

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное автономное образовательное учреждение высшего образования  
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»  
КАФЕДРА № 51

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ  
ПРЕПОДАВАТЕЛЬ

Профессор, д.т.н.

Н. Н. Мошак

\_\_\_\_\_  
должность, уч. степень,  
звание

\_\_\_\_\_  
подпись, дата

\_\_\_\_\_  
инициалы, фамилия

**ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3**

Администрирование и настройка политики безопасности  
сервера реляционной базы данных

по курсу: БЕЗОПАСНОСТЬ ИНФОРМАЦИОННЫХ СИСТЕМ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

5712

Коваленко Д.И.

\_\_\_\_\_  
подпись, дата

\_\_\_\_\_  
инициалы, фамилия

Санкт-Петербург 2020

## 1. Цель работы

Изучение команд MySQL и системы привилегий (privilege system). Научиться устанавливать и администрировать SQL-сервер на примере сервера MySQL, а также настраивать его параметры безопасности.

Используемое программное обеспечение: ОС версии не ниже WindowsXP. А именно Windows 7 Professional (x64).

**Тип ИС закрытого контура: 1Г.**

## 2. Теоретические сведения

### *Реляционные базы данных. Общие 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:

1. Иерархические – база данных, основанная на древовидной структуре хранения информации;
2. Реляционные – базы данных, данные в которых 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»;
3. Объектно-ориентированные – базы данных, в которых 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;
4. Гибридные – СУБД, совмещающие 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.

Кратко особенности реляционной базы данных можно описать следующим образом:

- Данные хранятся в таблицах, состоящих из столбцов и строк;
- На пересечении каждого столбца и строки стоит в точности одно значение;

➤ У каждого столбца есть своё имя, которое служит его названием, и все значения в одном столбце имеют один тип.

➤ 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 (Structured Query Language — язык структурированных запросов). Стандарт SQL определен ANSI (American National

Standart Institute). SQL предназначен для манипуляции данными, которые хранятся в Системах управления реляционными базами данных (RDBMS). SQL имеет команды, с помощью которых данные можно извлекать, сортировать, обновлять, удалять и добавлять.

SQL можно использовать с такими базами RDBMS как MySQL, mSQL, PostgreSQL, Oracle, Microsoft SQL Server, Access, Sybase, Ingres. Эти системы RDBMS поддерживают все важные и общепринятые операторы SQL, однако каждая из них имеет множество своих собственных патентованных операторов и расширений.

### ***База данных MySQL. Общие сведения.***

MySQL, которая является RDBMS с открытым исходным кодом, доступна для загрузки на сайте MySQL.com. Разработчиком MySQL является компания MySQL AB. В настоящее время компания куплена корпорацией Oracle, которой и принадлежит теперь продукт. Свое происхождение MySQL ведет от продукта mSQL, разработанного в конце 1970-х гг. компанией TeX 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. Вот как характеризуют MySQL её разработчики.

1. MySQL - это система управления реляционными базами данных.
2. Программное обеспечение MySQL – это ПО с открытым кодом.
3. Внутренние характеристики и переносимость:
  - 3.1. Написан на C и C++. Протестирован на множестве различных компиляторов.
  - 3.2. Работает на различных аппаратных платформах и разных операционных системах.
  - 3.3. Высокая производительность за счет максимально оптимизированного кода, эффективной системы распределения памяти и продуманной системы дисковых таблиц.
4. Масштабируемость:
  - 4.1. Способность работать с очень большими базами данных (десятки и сотни миллионов записей).
  - 4.2. Возможность кластеризации серверов и распределения обработки информации между серверами
5. Технические возможности СУБД:
  - 5.1. MySQL – является системой клиент-сервер, которая содержит многопоточный SQL-сервер, обеспечивающий поддержку различных вычислительных машин баз данных, а также несколько различных клиентских программ и библиотек, средства администрирования и широкий спектр программных интерфейсов (API).

## 6. Безопасность:

6.1. Система безопасности основана на привилегиях и паролях с возможностью верификации с удаленного компьютера, за счет чего обеспечивается гибкость и безопасность. Пароли при передаче по сети при соединении с сервером шифруются.

## 3. Ход работы

### 3.1 Подготовка к выполнению работы

В первую очередь была произведена установка макета для лабораторной работы, представляющего собой виртуальную машину на ОС Windows 7 для VirtualBox.

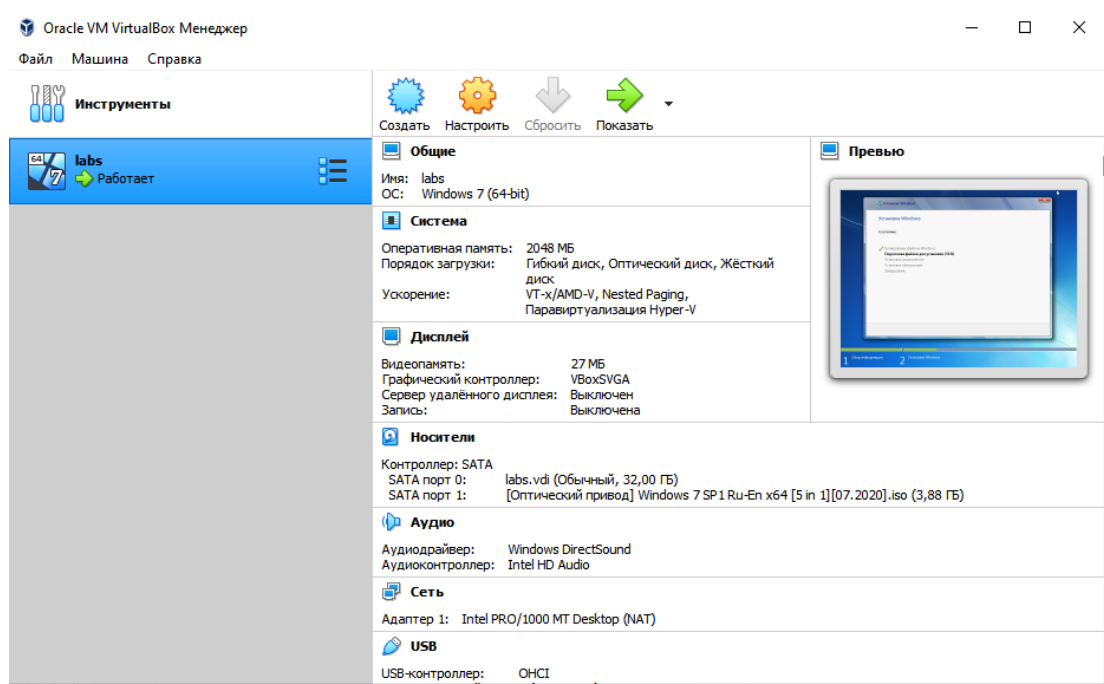


Рис. 1 – Макет лабораторной работы

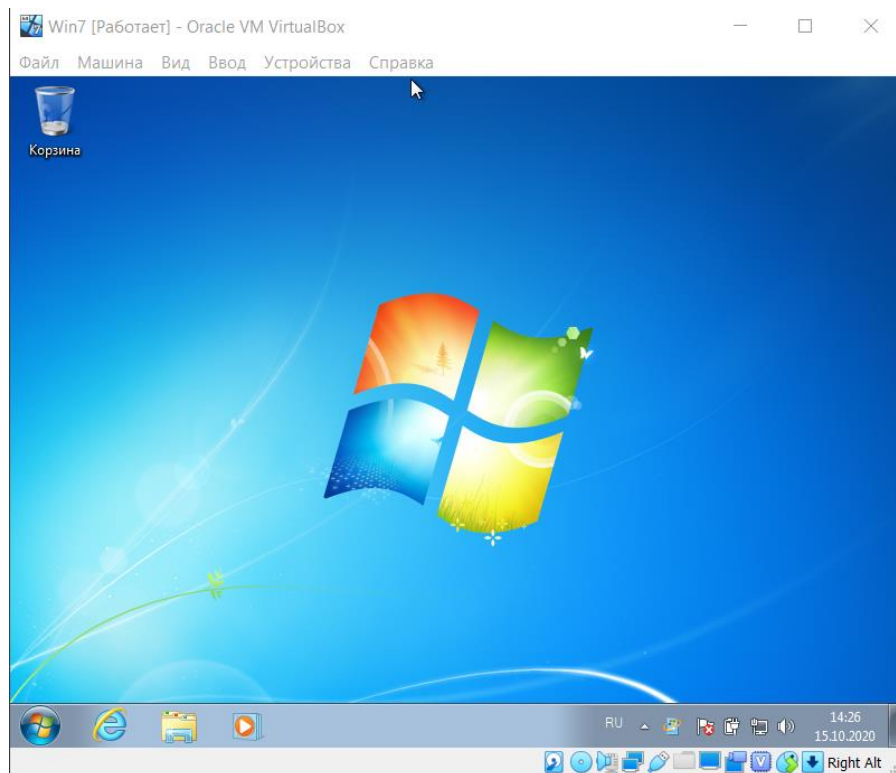


Рис. 2 – Рабочий стол

### 3.2 Установка MySQL

Для успешной установки необходимого ПО, учетная запись, под которой выполняются данные действия, должна входить в локальную группу «Администраторы» на локальном компьютере.

Чтобы удостовериться в этом: необходимо открыть «Учетные записи пользователей», перейдя по адресу: Пуск → Панель управления → Учетные записи пользователей и узнать, какие пользователи существуют на данном АРМ.

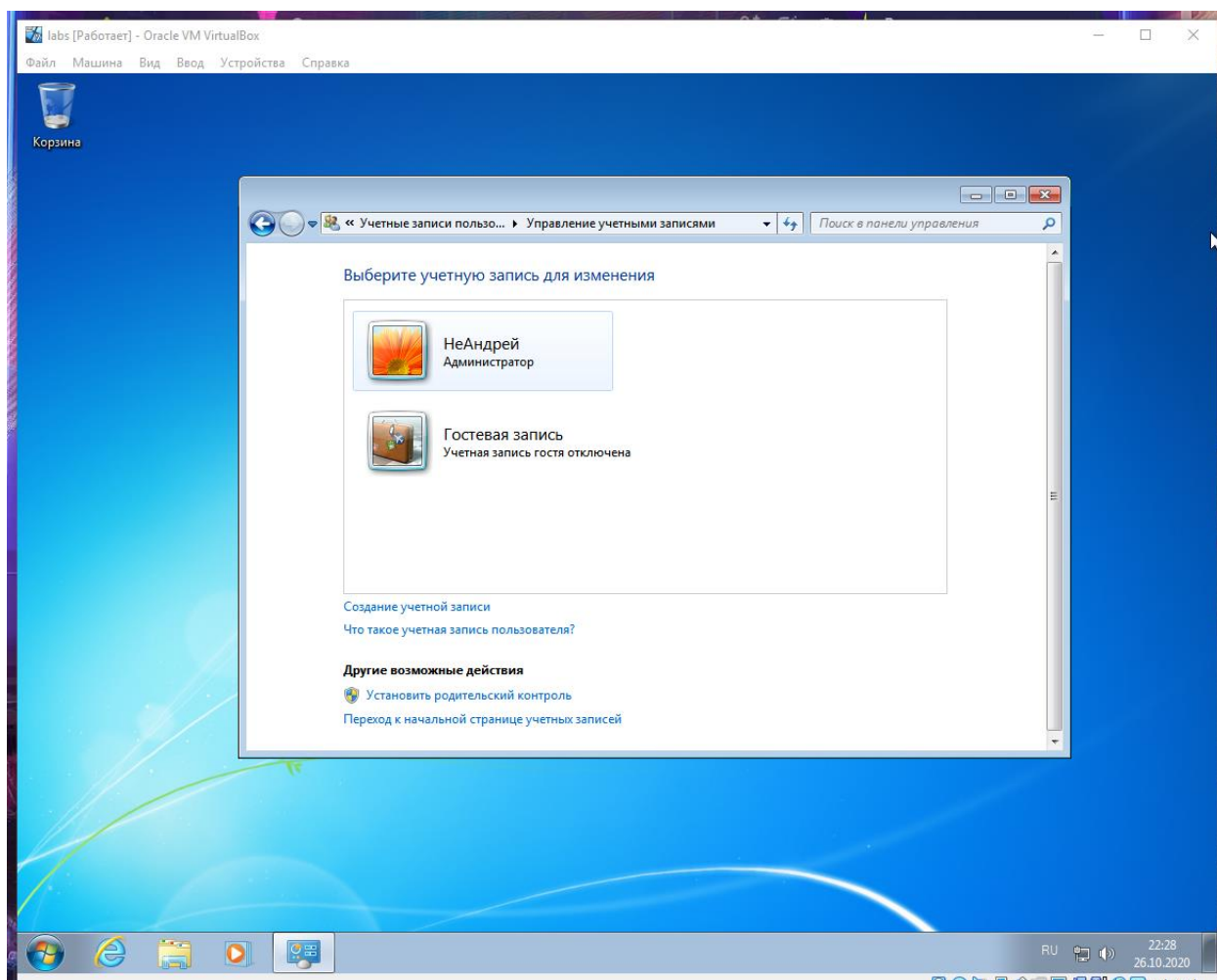


Рис. 3 – Учетная запись пользователя

Далее переходим непосредственно к установке ПО, скачанного с официального сайта [www.mysql.com](http://www.mysql.com).

Обязательно должны быть установлены:

- В группе «MySQL Server 5.6.11» - MySQL Server, Client Programs, Server Data Files.
- В группе Applications – MySQL Notifier.
- Группу MySQL Connectors – не устанавливать
- Группу Documentation – рекомендуется установить полностью.

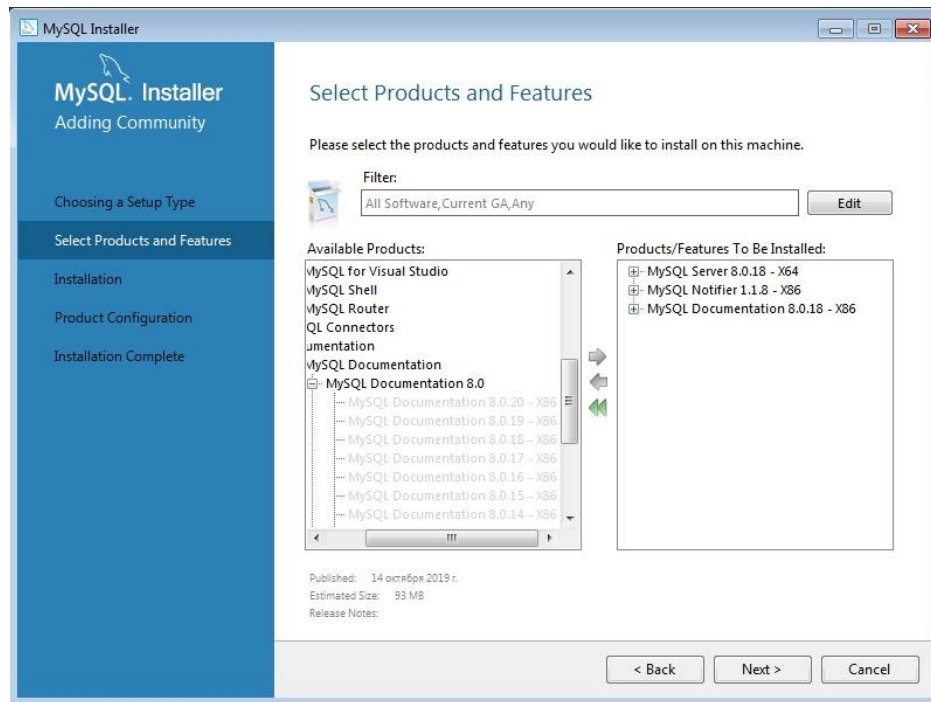


Рис. 4 – Выбор устанавливаемых продуктов

После окончания установки запустился Мастер настроек MySQL (он также доступен пользователю и после инсталляции).

В окне настроек выберем конфигурацию «Development Machine» – этот тип установки предназначен для разработки и тестирования сайтов, в этом случае ресурсы компьютера будут подвергаться минимальной нагрузке.

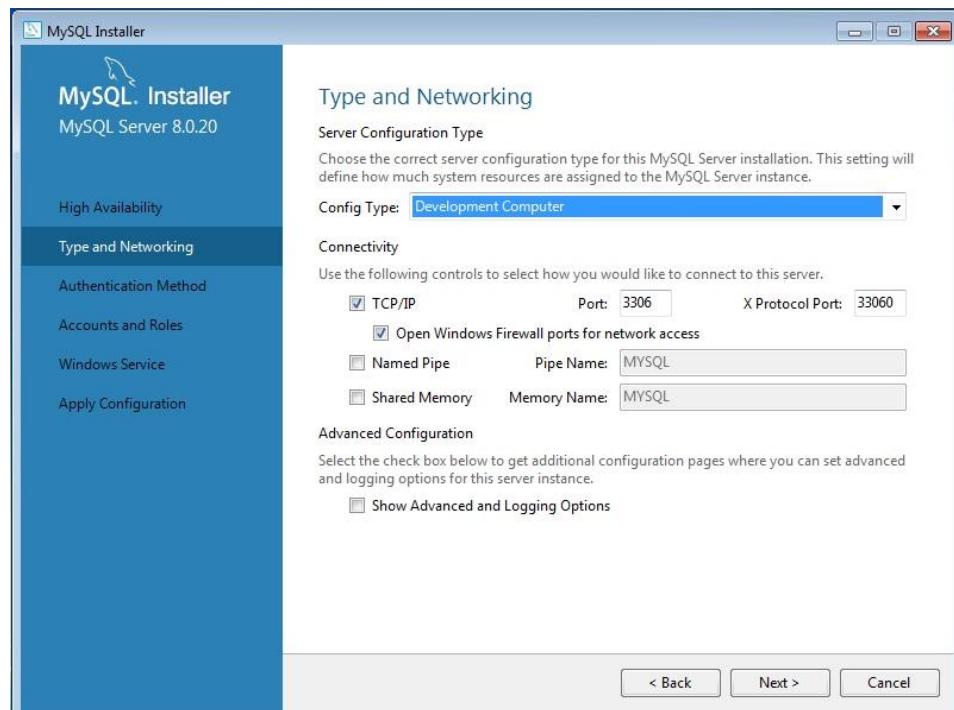


Рис. 5 – Окно настройки серверной части

Далее переходит к настройке пароля главного администратора сервера. Установим пароль, отвечающий следующему требованию к классу защищенности 2А:

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

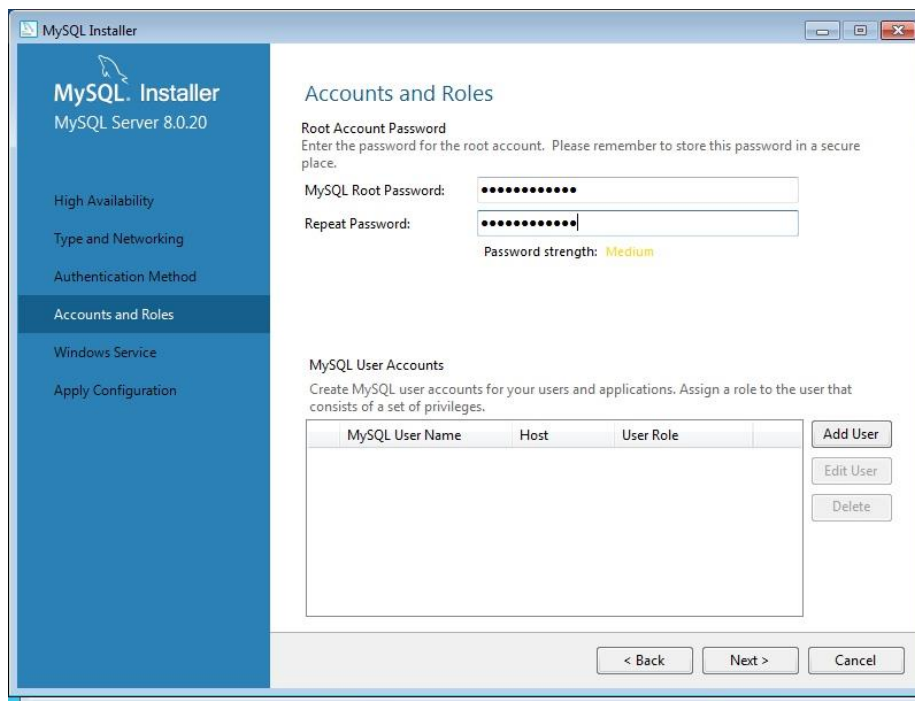


Рис. 6 – Установка пароля администратора сервера

Далее автоматическая настройка сервера в соответствии с заданной конфигурацией применит все ранее указанные настройки и автоматически применяет их на сервере. В результате выполнения показывает выполненные этапы и создает журнал применения изменений.



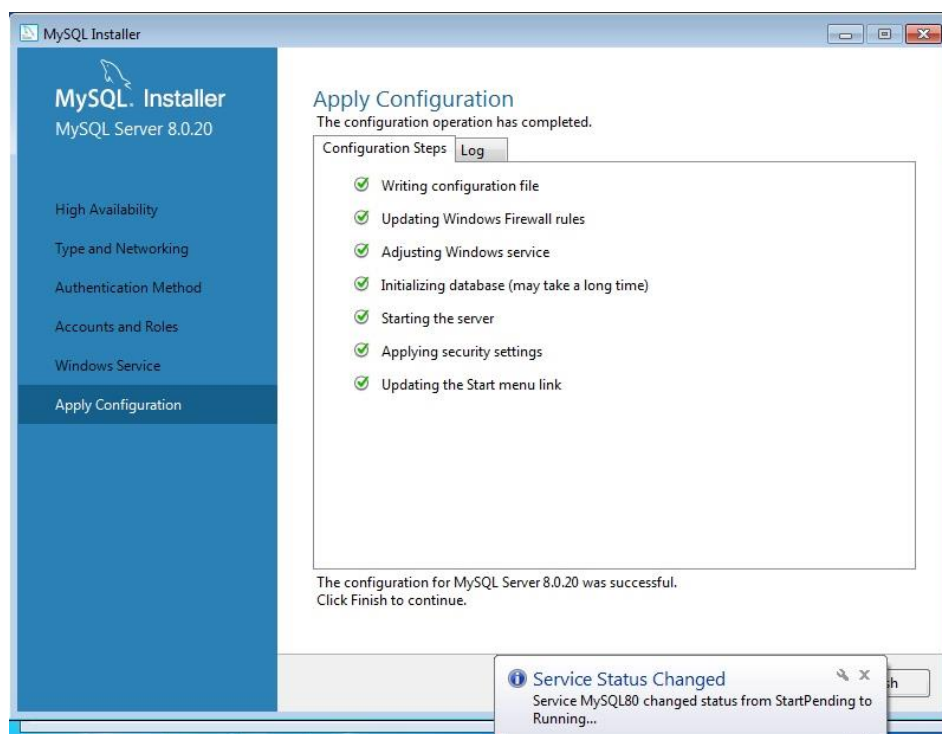


Рис. 7 – Успешное применение настроек

Для дальнейшего управления сервером используется утилита «MySQL Notifier» (вызывается из меню программ).

### 3.3 Создание базы данных MySQL

Для того, чтобы можно было использовать возможности сервера, такие как создание и назначение прав пользователям на таблицы и базы данных, требуется, создать пользовательскую базу данных и таблицу в ней. Имеющиеся по умолчанию базы данных можно посмотреть с использованием команды: Имеющиеся по умолчанию базы данных можно посмотреть с использованием команды *show databases*:

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

Рис. 8 – Имеющиеся по умолчанию базы данных

Очевидно, что изменение данных в системных таблицах нежелательно (например, в базе данных *mysql* находится таблица *user* с перечнем всех пользователей, их паролей и способов подключения к базам данных). Для того, чтобы иметь возможность добавлять, обрабатывать и удалять данные, целесообразно создание собственной БД.

Назовем ее *laboratoryWork3* и создадим с помощью следующей команды *create database laboratoryWork3*:

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

Рис. 9 – Создание собственной базы данных

Очевидно, база данных *laboratoryWork3* успешно создана. Укажем серверу, что далее мы будем работать именно с ней, используя следующую команду *use laboratoryWork3;* .

Создадим пользовательскую таблицу с названием «*phoneNumbers*» и с параметрами, заданными в соответствии с условиями лабораторной работы: *UserName* – тип данных *Text*, *UserAddress* – тип данных *Text*, *UserPhone* – тип данных *Text*. Также добавим дополнительную колонку, в которой пропишем автоматический счетчик записей *auto\_increment primary key* типа *Integer* – это значение будет увеличиваться с каждой новой записью и позволит более гибко оперировать содержимым таблицы.

Для этого используем команду:

1. *create table phoneNumbers (id integer auto increment primary key,*
2. *UserName text not null,*
3. *UserAddress text not null,*
4. *UserPhone text not null);*

```
mysql> create table phoneNumbers (id integer auto_increment primary key,
-> UserName text not null,
-> UserAddress text not null,
-> UserPhone text not null);
Query OK, 0 rows affected (0.12 sec)
```

Рис. 10 – Добавление таблицы в базу данных

Поскольку сейчас таблица пустая, заполним ее произвольными данными – в нашем примере это телефонная книга, поэтому добавим в нее записи, содержащие имя, адрес и телефонный номер абонентов с помощью команды *insert into phoneNumbers (UserName, UserAddress, UserPhone) values ('Tom', 'Alabaeva st.', '43', '777-88-99');* :

```
mysql> insert into phoneNumbers (UserName, UserAddress, UserPhone) values ('Tom',
-> 'Alabaeva st.', '43', '777-88-99')
-> ;
Query OK, 1 row affected (0.27 sec)

mysql> insert into phoneNumbers (UserName, UserAddress, UserPhone) values ('Arina',
-> 'Rapova st.', '459-96-22');
Query OK, 1 row affected (0.13 sec)
```

Рис. 11 – Добавление данных в созданную таблицу

Просмотрим содержимое таблицы. Это можно сделать командой *select \* from phoneNumbers;* :

```
mysql> select * from phoneNumbers;
+-----+-----+-----+-----+
| id | UserName | UserAddresss | UserPhone |
+-----+-----+-----+-----+
| 1 | Danya | Zverevoy st., 6 | 981-187-01-12 |
| 2 | Tanya | Deputatskaya st., 34 | 952-235-39-50 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

Рис. 12 – Просмотр содержимого созданной таблицы

### 3.4 Создание нового пользователя и настройка его прав

Создадим локального, т.е. существующего на текущем АРМ, пользователя с именем LocalUser и установим ему пароль Pa\$sw0rd, таким, чтобы он соответствовал требованиям «Руководящего документа» для АРМ «1Г» класса.

Для этого выполним следующую команду:

```
1. create user 'LocalUser'@'localhost' identified by 'Pa$sw0rd';
```

```
mysql> create user 'LocalUser'@'localhost' identified by 'Pa$sw0rd';
Query OK, 0 rows affected (0.02 sec)
```

Рис. 13 – Создание нового пользователя

```
1. grant all privileges on lab3.* to 'LocalUser'@'localhost';
```

```
mysql> grant all privileges on lab3.* to 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.01 sec)
```

Рис. 14 – Предоставление новому пользователю неограниченных прав

Теперь проверим, применились ли права, просмотрев их для пользователя с помощью команды *show grants for 'LocalUser'@'localhost'*:

```
mysql> grant all privileges on laboratorywork3.* to 'LocalUser'@'localhost';
Query OK, 0 rows affected (4.25 sec)

mysql> show grants for 'LocalUser'@'localhost';
+-----+-----+-----+-----+
| Grants for LocalUser@localhost |
+-----+-----+-----+-----+
| GRANT USAGE ON *.* TO 'LocalUser'@'localhost' |
| GRANT ALL PRIVILEGES ON 'laboratorywork3'.* TO 'LocalUser'@'localhost' |
+-----+-----+-----+-----+
2 rows in set (0.46 sec)
```

Рис. 15 – Права нового пользователя

Теперь попробуем подключиться к серверу MySQL с помощью утилиты «MySQL Workbench». Дополнительно установим ее с помощью той же программы, которой мы устанавливали «MySQL Server»:

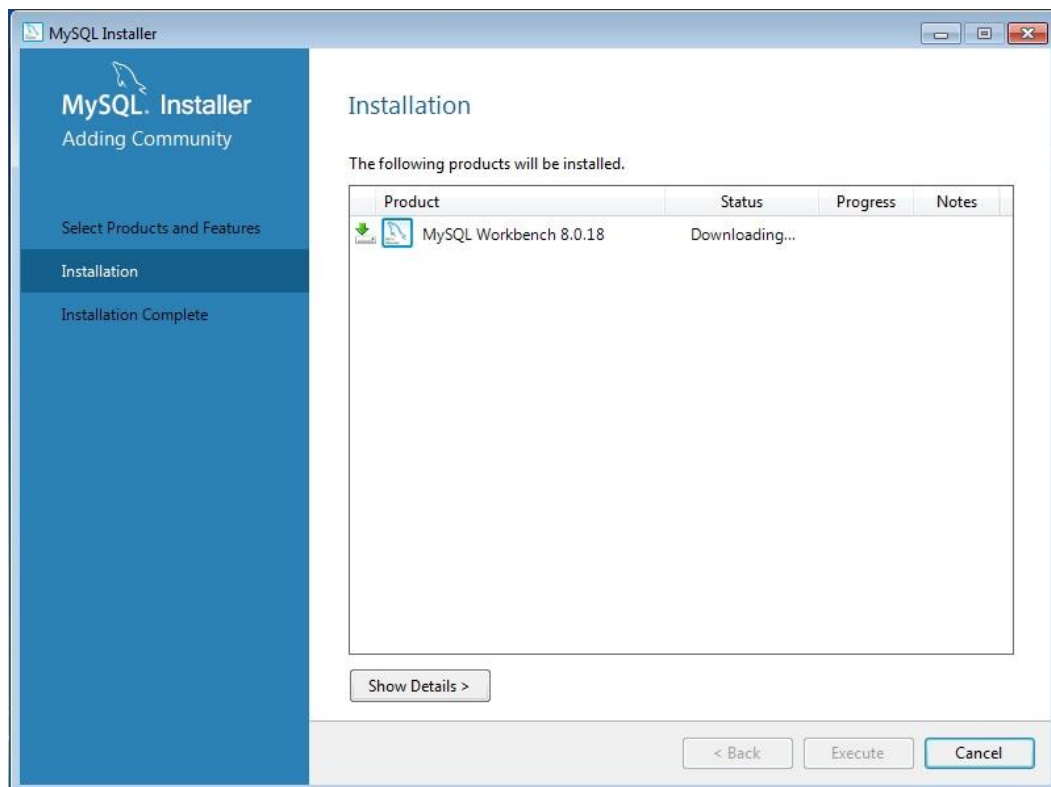


Рис. 16 – Установка утилиты MySQL Workbench

Чтобы запустить командную строку под новым пользователем необходимо зайти в интерфейс программы «MySQL Workbench» и создать новое подключение, вызвав *Manage Server Connections*, для которого настроить следующие параметры, а остальные оставить по умолчанию:

- Connection Name: *LocalUser*
- Password: *password*

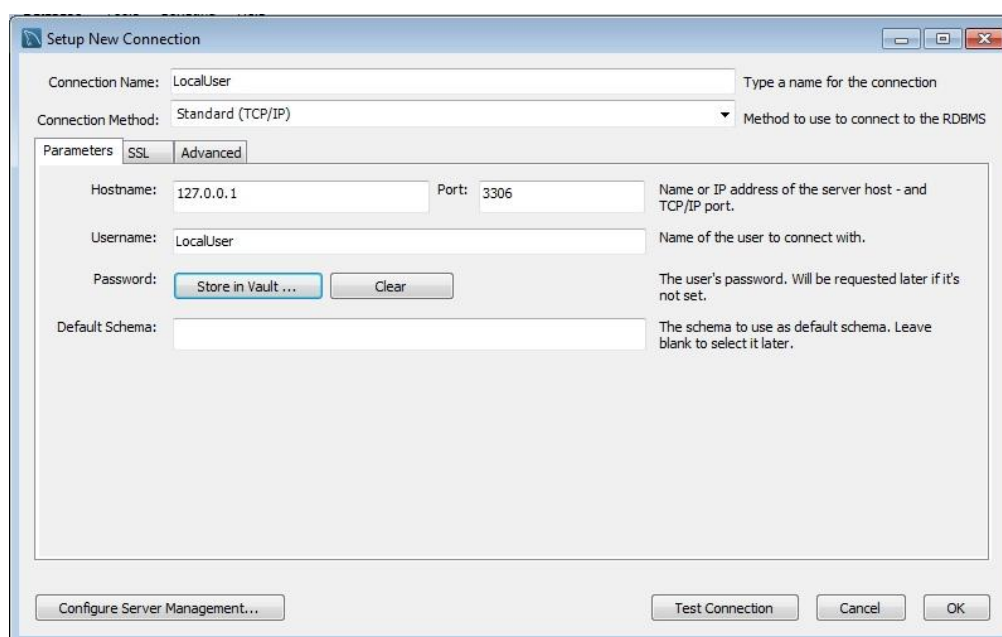
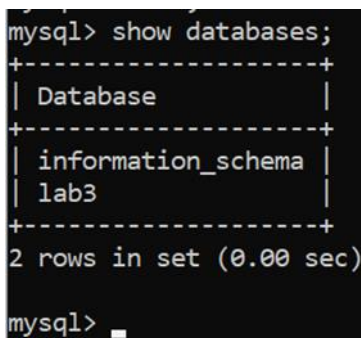


Рис. 17 – Настройка подключения для нового пользователя

После чего настройка нового соединения будет успешно завершена, а в утилите «MySQL Workbench» появится новое доступное соединение за нашего созданного не привилегированного пользователя.

Попробуем подключиться к серверу с использованием этого нового соединения, выполнив следующие действия: *Кликнуть на подключение LocalUser правой кнопкой мыши → Start Command Line Client*:



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

mysql> _
```

Рис. 18 – Подключение к серверу от лица нового пользователя

MySQL позволяет назначать права доступа с помощью следующей команды:

`GRANT [тип прав] ON [имя базы данных].[имя таблицы] TO 'имя пользователя'@'тип доступа на сервер';`

Нужно заменить значение «тип прав» на тот вид прав доступа, который необходимо предоставить новому пользователю, кроме того, необходимо указать базу данных и имена таблиц, доступ к которым предоставляется.

В MySQL есть несколько типов прав доступа, опишем некоторые из них:

**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.

С целью обеспечения безопасности информации, содержащейся в других БД, а также обеспечения принципа разделения доступа к защищаемой информации, разрешим пользователю только просмотр записей остальных баз данных, имеющихся в файловой системе SQL.

Для этого выполним следующую команду:

```
1. grant SELECT on *.* to 'LocalUser'@'localhost';
```

```
mysql> grant SELECT on *.* to 'LocalUser'@'localhost';
ERROR 2006 (HY000): MySQL server has gone away
No connection. Trying to reconnect...
Connection id: 12
Current database: laboratorywork3
Query OK, 0 rows affected (7.37 sec)
```

Рис. 19 – Предоставление пользователю прав

Установим права **SELECT** и **INSERT** на базу данных *laboratoryWork3* и непосредственно для таблицы *phoneNumbers* для пользователя *LocalUser* следующими командами:

- *grant SELECT, INSERT on laboratoryWork3.\* to 'LocalUser'@'localhost';*
- *grant SELECT, INSERT on laboratoryWork3.phoneNumbers to 'LocalUser'@'localhost';*

```
mysql> grant SELECT, INSERT on laboratoryWork3.* to 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.13 sec)

mysql> grant SELECT, INSERT on laboratoryWork3.phoneNumbers to 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.08 sec)
```

Рис. 20 – Предоставление пользователю прав

Теперь проверим, применились ли права, просмотрев их для пользователя с помощью команды *show grants for 'LocalUser'@'localhost';* :

```
mysql> show grants for 'LocalUser'@'localhost';
+-----+
| Grants for LocalUser@localhost |
+-----+
| GRANT SELECT ON *.* TO 'LocalUser'@'localhost' |
| GRANT SELECT, INSERT ON 'laboratorywork3'.* TO 'LocalUser'@'localhost' |
| GRANT SELECT, INSERT ON 'laboratorywork3'.'phonenumbers' TO 'LocalUser'@'localhost' |
+-----+
3 rows in set (0.05 sec)
```

Рис. 21 – Просмотр прав пользователя

### *Запрос выборок из таблицы*

Проверим корректность настроенных прав для пользователя «LocalUser», а также ознакомимся с механизмом запроса записей из таблицы в SQL.

Сделаем несколько выборок из таблицы «phoneNumbers»:

1. Выборка из таблицы «phoneNumbers» значений адреса и телефона для пользователей 'Danya' и 'Tanya' с помощью следующих команд

Проверим корректность настроенных прав для пользователя *LocalUser*. Сделаем несколько выборок из таблицы *phoneNumbers*.

- *select UserAddress, UserPhone from phoneNumbers where UserName= 'Tom';*
- *select UserAddress, UserPhone from phoneNumbers where UserName= 'Arina';*



```
mysql> select UserAddresss, UserPhone from phoneNumbers where UserName='Danya';
+-----+-----+
| UserAddresss | UserPhone |
+-----+-----+
| Zverevoy st., 6 | 981-187-01-12 |
+-----+-----+
1 row in set (0.00 sec)

mysql> select UserAddresss, UserPhone from phoneNumbers where UserName='Tanya';
+-----+-----+
| UserAddresss | UserPhone |
+-----+-----+
| Deputatskaya st., 34 | 952-235-39-50 |
+-----+-----+
1 row in set (0.00 sec)
```

Рис. 22 – Результаты выборки из таблицы телефонных номеров

Выборка всех записей из таблицы *phoneNumbers* с сортировкой по полю *UserName* в алфавитном порядке с использованием команды *select \* from phoneNumbers order by UserName asc;* :

```
mysql> select * from phoneNumbers order by UserName asc;
+----+-----+-----+-----+
| id | UserName | UserAddresss | UserPhone |
+----+-----+-----+-----+
| 3 | Alex | Zhykhova st., 24 | 999-521-64-09 |
| 1 | Danya | Zverevoy st., 6 | 981-187-01-12 |
| 2 | Tanya | Deputatskaya st., 34 | 952-235-39-50 |
| 4 | Yana | Parfenovskaya st., 5 | 911-753-55-11 |
+----+-----+-----+-----+
4 rows in set (0.01 sec)
```

Рис. 23 – Результаты выборки из таблицы телефонных номеров

Очевидно, что запросы пользователя *LocalUser* к таблице *phoneNumbers* успешно выполняются.

Проверим, корректно ли работает механизм задания прав пользователя, попробовав удалить таблицу *phoneNumbers* с использованием команды *drop table phoneNumbers;* :

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

Рис. 24 – Отказ пользователю при попытке удалить таблицу

### 3.5 Удаление таблиц, базы данных и пользователей

Рассмотрим механизм удаления таблиц, баз данных и пользователей из файловой системы SQL. Это можно сделать только привилегированным-root пользователем, поэтому дальнейшая работа будет осуществляться из его терминала.

Удалим таблицу *phoneNumbers* с помощью команды *drop table phoneNumbers;* :

```
mysql> drop table phoneNumbers;
Query OK, 0 rows affected (0.30 sec)

mysql>
```

Рис. 25 – Удаление таблицы

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

mysql>
```

Рис. 26 – Результат удаление таблицы

Удалим базу данных *laboratoryWork3* с помощью команды *drop database laboratoryWork3; :*

```
mysql> drop database laboratorywork3;
Query OK, 0 rows affected (0.13 sec)

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

Рис. 27 – Удаление базы данных

Удалим все права доступа пользователя *LocalUser* с помощью команды *revoke all privileges on \*.\* from 'LocalUser'@'localhost'; :*

```
mysql> revoke all privileges on *.* from 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.06 sec)

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

Рис. 28 – Удаление всех прав созданного нами пользователя

Посмотрим список всех пользователей до удаления *LocalUser*, обратившись к таблице *mysql.user* и запросив из нее всех пользователей и их способ подключения к серверу с помощью команды *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)
```

Рис. 29 – Список всех имеющихся пользователей

Удалим пользователя *LocalUser* с помощью команды *drop user 'LocalUser'@'localhost'; :*



```
mysql> drop user 'LocalUser'@'localhost';
Query OK, 0 rows affected (0.10 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)
```

Рис. 30 – Удаление пользователя

Очевидно, что удаление баз данных, пользователей и таблиц успешно осуществлено.

## Выводы

В ходе выполнения лабораторной работы были выполнены установка MySQL на макет для лабораторной работы, представляющего собой виртуальную машину на ОС Windows 7, и настройка собственной базы данных в MySQL.

Была создана собственная база данных с таблицей, хранящей информацию о телефонных номерах сотрудников. Для работы с этой базой данных был создан новый пользователь, права которого были ограничены просмотром содержимого таблицы и добавлением в нее новых записей. Так же выполнили проверку корректности работы механизма задания прав пользователей.