

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)

Факультет информатики
Кафедра технической кибернетики

**Отчет по лабораторной работе №2
по курсу «Веб-программирование»**

Выполнила: Поликпашева А.А.

Проверил: Головастиков Н. В.

Группа: 6308

Самара – 2020

Задание

Задание 1

Написать класс, соответствующий пользователю Web-приложения. Этот класс должен:

- содержать приватные поля: id, имя, логин, пароль; а также методы получения и изменения их значений;
- содержать массив списков объектов из лабораторной №1 согласно варианту
- реализовывать методы работы с этим массивом (добавление, получение, удаление, размер).

Задание 2

Создать реляционную модель данных, соответствующую реализованным классам. Как минимум одна из связей между сущностями должна иметь тип «многие ко многим». Написать SQL-скрипты для:

- создания таблиц и заполнения их исходными данными (минимум 5 пользователей, для каждого по 2-3 списка из 5-10 элементов; воспользоваться процедурой чтения из файла, реализованной в лабораторной №1);
- добавления нового пользователя, нового списка или нового элемента списка;
- удаления списка или элемента списка по ключу;
- проверки существования пользователя с заданными логином и паролем;
- выбора всех списков, соответствующих заданному пользователю;
- выбора всех элементов заданного списка;
- + ещё 2 запроса на ваш выбор для демонстрации возможностей базы данных.

Задание 3

Написать класс, осуществляющий подключение к реляционной базе данных. Этот класс должен:

- содержать приватные поля с константными выражениями, необходимыми для подключения к базе данных; • методы для выполнения запросов, написанных в задании 2 (один запрос – один метод);
- аргументами и возвращаемыми значениями методов должны быть объекты написанных ранее пользовательских типов;
- использовать классы библиотеки `java.sql.*`, в том числе запросы с параметрами и транзакции.

Задание 4

Продемонстрировать работу программы на примере написанных запросов. Результаты запросов выводить в консоль.

Вариант 12.

12	Пункты самовывоза: список доступных товаров	Адрес пункта	Наименование Цена (за ед.) Количество Дата поступления	Наименование
----	---	--------------	---	--------------

Основная часть

Задание 1

В процессе выполнения задания создан класс Customer. Он отвечает за пользователя Web-приложения.

Он содержит приватные поля такие как id, имя, логин, пароль, а также методы для получения и изменения их значений. Также класс содержит массив списков объектов DeliveryPoint и реализацию методов для работы с ним, таких как add(), remove(), size(), get().

В работе используется библиотека java.io.IOException, которая отвечает за выброс исключений, когда операция ввода-вывода завершилась неудачно или была прервана.

Также используется библиотека java.util.ArrayList – это автоматически расширяемый массив (один из видов списков).

Задание 2

В процессе выполнения задания была создана реляционная модель данных.

Была создана база данных «delivery_point» на языке запросов MySql.

Также были написаны SQL-запросы для создания таблиц; заполнения таблиц значениями; добавления нового пользователя; нового списка и нового элемента списка; удаления списка и элемента списка по ключу; проверка существования пользователя с заданными логином и паролем; выбор всех списков, соответствующих заданному пользователю; выбора всех элементов заданного списка; вывод всех товаров, заказанных заданным пользователем; вывод клиентов, не имеющих ни одного списка заказов.

Задание 3

В процессе выполнения задания создан класс ClassDB. Он отвечает за подключение к реляционной базе данных.

Он содержит приватные статические поля с константными значениями для подключения к базе данных. А также методы для выполнения запросов, написанных в задании 2: createTable(), insertTable(), addCustomer(), addDeliveryPoint(), addProduct(), deleteDeliveryPoint(), deleteProduct(), controlCustomer(), get_car_by_id_customer(), rent_point_where_max_sum_cost(), model_in_car().

В классе использованы библиотеки класса java.sql.*, которые обеспечивают корректную работу с методами класса. А также библиотеки java.io.IOException и java.util.ArrayList.

Ход работы программы:

1. Запускаем класс ClassDB.
2. С помощью метода createTable() создаются 4 таблицы баз данных – product, order_product, delivery_point, customer. Созданные таблицы можно увидеть в MySQL Workbench (рисунок 1). Также результаты выполнения программы отображаются в консоли программы.

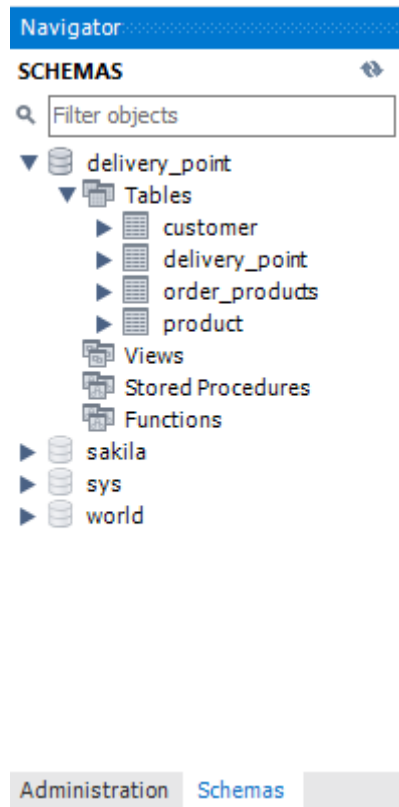


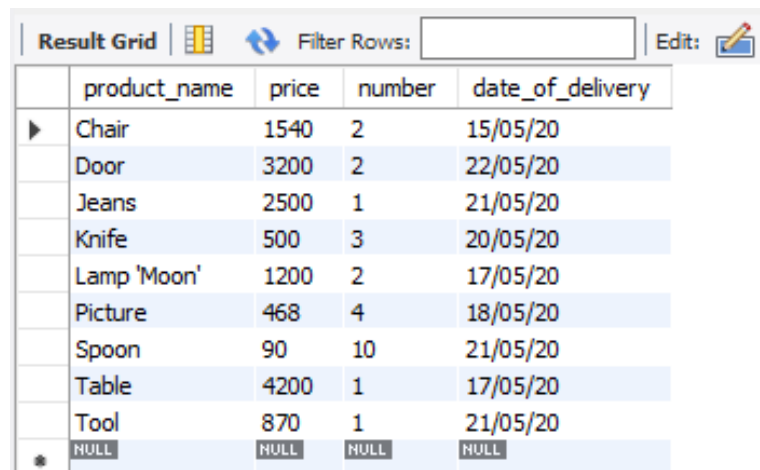
Рисунок 1 – Созданные таблицы в MySQL Workbench

3. Заполняем таблицы с помощью Метод insertTable() заполняет таблицы значениями. Видим результат выполнения в консоли (рисунок 2):

```
"C:\Program Files\Java\jdk-11.0.5\bin\java.exe" "-javaagent:C:\Pro  
  
1 rows added in table PRODUCT  
1 rows added in table ORDER_PRODUCTS  
1 rows added in table DELIVERY_POINT  
1 rows added in table CUSTOMER  
Process finished with exit code 0
```

Рисунок 2 – Заполнение таблиц значениями

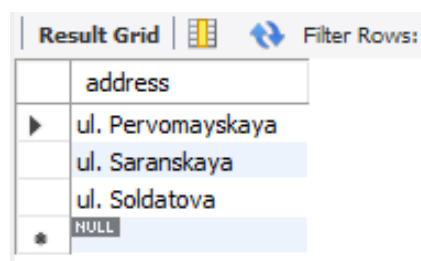
В MySQL Workbench таблицы выглядят так:



The screenshot shows the MySQL Workbench interface with the 'Result Grid' tab selected. The table 'Product' is displayed with the following columns: product_name, price, number, and date_of_delivery. The data is as follows:

	product_name	price	number	date_of_delivery
▶	Chair	1540	2	15/05/20
	Door	3200	2	22/05/20
	Jeans	2500	1	21/05/20
	Knife	500	3	20/05/20
	Lamp 'Moon'	1200	2	17/05/20
	Picture	468	4	18/05/20
	Spoon	90	10	21/05/20
	Table	4200	1	17/05/20
	Tool	870	1	21/05/20
*	NULL	NULL	NULL	NULL

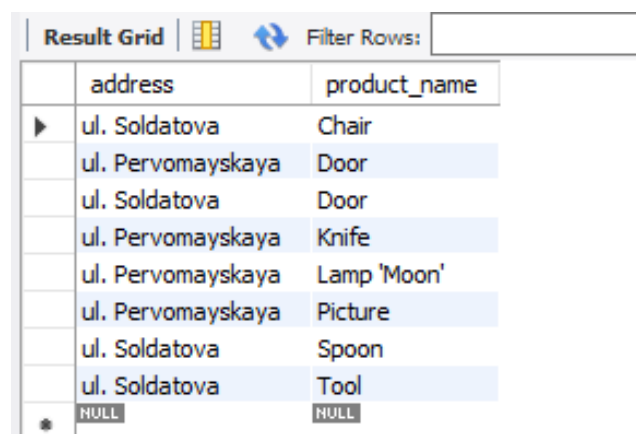
Рисунок 3 – Таблица Product



The screenshot shows the MySQL Workbench interface with the 'Result Grid' tab selected. The table 'Delivery Point' is displayed with the following columns: address. The data is as follows:

	address
▶	ul. Pervomayskaya
	ul. Saranskaya
	ul. Soldatova
*	NULL

Рисунок 4 – Таблица Delivery Point



The screenshot shows the MySQL Workbench interface with the 'Result Grid' tab selected. The table 'Order Products' is displayed with the following columns: address and product_name. The data is as follows:

	address	product_name
▶	ul. Soldatova	Chair
	ul. Pervomayskaya	Door
	ul. Soldatova	Door
	ul. Pervomayskaya	Knife
	ul. Pervomayskaya	Lamp 'Moon'
	ul. Pervomayskaya	Picture
	ul. Soldatova	Spoon
	ul. Soldatova	Tool
*	NULL	NULL

Рисунок 5 – Таблица Order Products



Result Grid

Filter Rows:

	customer_id	name	login	password
▶	1	Ivan	esquire01	780
	2	Olga	smart23	111
	3	Oleg	haha90	635
	4	Rodion	love11	000
	5	Anvar	soldat	777
	6	Ilya	sobol	102
	7	Efrem	toro	365
*	NULL	NULL	NULL	NULL

Рисунок 6 – Таблица Customer

Result Grid

Filter Rows:

	address	customer_id
▶	ul. Pervomayskaya	1
	ul. Saranskaya	1
	ul. Soldatova	1
	ul. Pervomayskaya	2
	ul. Pervomayskaya	3
	ul. Soldatova	4
	ul. Soldatova	5
	ul. Soldatova	6
✱	NULL	NULL

Рисунок 7 – Таблица Point and Customer

Схема БД была создана таким образом: три таблицы - Product, Delivery Point, Customer - отвечают за создание экземпляров классов - уникальных по первичному ключу. Другие две таблицы - Order Products и Point and Customer - отвечают за связи между экземплярами: формируются списки товаров и списки пользователей, относящихся к определенному пункту выдачи товара. Эти две таблицы могут иметь повторяющиеся записи только по одному полю, а по двум полям не могут быть идентичными. Это достигается за счет составного первичного ключа в этих таблицах.

Важно, что создать таблицы Order Products и Point and Customer нельзя, пока не созданы таблицы, отвечающие за экземпляры класса, так как они состоят из первичных ключей этих трех таблиц; по этой же причине нельзя заполнять таблицы Order Products и Point and Customer значениями, которых пока не существует в таблицах Product, Delivery Point, Customer.