МИНОБРНАУКИ РОССИИ

ФГБОУ ВО

«МИРЭА - Российский технологический университет»

КАФЕДРА ИНСТРУМЕНТАЛЬНОГО И ПРИКЛАДНОГО

ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ (ИППО)



ЛАБОРАТОРНАЯ РАБОТА № 1

ПО ДИСЦИПЛИНЕ «Проектирование и разработка баз данных»

«СОЗДАНИЕ БАЗЫ ДАННЫХ В СРЕДЕ СУБД MYSQL»

Вариант 3

Выполнил

Валяев Данила Андреевич

группа ИКБО-08-18

Преподаватель

Матчин Василий Тимофеевич

Москва 2020

**Цель работы**

Получить навыки работы с СУБД MySQL.

**Теоретическое введение**

Базой данных называют структурированный набор данных. Это может быть чем угодно: от простого перечня покупок до галереи изображений. Чтобы добавлять, обращаться и обрабатывать данные, сохраненные в компьютерной базе данных, Вы нуждаетесь в системе управления базы данных, типа MySQL. Так как компьютеры очень хороши при обработке больших количеств данных, базы данных играют центральную роль в вычислениях, как автономные утилиты, или как части других пакетов прикладных программ.

MySQL является реляционной СУБД. Реляционная база данных сохраняет данные в отдельных таблицах. Это добавляет быстродействие и гибкость. Таблицы связаны определенными отношениями, делающими возможным объединить данные из нескольких таблиц в одном запросе. SQL-часть MySQL ориентирована на Structured Query Language, наиболее общий стандартизированный язык, используемый, чтобы обратиться к компьютерным базам данных.

MySQL является Open Source Software, то есть тексты открыты для чтения и правки всем желающим. Любой может скачать MySQL из Интернета и использовать его совершенно бесплатно. Любой желающий может изучать исходный текст и изменять его по своему усмотрению.

Следующий перечень описывает наиболее важные возможности MySQL:

* Интерфейсы для языков C, C++, Eiffel, Java, Perl, PHP, Python и Tcl.
* Работает на многих различных платформах.
* Много типов столбцов: целые со знаком или без него длиной 1, 2, 3, 4 и 8 байт, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET и ENUM.
* Очень быстрые объединения, использующие оптимизированное однопроходное объединение многих таблиц.
* SQL-функции выполнены через хорошо оптимизированную библиотеку классов. Обычно не имеется никакого распределения памяти после инициализации запроса.
* Полная поддержка предложений SQL GROUP BY и ORDER BY. Поддержка групповых функций: COUNT(), COUNT(DISTINCT...),  AVG(), STD(),  SUM(), MAX() и MIN().
* Поддержка LEFT OUTER JOIN и RIGHT OUTER JOIN с синтаксисами ANSI SQL и ODBC.
* Вы можете смешивать таблицы из разных баз данных в одном запросе.
* Привилегии и система паролей, которая является очень гибкой и безопасной, и позволяет проверку, основанную на имени хоста. Пароли безопасны потому, что вся передача пароля шифрована, когда Вы соединяетесь с сервером.
* Очень быстрые дисковые таблицы B-tree с индексным сжатием.
* Можно иметь до 32 индексов на таблицу. Каждый индекс может состоять от 1 до 16 столбцов или частей столбцов. Максимальная индексная длина 500 байт (это может быть изменено при компиляции MySQL). Индекс может использовать префикс поля CHAR или VARCHAR.
* Записи фиксированной и переменной длины.
* Таблицы в памяти, которые используются как временные таблицы.
* Все столбцы имеют значения по умолчанию. Вы можете использовать вызов INSERT, чтобы вставить подмножество столбцов таблицы. Те столбцы, которым явно не заданы значения, будут автоматически установлены к их значениям по умолчанию.
* Очень быстрая поточно-безопасная система управления памятью.

**Выполнения лабораторной работы**

1. *Задание:*

*SQL-скрипт, использованный для создания БД.*

*Решение:*

CREATE DATABASE \_ikbo\_08\_18\_03

CREATE TABLE Location (

location\_id INT(6) PRIMARY KEY AUTO\_INCREMENT,

regional\_group VARCHAR(20) UNIQUE

);

CREATE TABLE Job (

job\_id INT(6) PRIMARY KEY AUTO\_INCREMENT,

func VARCHAR(30) UNIQUE

);

CREATE TABLE Departament (

departament\_id INT(6) PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(14),

location\_id INT(6),

FOREIGN KEY (location\_id) REFERENCES Location(location\_id)

);

CREATE TABLE Product (

product\_id INT(6) PRIMARY KEY AUTO\_INCREMENT,

description VARCHAR(30)

);

CREATE TABLE Price(

product\_id INT(6) PRIMARY KEY,

FOREIGN KEY (product\_id) REFERENCES Product(product\_id),

start\_date DATE,

end\_date DATE,

list\_price FLOAT(8, 2),

min\_price FLOAT(8, 2)

);

CREATE TABLE Employee (

employee\_id INT(6) PRIMARY KEY AUTO\_INCREMENT,

last\_name VARCHAR(15),

first\_name VARCHAR(15),

middle\_initial VARCHAR(1),

job\_id INT(6),

FOREIGN KEY (job\_id) REFERENCES Job(job\_id),

manager\_id INT(6),

hire\_date DATE,

salary FLOAT(7, 2),

commission FLOAT(7, 2),

departament\_id INT(6),

FOREIGN KEY (departament\_id) REFERENCES Departament(departament\_id)

);

CREATE TABLE Customer (

customer\_id INT(6) PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(45),

address VARCHAR(40),

city VARCHAR(30),

state VARCHAR(3),

zip\_code VARCHAR(9),

area\_code INT(3),

phone\_number INT(7),

salesperson\_id INT(6),

FOREIGN KEY (salesperson\_id) REFERENCES Employee(employee\_id),

credit\_limit FLOAT(9, 2),

comments LONG

);

CREATE TABLE Sales\_Order (

order\_id INT(6) PRIMARY KEY AUTO\_INCREMENT,

order\_date DATE,

customer\_id INT(6),

FOREIGN KEY (customer\_id) REFERENCES Customer(customer\_id),

ship\_date DATE,

total FLOAT(8, 2)

);

CREATE TABLE Item (

item\_id INT(6) PRIMARY KEY AUTO\_INCREMENT,

order\_id INT(6),

product\_id INT(6),

FOREIGN KEY (order\_id) REFERENCES Sales\_Order(order\_id),

FOREIGN KEY (product\_id) REFERENCES Product(product\_id),

actual\_price FLOAT(8, 2),

quantity INT(8),

total FLOAT(8, 2)

);

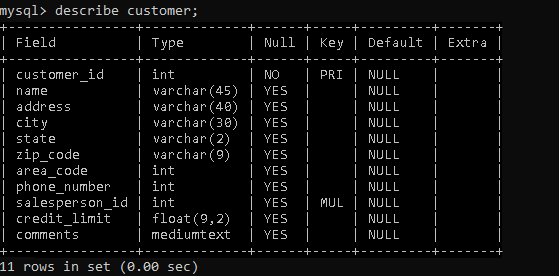
1. *Задание:*

*Структура созданной одной таблицы.*

*Решение:*

*DESCRIBE customer;*

*Результат:*

**

1. *Задание:*

*Пример наполнения одной таблицы.*

*Решение:*

INSERT INTO CUSTOMER (customer\_id, name, address, city,

state, zip\_code, area\_code, phone\_number, salesperson\_id, credit\_limit, comments)

VALUES (1, "Данила", "ул Пушкина", "МСК", "ее", "228", 51, 371557788, 5, 1000, "коммент");

INSERT INTO CUSTOMER (customer\_id, name, address, city,

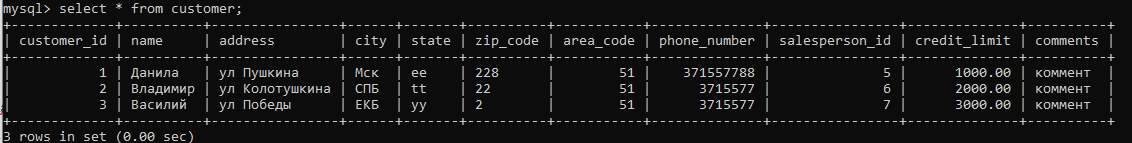
state, zip\_code, area\_code, phone\_number, salesperson\_id, credit\_limit, comments)

VALUES (2, "Владимир", "ул Колотушкина", "СПБ", "tt", "22", 51, 3715577, 6, 2000, "коммент");

INSERT INTO CUSTOMER (customer\_id, name, address, city,

state, zip\_code, area\_code, phone\_number, salesperson\_id, credit\_limit, comments)

VALUES (3, "Василий", "ул Победы", "ЕКБ", "yy", "2", 51, 3715577, 7, 3000, "коммент");*Результат:*



*4.Задание:*

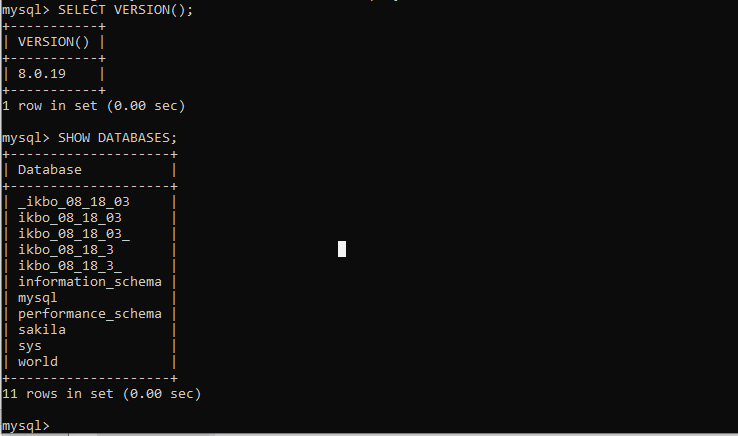
*Версия СУБД и доступные пользователю базы данных.*

*Решение:*

SELECT VERSION ();

SHOW DATABASES;

*Результат:*

**

**Выводы по работе:**

* результате проведения лабораторной работы был получен опыт работы с реляционными базами данных, на примере СУБД MySQL. Были изучены операции, позволяющие отобразить доступные базы данных, структуру таблиц и данные внутри их, а также операции для добавления и изменения самих данных в удобном виде.

**Список использованной литературы.**

1. Сайт о программировании Metanit [Электронный ресурс]: / 2012-2020. URL: <https://metanit.com/sql/mysql/3.1.php>
2. Кузнецов М. В., Симдянов И. В. К89 Самоучитель MySQL 5. — СПб.: БХВ-Петербург, 2006. — 560 с.: ил. ISBN 978-5-94157-754-5
3. Роберт Шелдон, Джоффрей Мойе. MySQL 5: базовый курс = Beginning MySQL. — М.: «Диалектика», 2007. — 880 с. — ISBN 978-5-8459-1167-4.
4. В. Васвани. MySQL: использование и администрирование = MySQL Database Usage & Administration. — М.: «Питер», 2011. — 368 с. — ISBN 978-5-459-00264-5.