хранения защищаемой информации;
- должна осуществляться физическая охрана СВТ (устройств и носителей информации), предусматривающая контроль доступа в помещения АС посторонних лиц, наличие надежных препятствий для несанкционированного проникновения в помещения АС и хранилище носителей информации, особенно в нерабочее время;
- должно проводиться периодическое тестирование функций СЗИ НСД при изменении программной среды и персонала АС с помощью тест - программ, имитирующих попытки НСД;
- должны быть в наличии средства восстановления СЗИ НСД, предусматривающие ведение двух копий программных средств СЗИ НСД и их периодическое обновление и контроль работоспособности.

Подсистемы и требования	Класс 2Б
<b>1. Подсистема управления доступом</b>	
<b>1.1. Идентификация, проверка подлинности и контроль доступа субъектов:</b>	
в систему	+
к терминалам, ЭВМ, узлам сети ЭВМ, каналам связи, внешним устройствам ЭВМ	-
к программам	-

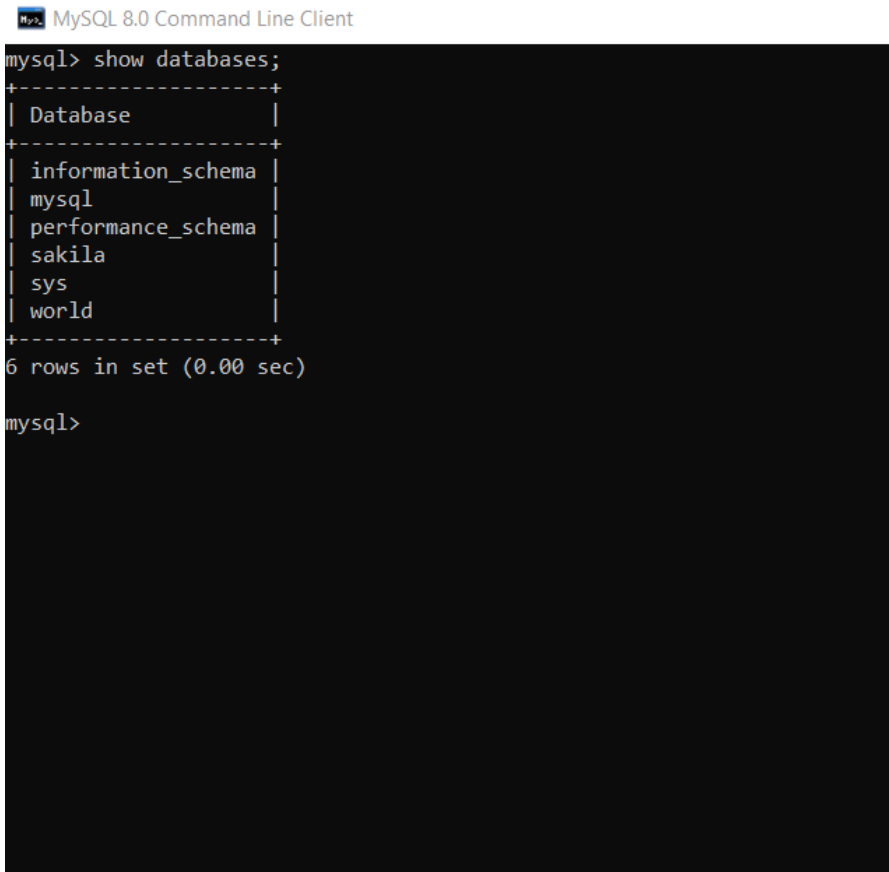
к томам, каталогам, файлам, записям, полям записей	-
<b>1.2. Управление потоками информации</b>	
<b>2. Подсистема регистрации и учета</b>	
<b>2.1. Регистрация и учет:</b>	
входа (выхода) субъектов доступа в (из) систему (узел сети)	+
выдачи печатных (графических) выходных документов	-
запуска (завершения) программ и процессов (заданий, задач)	-
доступа программ субъектов доступа к защищаемым файлам, включая их создание и удаление, передачу по линиям и каналам связи	-
доступа программ субъектов доступа к терминалам, ЭВМ, узлам сети ЭВМ, каналам связи, внешним устройствам ЭВМ, программам, томам, каталогам, файлам, записям, полям записей	-
изменения полномочий субъектов доступа	-
создаваемых защищаемых объектов доступа	-
<b>2.2. Учет носителей информации</b>	+
<b>2.3. Очистка (обнуление, обезличивание) освобождаемых областей оперативной памяти ЭВМ и внешних накопителей</b>	-
<b>2.4. Сигнализация попыток нарушения защиты</b>	-
<b>3. Криптографическая подсистема</b>	
<b>3.1. Шифрование конфиденциальной информации</b>	-
<b>3.2. Шифрование информации, принадлежащей различным субъектам доступа (группам субъектов) на разных ключах</b>	-
<b>3.3. Использование аттестованных (сертифицированных) криптографических средств</b>	-
<b>4. Подсистема обеспечения целостности</b>	
<b>4.1. Обеспечение целостности программных средств и обрабатываемой информации</b>	+
<b>4.2. Физическая охрана средств вычислительной техники и носителей информации</b>	+
<b>4.3. Наличие администратора (службы) защиты информации в АС</b>	-
<b>4.4. Периодическое тестирование СЗИ НСД</b>	+
<b>4.5. Наличие средств восстановления СЗИ НСД</b>	+
<b>4.6. Использование сертифицированных средств защиты</b>	-

### Ход выполнения работы:

После установки базы данных MySQL можно приступить к работе.

Для начала зайдём в командную строку клиента под именем администратора (root), посмотрим, какие базы данных имеются на данный момент.

**Команда:** show databases;



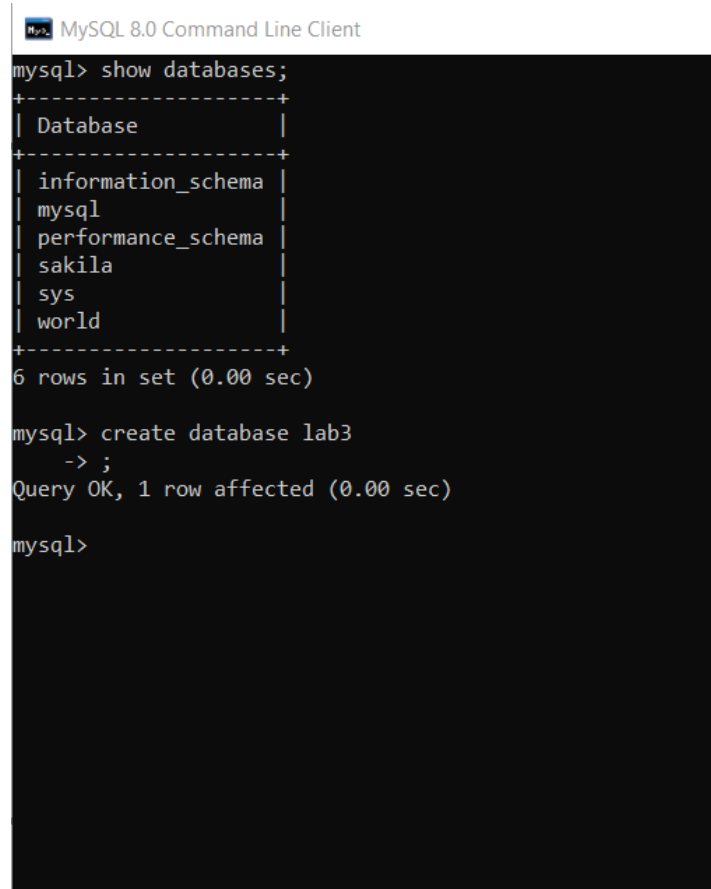
```
MySQL 8.0 Command Line Client
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql       |
| performance_schema |
| sakila      |
| sys         |
| world       |
+-----+
6 rows in set (0.00 sec)

mysql>
```

*Рисунок 1 – Существующие базы данных*

Для начала создадим учебную базу данных, на примере которой и будет проводиться вся работа. Назовем её lab3.

**Команда:** create database lab3;



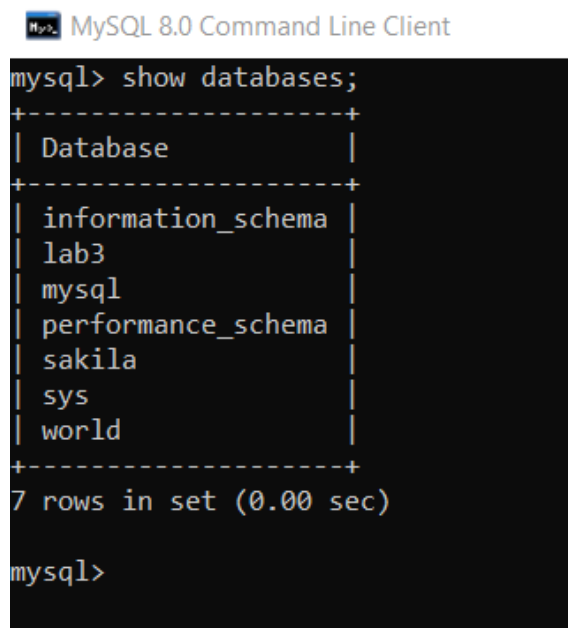
```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sakila |
| sys |
| world |
+-----+
6 rows in set (0.00 sec)

mysql> create database lab3
-> ;
Query OK, 1 row affected (0.00 sec)

mysql>
```

Рисунок 2 – Создание своей базы данных

Ещё раз используем команду show databases, чтобы удостовериться в том, что база данных была создана. Действительно, в списке появилась новая база данных.



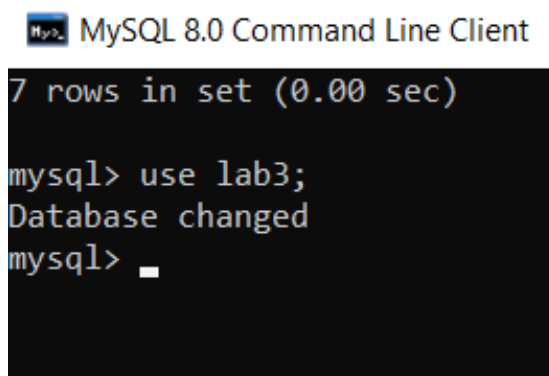
```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| lab3 |
| mysql |
| performance_schema |
| sakila |
| sys |
| world |
+-----+
7 rows in set (0.00 sec)

mysql>
```

Рисунок 3 – Новый список баз данных



Для работы с конкретной базой данных нужно указать эту самую базу.  
Команда: use lab3;



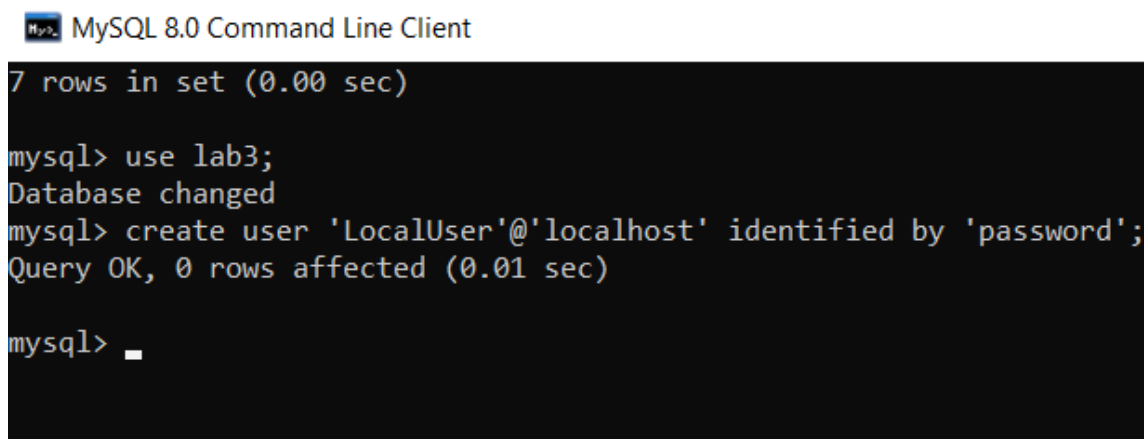
```
MySQL 8.0 Command Line Client
7 rows in set (0.00 sec)

mysql> use lab3;
Database changed
mysql> _
```

Рисунок 4 – Выбор базы данных

Теперь создадим локального пользователя с именем LocalUser и паролем password, который будет имитировать обычного работника, которому требуется доступ к базе с информацией. Создание именно локального пользователя требуется для более простой выдачи прав доступа. Дело в том, что для глобального пользователя требуется задавать права доступа на *каждой* машине, под которой он может подключиться к базе данных, а это не удобно и сильно усложняет работу специалистов ИБ.

**Команда:** create user 'LocalUser'@'localhost' identified by 'password';



```
MySQL 8.0 Command Line Client
7 rows in set (0.00 sec)

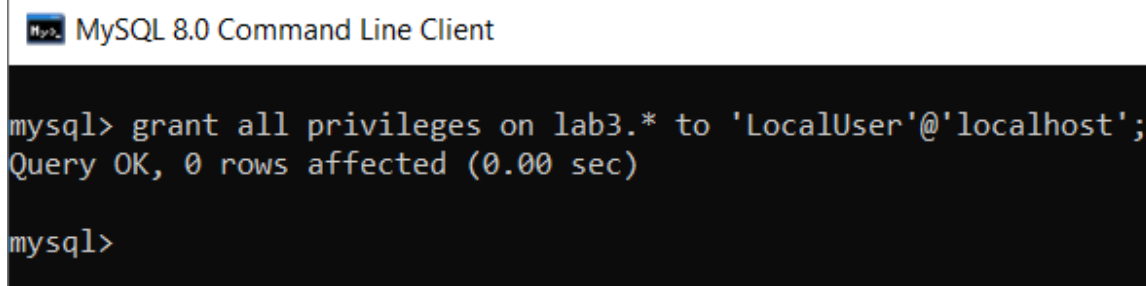
mysql> use lab3;
Database changed
mysql> create user 'LocalUser'@'localhost' identified by 'password';
Query OK, 0 rows affected (0.01 sec)

mysql> _
```

Рисунок 5 – Создание локального пользователя

Далее пользователю требуется выдать права. В качестве примера выдадим пользователю все возможные права.

**Команда:** grant all privileges on lab3.\* to 'LocalUser'@'localhost';



```
MySQL 8.0 Command Line Client

mysql> grant all privileges on lab3.* to 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.00 sec)

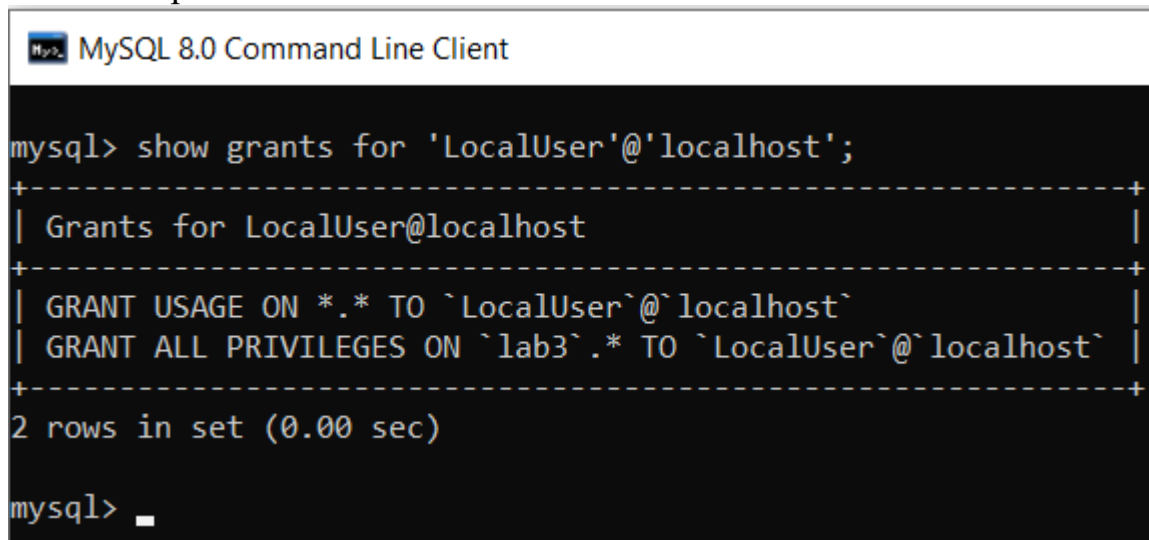
mysql>
```

Рисунок 6 – Выдача прав пользователю

Проверим, какими правами обладает пользователь.

**Команда:** `show grants for 'LocalUser'@'localhost';`

Как видно из рисунка 7, все в порядке, пользователь имеет все возможные права.

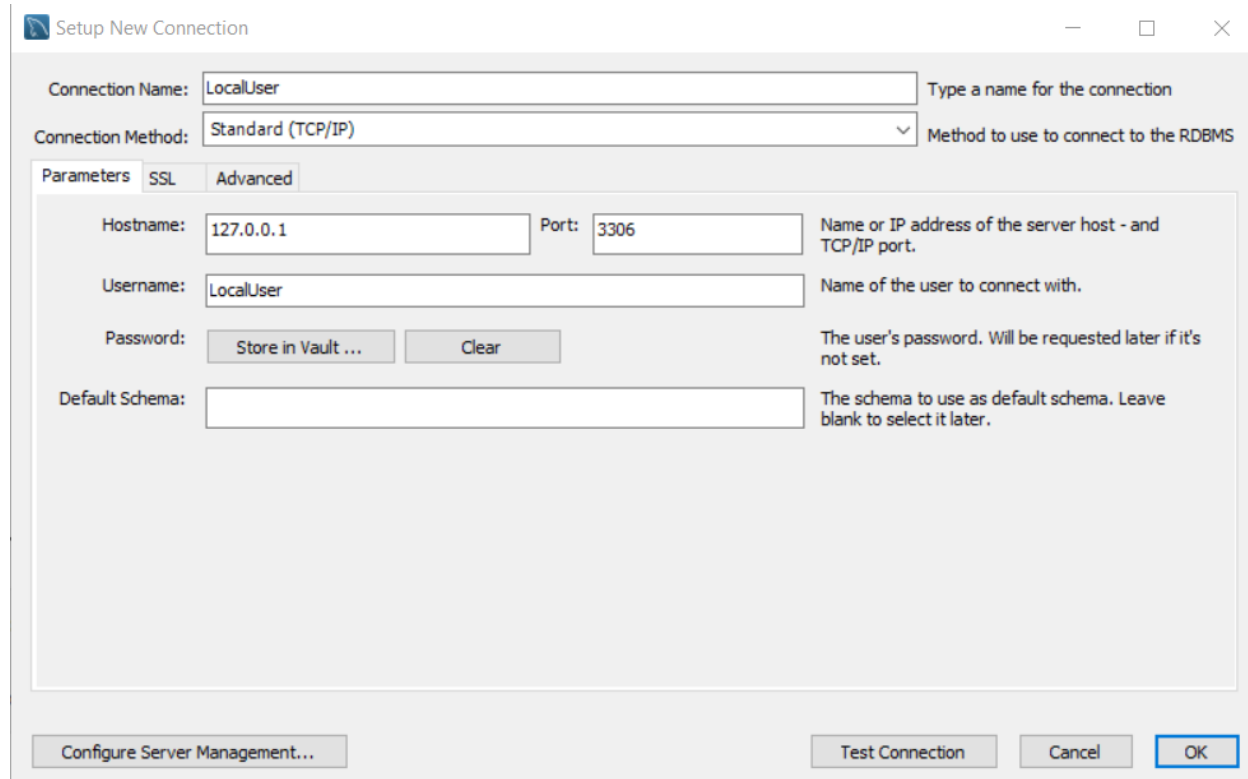


```
mysql> show grants for 'LocalUser'@'localhost';
+-----+
| Grants for LocalUser@localhost |
+-----+
| GRANT USAGE ON *.* TO `LocalUser`@`localhost` |
| GRANT ALL PRIVILEGES ON `lab3`.* TO `LocalUser`@`localhost` |
+-----+
2 rows in set (0.00 sec)

mysql> _
```

Рисунок 7 – Права пользователя

Теперь требуется подключить созданного пользователя к серверу базы данных. Для этого воспользуемся утилитой MySQL Workbench и добавим нашего пользователя.



The screenshot shows the 'Setup New Connection' dialog box in MySQL Workbench. The 'Connection Name' field is set to 'LocalUser'. The 'Connection Method' is set to 'Standard (TCP/IP)'. The 'Parameters' tab is selected, showing the following fields: 'Hostname' (127.0.0.1), 'Port' (3306), 'Username' (LocalUser), 'Password' (with 'Store in Vault ...' and 'Clear' buttons), and 'Default Schema' (empty). The 'SSL' and 'Advanced' tabs are also visible. At the bottom, there are buttons for 'Configure Server Management...', 'Test Connection', 'Cancel', and 'OK'.

Рисунок 8 – Создание нового соединения

После подключения проверяем MySQL Connection и видим, что все в порядке. Теперь у нас есть 2 пользователя: Администратор и локальный пользователь.

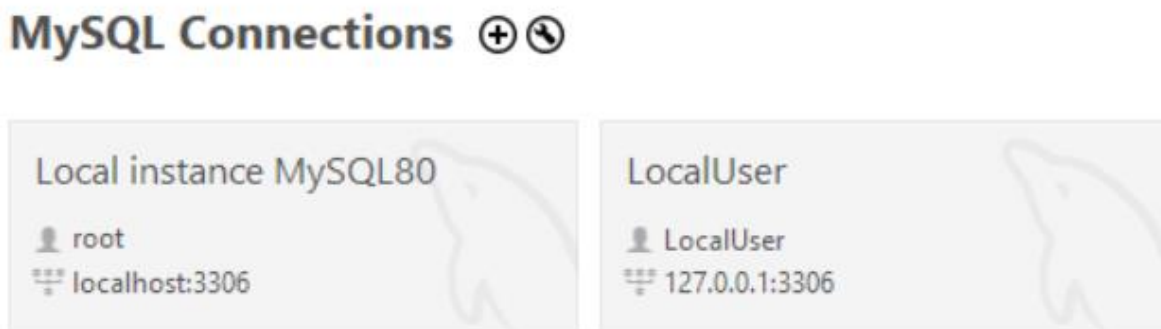


Рисунок 9 – Панель пользователей

Для дальнейшей работы нам потребуется заполненная база данных.

Для этого создаем таблицу в нашей учебной базе данных и заполняем её данными. Это можно сделать как при помощи учетной записи администратора, так и при помощи учетной записи пользователя, так как мы выдали ему нужные права ранее.

В качестве примера будет создана таблица с телефонными номерами.

**Команда для создания таблицы:** CREATE TABLE phoneNumbers (UserName text, UserAddress text, UserPhone text);

```
mysql> CREATE TABLE phoneNumbers(UserName text,UserAddress text,UserPhone text)
-> ;
Query OK, 0 rows affected (0.02 sec)
```

Рисунок 10 – Создание таблицы

**Команда для вставки данных:** INSERT INTO phoneNumbers (UserName, UserAddress, User Phone) values ('Kirill', 'Murmansk', '+78005553535');

```
MySQL 8.0 Command Line Client
Empty set (0.00 sec)

mysql> INSERT INTO phoneNumbers(UserName, UserAddress, UserPhone) values('Kirill','Murmansk','+78005553535');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO phoneNumbers(UserName, UserAddress, UserPhone) values('Maksim','SPB','+79091234578');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO phoneNumbers(UserName, UserAddress, UserPhone) values('Maria','SPB','+78509567183');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO phoneNumbers(UserName, UserAddress, UserPhone) values('Petr','Ryazan','+79215734975');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO phoneNumbers(UserName, UserAddress, UserPhone) values('Andrey','Murmansk','+79216238931');
Query OK, 1 row affected (0.00 sec)

mysql>
```

Рисунок 11 – Добавление данных

Проверим созданную таблицу.

Команда: `SELECT * FROM phoneNumbers;`

```
mysql> SELECT * FROM phoneNumbers
-> ;
+-----+-----+-----+
| UserName | UserAddress | UserPhone |
+-----+-----+-----+
| Kirill   | Murmansk   | +78005553535 |
| Maksim   | SPB        | +79091234578 |
| Maria    | SPB        | +78509567183 |
| Petr     | Ryazan     | +79215734975 |
| Andrey   | Murmansk   | +79216238931 |
+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

*Рисунок 12 – Вывод данных из таблицы*

Таблица создана, теперь требуется выдать пользователю нужные права доступа и протестировать их.

Для этого заберем все ранее выданные права и выдадим нужные.

Существующие права в MySQL:

- **CREATE** – Позволяет пользователям создавать базы данных/таблицы
- **SELECT** – Позволяет пользователям делать выборку данных
- **INSERT** – Позволяет пользователям добавлять новые записи в таблицы
- **UPDATE** – Позволяет пользователям изменять существующие записи в таблицах
- **DELETE** – Позволяет пользователям удалять записи из таблиц
- **DROP** – Позволяет пользователям удалять записи в базе данных/таблицах

Поскольку наш пользователь является обычным работником и при этом не является привилегированным пользователем, то выдадим ему только права на выборку данных и на добавление новых данных. Остальные права могут нанести вред организации, так как позволяют рядовому пользователю изменять уже существующие данные, удалять как строки, так и целые таблицы, а также создавать новые таблицы.

**Команда для удаления прав:** `revoke all privileges on *.* from 'LocalUser'@'localhost';`

```
mysql> revoke all privileges on *.* from 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

*Рисунок 13 – Удаление существующих прав*

**Команда для выдачи прав:** `grant SELECT, INSERT on lab3.phoneNumbers to 'LocalUser'@'localhost';`

```
mysql> grant SELECT, INSERT on lab3.phoneNumbers to 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

*Рисунок 14 – Выдача новых прав*

**Команда для проверки прав:** `show grants for 'LocalUser'@'localhost';`

```
mysql> show grants for 'LocalUser'@'localhost';
+-----+
| Grants for LocalUser@localhost |
+-----+
| GRANT USAGE ON *.* TO `LocalUser`@`localhost` |
| GRANT SELECT, INSERT ON `lab3`.`phonenumbers` TO `LocalUser`@`localhost` |
+-----+
2 rows in set (0.00 sec)
```

*Рисунок 15 – Проверка существующих прав*

Теперь можно приступить к работе под учетной записью обычного пользователя.

Для этого заходим в консоль пользователя через утилиту MySQL Workbench

После введения пароля используем ранее рассмотренную команду и выводим на экран список доступных баз. Учебная база доступна пользователю.

```
C:\Windows\system32\cmd.exe
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 19
Server version: 8.0.31 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| lab3 |
| performance_schema |
+-----+
3 rows in set (0.00 sec)

mysql>
```

*Рисунок 16 – Консоль обычного пользователя*

Попробуем вывести на экран данные о конкретном человеке.

**Команда:** SELECT UserAddress, UserPhone FROM phoneNumbers where UserName='Kirill';

**Команда:** SELECT UserAddress, UserPhone FROM phoneNumbers where UserName='Andrey';

```
mysql> SELECT UserAddress, UserPhone FROM phoneNumbers where UserName='Kirill';
+-----+-----+
| UserAddress | UserPhone |
+-----+-----+
| Murmansk   | +78005553535 |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT UserAddress, UserPhone FROM phoneNumbers where UserName='Andrey';
+-----+-----+
| UserAddress | UserPhone |
+-----+-----+
| Murmansk   | +79216238931 |
+-----+-----+
1 row in set (0.00 sec)

mysql>
```

Рисунок 17 – Вывод данных на экран

Выведем на экран всю таблицу, но с сортировкой столбца UserName по алфавиту.

**Команда:** SELECT \* FROM phoneNumbers order by UserName asc;

```
mysql> SELECT * FROM phoneNumbers order by UserName asc;
+-----+-----+-----+
| UserName | UserAddress | UserPhone |
+-----+-----+-----+
| Andrey   | Murmansk   | +79216238931 |
| Kirill   | Murmansk   | +78005553535 |
| Maksim   | SPB        | +79091234578 |
| Maria    | SPB        | +78509567183 |
| Petr     | Ryazan     | +79215734975 |
+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

Рисунок 18 – Отсортированные данные

Попробуем применить запрещенную команду.

**Команда:** drop table phoneNumbers;

```
mysql> drop table phoneNumbers;
ERROR 1142 (42000): DROP command denied to user 'LocalUser'@'localhost' for table 'phonenumbers'
mysql>
```

Рисунок 19 – Попытка удаления таблицы

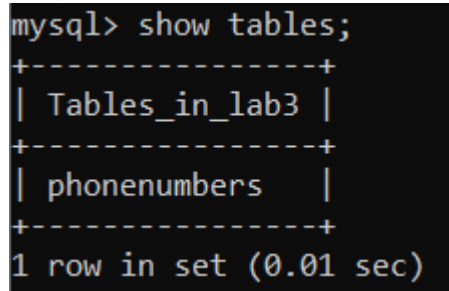
Как видно из скриншота, пользователь не смог удалить таблицу, поскольку не имел нужных прав.

Можно сделать вывод о том, что учетная запись обычного пользователя была настроена верно, поэтому он не сможет нанести вред организации.

В качестве завершения, вернемся в учетную запись администратора и последовательно удалим таблицу, базу данных и локального пользователя.

Проверяем существующие таблицы в базе данных.

**Команда:** show tables;

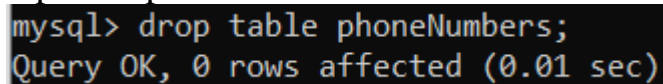


```
mysql> show tables;
+-----+
| Tables_in_lab3 |
+-----+
| phonenumbers   |
+-----+
1 row in set (0.01 sec)
```

*Рисунок 20 – Существующие таблицы*

Удаляем таблицу.

**Команда:** drop table phoneNumbers'

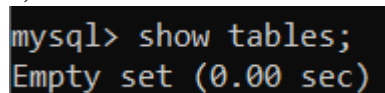


```
mysql> drop table phoneNumbers;
Query OK, 0 rows affected (0.01 sec)
```

*Рисунок 21 – Удаление таблицы*

Проверяем корректность удаления таблицы.

**Команда:** show tables;



```
mysql> show tables;
Empty set (0.00 sec)
```

*Рисунок 22 – Проверка удаления таблицы*

Смотрим список существующих баз, удаляем базу и осуществляем проверку удаления.

**Команда:** show databases;

**Команда:** drop database lab3;

**Команда:** show databases;

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| lab3 |
| mysql |
| performance_schema |
| sakila |
| sys |
| world |
+-----+
7 rows in set (0.00 sec)

mysql> drop database lab3;
Query OK, 0 rows affected (0.01 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sakila |
| sys |
| world |
+-----+
```

*Рисунок 23 – Удаление базы данных*

Удаляем права локального пользователя.

**Команда:** revoke all privileges on \*.\* from 'LocalUser'@'localhost';

```
mysql> revoke all privileges on *.* from 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> show grants for 'LocalUser'@'localhost';
+-----+
| Grants for LocalUser@localhost |
+-----+
| GRANT USAGE ON *.* TO `LocalUser`@`localhost` |
+-----+
1 row in set (0.00 sec)
```

*Рисунок 24 – Удаление всех прав пользователя*



Просматриваем список существующих пользователей, удаляем нужного и проверяем корректность удаления.

**Команда:** SELECT User, Host FROM mysql.user;

**Команда:** drop user 'LocalUser'@'localhost';

**Команда:** SELECT User, Host FROM mysql.user;

```
mysql> SELECT User, Host FROM mysql.user;
+-----+-----+
| User          | Host          |
+-----+-----+
| LocalUser     | localhost     |
| mysql.infoschema | localhost     |
| mysql.session  | localhost     |
| mysql.sys      | localhost     |
| root          | localhost     |
+-----+-----+
5 rows in set (0.00 sec)

mysql> drop user 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT User, Host FROM mysql.user;
+-----+-----+
| User          | Host          |
+-----+-----+
| mysql.infoschema | localhost     |
| mysql.session    | localhost     |
| mysql.sys        | localhost     |
| root           | localhost     |
+-----+-----+
4 rows in set (0.00 sec)
```

*Рисунок 25 – Удаление локального пользователя*

#### **Вывод:**

В результате выполнения лабораторной работы была проведена установка MySQL, создана база данных. Назначены и проверены права пользователя по доступу к ресурсам сервера базы данных. Настройка параметров безопасности с учетом принятой политики информационной безопасности в организации - прав доступа к ресурсам сервера базы данных, а именно, SELECT, права на выборку из всех записей таблицы (с целью поиска определенного человека, например, и уточнения его личных данных), и INSERT, право на добавление новых записей (если потребуется клиентскую базу) позволяет повысить уровень защиты базы данных от НСД.

Что же касается требования ФСТЭК к классу защищенности 2Б, ограничения прав пользователей базы данных помогает частично выполнить пункт 4.1. «Обеспечение целостности программных средств и обрабатываемой информации», а именно целостность обрабатываемой информации.