МИНИСТЕРСТВО СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУГРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ им. проф. М.А. БОНЧ-БРУЕВИЧА»

Факультет Информационных систем и технологий
Кафедра Информационных управляющих систем

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

проф., д.т.н.		Н.Н. Мошак
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ЛАБОРАТОРНАЯ РАБОТА № 3

«АДМИНИСТРИРОВАНИЕ И НАСТРОЙКА ПОЛИТИКИ БЕЗОПАСНОСТИ СЕРВЕРА РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ MySQL»

по курсу: Безопасность информационных технологий и систем

РАБОТУ ВЫПОЛНИЛ(А)

СТУДЕНТ(КА) ГР. _____ _____ _____

подпись, дата | инициалы, фамилия

Санкт-Петербург

2020

Цель – изучить команды MySQL и научиться устанавливать, администрировать SQL-сервер на примере сервера MySQL и настраивать его параметры безопасности

Используемое программное обеспечение: операционная система Windows XP (или любой другой), ПО сервера MySQL.

2. Основные сведения и рекомендации по выполнению лабораторной работы

2.1. 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:

- 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 Standart 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 корпорцией 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 TcX 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 C и C++. 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.

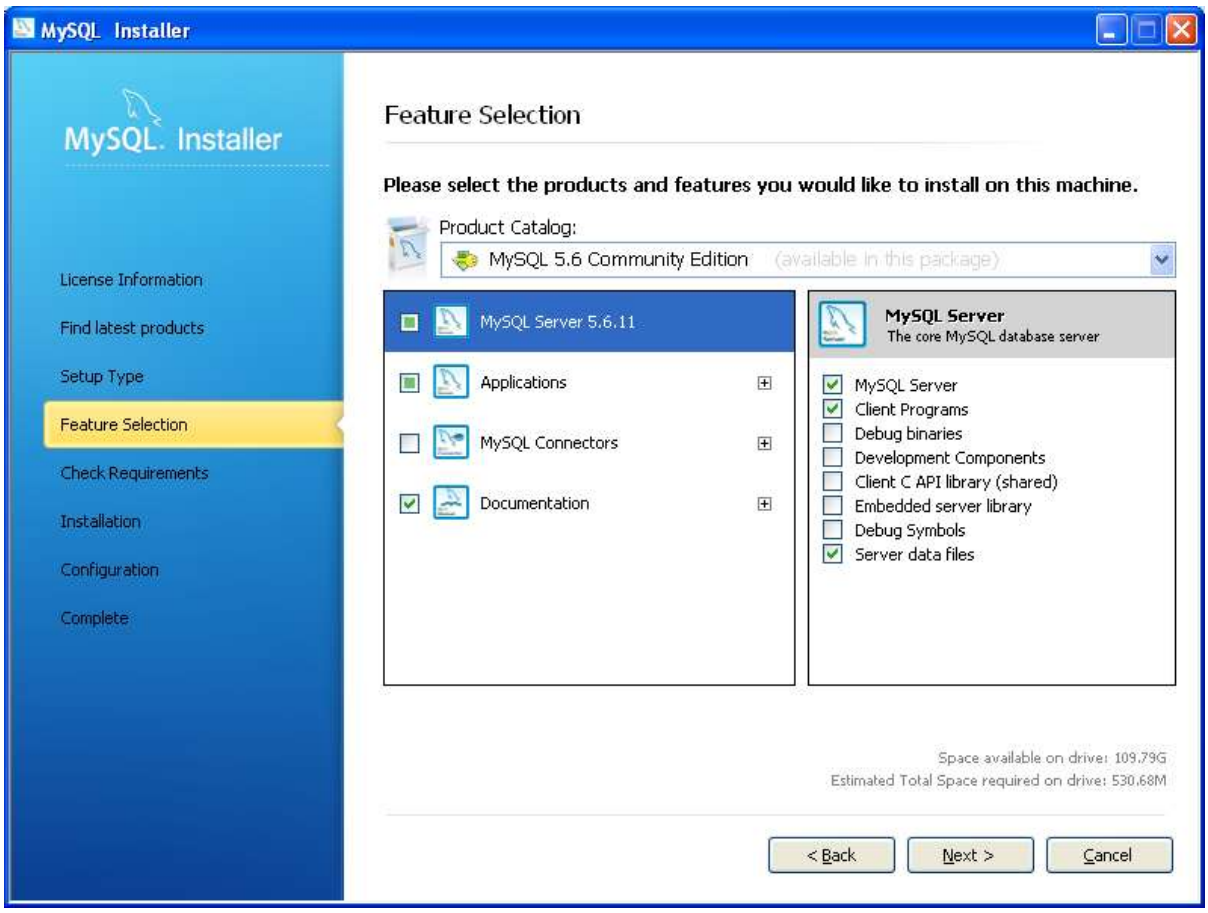
- Масштабируемость
  - Способность работать с очень большими базами данных (десятки и сотни миллионов записей).
  - Возможность кластеризации серверов и распределения обработки информации между серверами

2.3. Установка MySQL на платформу ОС Windows

Установка MySQL возможна на ОС Windows 2000 и выше. Дистрибутив можно скачать с www.mysql.com. Онлайн-документация (на английском языке) расположена по адресу http://dev.mysql.com/doc/

Для установки необходимо иметь административные права. Перед установкой необходимо установить Microsoft. Net Framework 4.0, использующийся в программах администрирования в среде Microsoft Windows. Для других операционных систем используются другие методы администрирования.

Непосредственно процесс установки производится аналогично другим программам в Windows, поэтому здесь не рассматривается. При выборе типа устанавливаемого сервера рекомендуется выбрать пользовательскую установку (допускается выбрать полную, но будут установлены неиспользуемые компоненты и потребуется дополнительно установить библиотеку Visual C++ 2010 Runtime, не входящую в дистрибутив).



Обязательно должны быть установлены:

В группе «MySQL Server 5.6.11» - MySQL Server, Client Programs, Server Data Files.

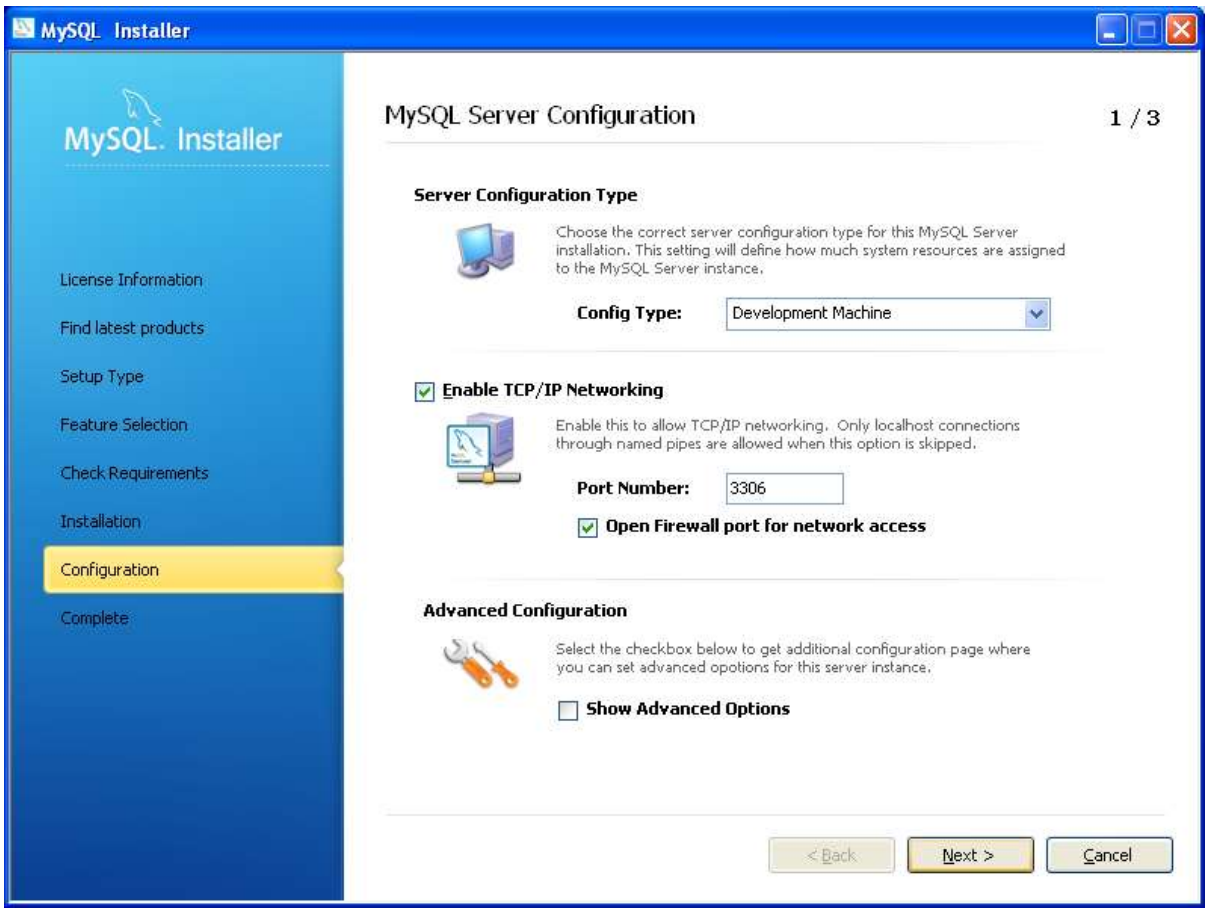
В группе Applications – MySQL Notifier.

Группу MySQL Connectors – не устанавливать

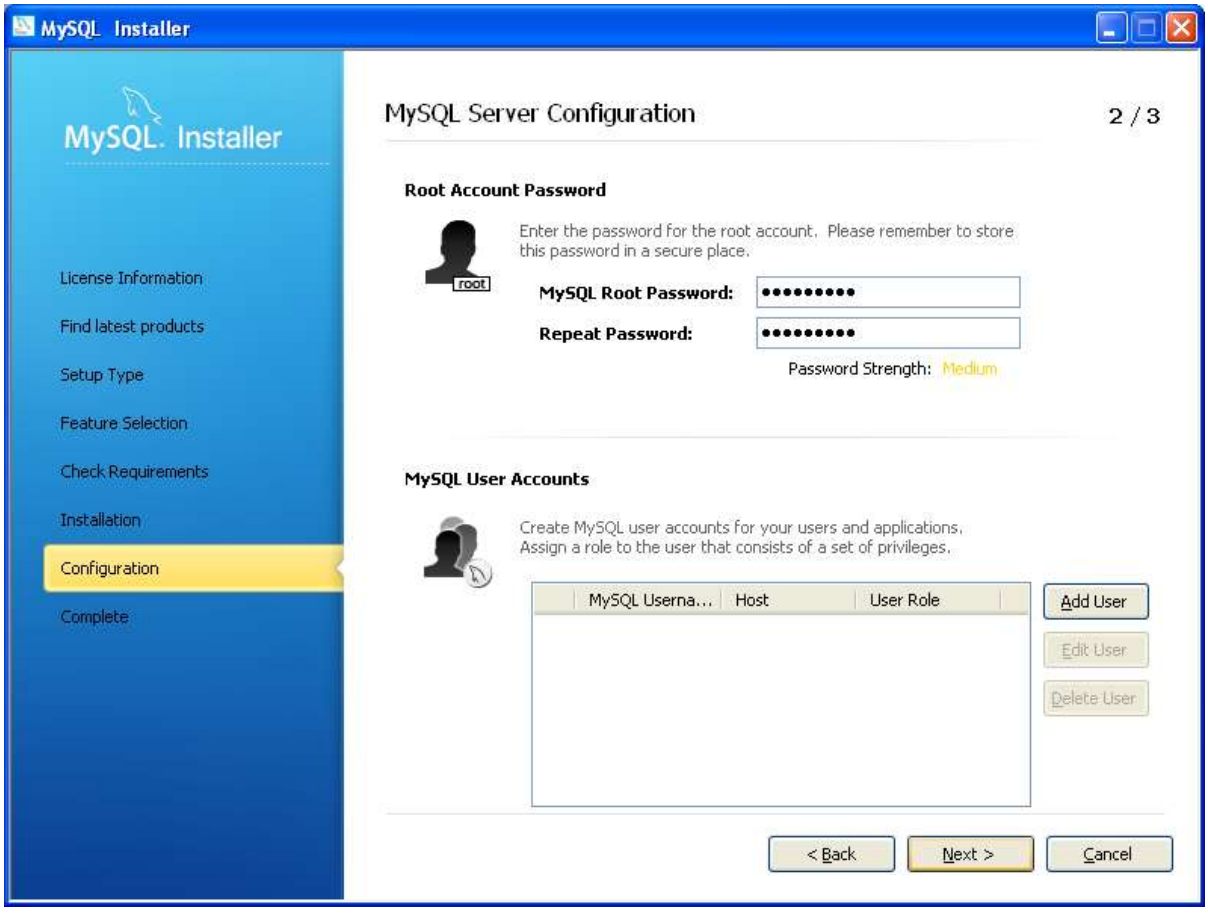
Группу Documentation – рекомендуется установить полностью.

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

1. Окно настройки серверной части. Рекомендуется выбрать конфигурацию «Development Machine».

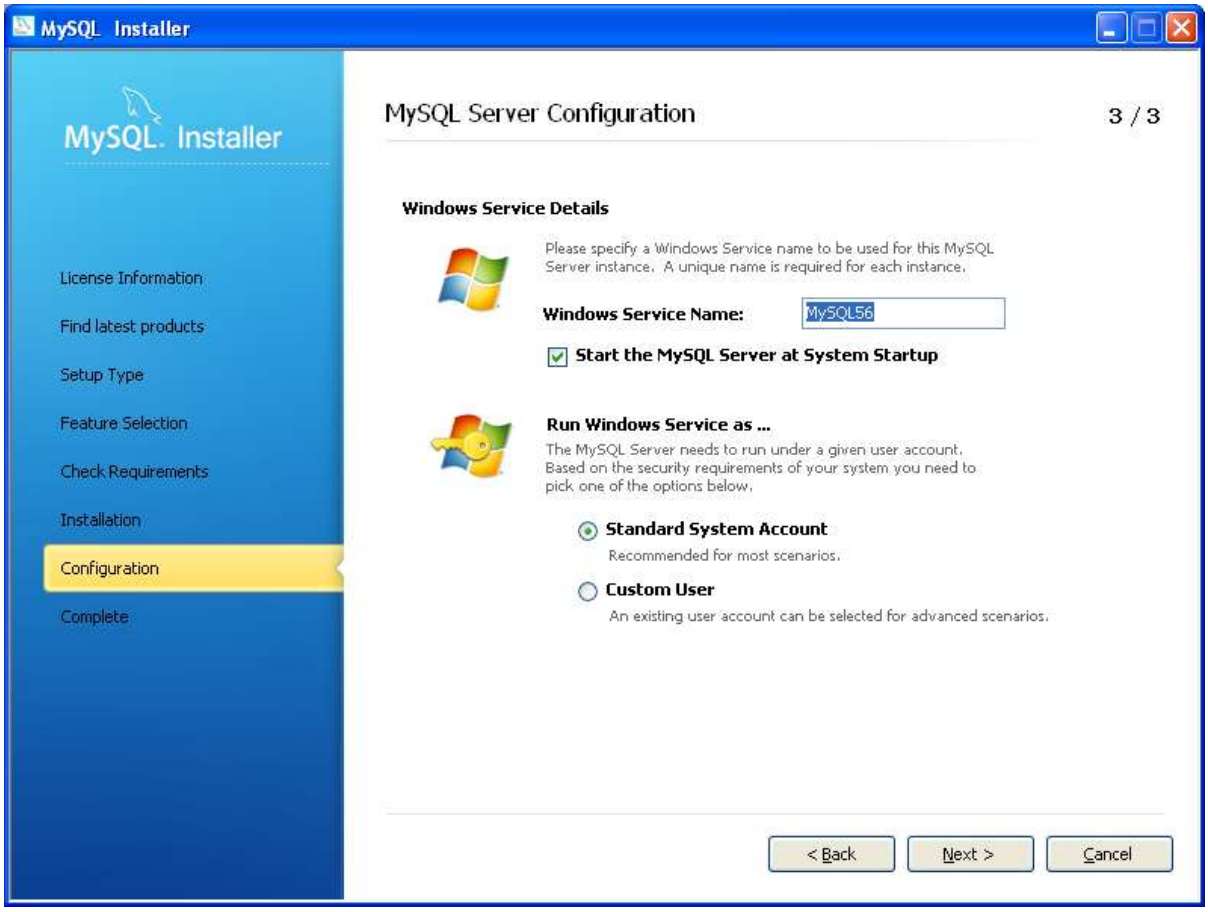


2. Настройка пароля главного администратора сервера (учетная запись – root)



Также имеется возможность добавить дополнительных пользователей, установив им необходимый уровень доступа.

3. Окно настройка запуска сервиса можно оставить без изменений.



На следующем этапе производится автоматическая настройка сервера в соответствии с заданной конфигурацией и его запуск..

Для управления сервером используется утилита «MySQL Notifier» (вызывается из меню программ). Утилита выводит иконку в панели задач, являющуюся индикатором состояния сервера базы данных, а также позволяющую управлять сервером (запуск и останов сервера).

2. 4. Настройка параметров безопасности

Основой системы безопасности является система привилегий (privilege system), позволяющая очень гибко управлять правами доступа как к управлению сервером, так и к отдельным базам, таблицам и полям таблиц, а также встроенным функциям и хранимым процедурам. Для изучения всей системы привилегий рекомендуется обратиться к документации на программу, здесь же рассмотрим основные моменты.

Настройку параметров доступа можно производить из клиента базы данных – программы mysql.exe (находится в каталоге установки продукта в подкаталоге *bin*). При запуске программе можно указать много параметров, но для изучения достаточно использовать синтаксис:

mysql.exe –u <имя пользователя> -p <база данных>,

-u – флаг, за которым через пробел указывается имя пользователя

-p – флаг, указывающий на необходимость запроса пароля

<база данных> - имя базы, с таблицами которой будет проводиться работа. Этот параметр не является обязательным, т.к. из клиента в любой момент можно переключиться на работу с другой базой с использованием команды «use <база данных>».

Внимание! Все команды, вводимые в клиенте, обязательно должны заканчиваться точкой с запятой.

Все параметры безопасности MySQL хранятся в виде таблиц системной базы данных «mysql», поэтому первый запуск клиента рекомендуется осуществить командой

mysql.exe –u root –p mysql (пароль был установлен в процессе инсталляции)

Создадим нового пользователя с заданным паролем:

CREATE USER 'имя пользователя'@'localhost' IDENTIFIED BY 'пароль';

В данном примере localhost означает, что пользователю будет доступен только локальный вход.

Предоставим пользователю полные права на все таблицы созданной при инсталляции программы базу данных test:

GRANT ALL PRIVILEGES ON test.* TO 'имя пользователя'@'localhost';

В данном примере *test.** означает все таблицы базы test. При необходимости можно предоставить права на отдельную таблицу (например, на таблицу table), указав *test.table*.

Для проверки прав доступа необходимо выполнить команду:

SHOW GRANTS FOR 'имя пользователя'@'localhost';

Удаление прав пользователя производится, например, следующим образом:

REVOKE ALL PRIVILEGES ON test.* FROM 'имя пользователя'@'localhost';

Выход из программы клиента осуществляется командой QUIT (можно без точки с запятой на конце).

Если теперь попробовать соединиться от имени созданного пользователя с базой mysql, то соединение будет отвергнуто сервером. Соединение же с базой test будет успешно установлено.

2.5. Краткий обзор команд MySQL

Создание базы данных выполняется с помощью команды **CREATE DATABASE**.

Синтаксис команды:

CREATE DATABASE *database_name*

- *database_name* - Имя, которое будет присвоено создаваемой базе данных.

Для **удаления** базы данных используется команда **DROP DATABASE**.

Синтаксис:

DROP DATABASE *database_name*

где

- *database_name* - задает имя базы данных, которую необходимо удалить.

Создание таблицы производится командой **CREATE TABLE**.

CREATE TABLE table_name(column_name1 *type*, column_name2 *type*,...)

- table_name - имя новой таблицы;
- column_name - имена колонок (полей), которые будут присутствовать в создаваемой таблице.
- *type* - определяет тип данных создаваемой колонки.

Например, надо создать таблицу учетных записей и паролей с именем user_pass. Пусть таблица будет состоять из двух столбцов – UserName и UserPassword с типом данных text:

```
CREATE TABLE user_pass(UserName text, UserPassword text);
```

Удаление таблицы производится командой **DROP TABLE**

DROP TABLE *table_name*

- *table_name* - имя удаляемой таблицы.

Вставка записи осуществляется командой **INSERT INTO**

INSERT INTO *table_name*(field_name1, field_name2,...) values('content1', 'content2',...)

Данная команда добавляет в таблицу *table_name* запись, у которой поля, обозначеные как field_nameN, установлены в значение contentN.

Например, в таблицу учетных записей и паролей можно добавить запись следующим образом:

```
INSERT INTO user_pass(UserName, UserPassword)
        values('Operator', '1qAz_weR');
```

Поиск записей осуществляется командой **SELECT**

```
SELECT * FROM table_name WHERE (выражение) [order by field_name [desc][asc]]
```

Эта команда ищет все записи в таблице *table_name*, которые удовлетворяют выражению *выражение*.

Если записей несколько, то при указаном предложении order by они будут отсортированы по тому полю, имя которого записывается правее этого ключевого слова (если задано слово desc, то упорядочивание происходит в обратном порядке). В предложении order by могут также задаваться несколько полей. Особое значение имеет символ *. Он предписывает, что из отобранных записей следует извлечь все поля, когда будет выполнена команда получения выборки. С другой стороны, вместо звездочки можно через запятую непосредственно перечислить имена полей, которые требуют извлечения.

Например, выбрать все записи из таблицы можно следующим образом:

```
SELECT * FROM user_pass;
```

Выбрать все записи с сортировкой по имени в обратном алфавитном порядке:

```
SELECT * FROM user_pass ORDER BY UserName desc;
```

Выбрать пароль для пользователя Admin:

```
SELECT UserPassword FROM user_pass WHERE UserName=’Admin’;
```

3. Содержание работы.

1. Установить сервер MySQL (на любую операционную систему).
2. Создать новую базу данных (с произвольным именем).
3. Создать пользователя MySQL с полным доступом к созданной на шаге 2 базе только с локального хоста. Осуществить соединение с сервером от имени созданного пользователя. Дальнейшая работа (шаги 4-7) производится от имени данного пользователя.
4. Создать (как пример) таблицу телефонного справочника (с произвольным именем) в созданной на шаге 2 базы, имеющей следующие колонки:

   UserName – тип данных Text

   UserAddress – тип данных Text

UserPhone – тип данных Text

5. Заполнить таблицу произвольными значениями (5-6 записей).
6. Настроить права пользователя по доступу к управлению сервером и к базе данных, отдельным таблицам и полям таблиц, а также встроенным функциям и хранимым процедурам.
7. Сделать выборку из таблицы значения адреса и телефона для указанного UserName (любое из имен, введенных на шаге 5).
8. Сделать выборку из таблицы с сортировкой по полю UserName в алфавитном порядке.
9. Соединиться с сервером от имени администратора (root) и удалить созданных пользователя, таблицу и базу данных.

4. Содержание отчета.

4.1. Цель работы

4.2. Основные сведения по серверу MySQL

4.3. Описать последовательность выполнения лабораторной работы (установка севера MySQL ,создание новой базы данных (с произвольным именем), создание пользователя MySQL с полным доступом к созданной на шаге 2 базе только с локального хоста; соединение с сервером от имени созданного пользователя; От имени данного пользователя: создание (как пример) таблицы телефонного справочника (с произвольным именем) в созданной на шаге 2 базы, имеющей следующие колонки: UserName – тип данных Text, UserAddress – тип данных Text, UserPhone – тип данных Text с заполнением таблицы произвольными значениями (5-6 записей); настройка прав пользователя по доступу к управлению сервером, базе данных и к отдельным таблицам и полям таблиц, а также встроенным функциям и хранимым процедурам; выборка из таблицы значения адреса и телефона для указанного UserName (любое из имен, введенных на шаге 5); выборка из таблицы с сортировкой по полю UserName в алфавитном порядке; соединение с сервером от имени администратора (root) и удаление созданных пользователя, таблицу и базу данных.

Указать последовательность команд SQL, используемая при выполнении задания, с комментариями по каждой команде, а также ответы сервера.

4.4. Настроить права пользователя по доступу к управлению сервером и к отдельным таблицам и полям таблиц, а также встроенным функциям и хранимым процедурам.

4.5. Обосновать настройки каждого параметра безопасности и сформулировать выводы о значимости настроенной политики для защиты базы данных от НСД

4.6. Сформулировать общий вывод по настройкам параметров безопасности базы данных от НСД.

ВЫВОДЫ