Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт космических и информационных технологий |
| институт |
|  |
| Кафедра «Информатика» |
| кафедра |

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №3**

|  |
| --- |
| Работа с базами данных в Spring Framework |
| Тема |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Преподаватель | |  |  |  |  |  | А.С. Черниговский |
|  | |  |  |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ19-16/1б 031939175 | | |  |  |  | А.Д. Непомнящий |
|  | номер группы, зачетной книжки | | |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2021

1. Цель работы

Цель состоит в ознакомлении с механизмами работы с базами данных в Spring Framework.

1. Задачи

Выполнение работы сводится к следующим задачам:

1. описать класс сущности в соответствии с вариантом, который имеет как минимум три текстовых поля, два числовых и ID;
2. создать таблицу базы данных в реляционной БД;
3. реализовать консольное Spring приложение, которое должно позволять:
   1. вводить пользователю поля сущности и; добавлять её в таблицу БД;
   2. выводить в консоль все записи из таблицы БД;
   3. редактировать запись таблицы БД по ID;
   4. удалять запись по ID;
   5. осуществлять поиск по какому-то из признаков

Вариант 10. Одежда.

1. Ход работы
   1. Создание базы данных

В качестве СУБД была выбрана PostgreSQL.

В качестве характеристик предметов одежды были взяты тип, размер, цена, целевая половая принадлежность и количество одежды на складе.

На рисунке 1 приведен скриншот созданной таблицы и SQL-кода для ее создания.

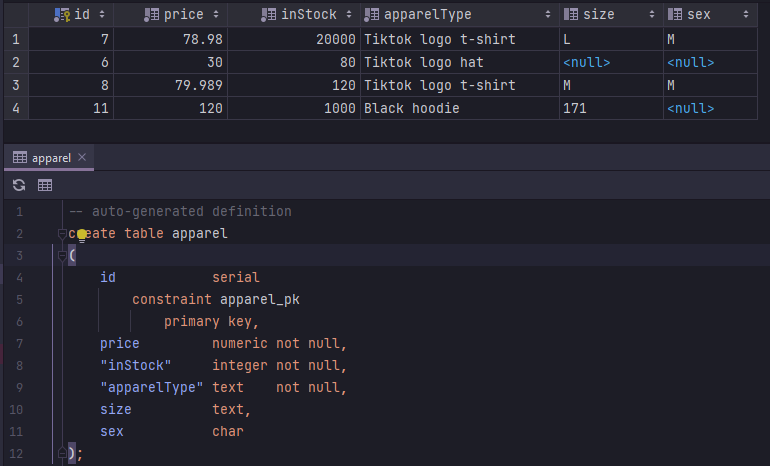


Рисунок 1 – Таблица и ее объявление

* 1. Разработка приложения

Было создано Spring-приложение в соответствии с задачами. На рисунке 2 приведена файловая структура проекта.

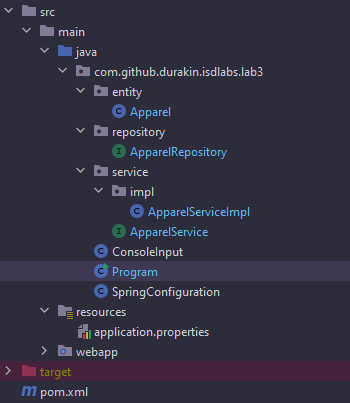


Рисунок 2 – Файловая структура проекта

На листингах далее приведен исходный код программы.

Листинг 1 – код класса Apparel

package com.github.durakin.isdlabs.lab3.entity;

import javax.persistence.\*;

@Entity

public class Apparel {

@Id

@GeneratedValue(strategy =

GenerationType.IDENTITY)

private Integer id;

@Column(nullable = false)

private Double price;

@Column(name = "\"inStock\"", nullable = false)

private Integer inStock;

@Column(name = "\"apparelType\"", nullable = false)

private String apparelType;

@Column

private String size;

@Column

private String sex;

Продолжение листинга 1

public Apparel() {

}

public Apparel(Integer id, Double price, Integer inStock, String apparelType, String size, String sex) {

this.id = id;

this.price = price;

this.inStock = inStock;

this.apparelType = apparelType;

this.size = size;

this.sex = sex;

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public Double getPrice() {

return price;

}

public void setPrice(Double price) {

this.price = price;

}

public Integer getInStock() {

return inStock;

}

public void setInStock(Integer inStock) {

this.inStock = inStock;

}

public String getApparelType() {

return apparelType;

}

public void setApparelType(String apparelType) {

this.apparelType = apparelType;

}

public String getSize() {

return size;

}

public void setSize(String size) {

this.size = size;

}

public String getSex() {

return sex;

}

public void setSex(String sex) {

Окончание листинга 1

this.sex = sex;

}

@Override

public String toString() {

return "id " + id +

"\nType " + apparelType +

"\nSex " + sex +

"\nSize " + size +

"\nPrice " + String.format("%.2f", price) +

"\nLeft in stock " + inStock +

'\n';

}

}

Листинг 2 – Код интерфейса ApparelRepository

package com.github.durakin.isdlabs.lab3.repository;

import com.github.durakin.isdlabs.lab3.entity.Apparel;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import java.util.List;

@Repository

public interface ApparelRepository extends JpaRepository<Apparel, Integer> {

List<Apparel> findDistinctByPriceLessThan(Double price);

}

Листинг 3 – Код интерфейса ApparelService

package com.github.durakin.isdlabs.lab3.service;

import com.github.durakin.isdlabs.lab3.entity.Apparel;

import java.util.List;

public interface ApparelService {

Integer add(Apparel apparel);

boolean delete(Integer id);

Apparel findById(Integer id);

boolean update(Integer id, Integer newInStock, Double newPrice);

List<Apparel> findByMaxPrice(Double maxPrice);

List<Apparel> findAll();

}

Листинг 4 – Код класса ApparelServiceImpl

package com.github.durakin.isdlabs.lab3.service.impl;

import com.github.durakin.isdlabs.lab3.entity.Apparel;

import com.github.durakin.isdlabs.lab3.repository.ApparelRepository;

import com.github.durakin.isdlabs.lab3.service.ApparelService;

import org.springframework.beans.factory.annotation.Autowired;

Окончание листинга 4

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class ApparelServiceImpl implements ApparelService {

@Autowired

private ApparelRepository apparelRepository;

@Override

public Integer add(Apparel apparel) {

this.apparelRepository.save(apparel);

return apparel.getId();

}

@Override

public boolean delete(Integer id) {

if (!checkId(id)) {

return false;

}

this.apparelRepository.deleteById(id);

return true;

}

@Override

public boolean update(Integer id, Integer newInStock, Double newPrice) {

var oldApparel = this.findById(id);

if (oldApparel == null) return false;

if (newPrice != null) {

oldApparel.setPrice(newPrice);

}

if (newInStock != null) {

oldApparel.setInStock(newInStock);

}

this.apparelRepository.save(oldApparel);

return true;

}

@Override

public List<Apparel> findByMaxPrice(Double maxPrice) {

return this.apparelRepository.findDistinctByPriceLessThan(maxPrice);

}

@Override

public List<Apparel> findAll() {

return this.apparelRepository.findAll(Sort.by("id"));

}

@Override

public Apparel findById(Integer id) {

return this.apparelRepository.findById(id).orElse(null);

}

private boolean checkId(Integer id) {

return id != null && this.apparelRepository.existsById(id);

}

}

Листинг 5 – Код конфигурации Spring

package com.github.durakin.isdlabs.lab3;

import lombok.RequiredArgsConstructor;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

import org.springframework.context.annotation.PropertySource;

import org.springframework.core.env.Environment;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

import org.springframework.jdbc.datasource.DriverManagerDataSource;

import org.springframework.orm.jpa.JpaTransactionManager;

import org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean;

import org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter;

import org.springframework.transaction.PlatformTransactionManager;

import javax.persistence.EntityManagerFactory;

import javax.sql.DataSource;

import java.util.Objects;

import java.util.Properties;

@Configuration

@ComponentScan("com.github.durakin.isdlabs.lab3")

@PropertySource("classpath:application.properties")

@EnableJpaRepositories

@RequiredArgsConstructor

public class SpringConfiguration {

private final Environment env;

@Bean

public DataSource dataSource() {

DriverManagerDataSource dataSource = new DriverManagerDataSource();

dataSource.setDriverClassName(Objects.requireNonNull(env.getProperty("spring.datasource.driverClassName")));

dataSource.setUrl(env.getProperty("spring.dataSource.url"));

dataSource.setUsername(env.getProperty("spring.dataSource.username"));

dataSource.setPassword(env.getProperty("spring.dataSource.password"));

return dataSource;

}

@Bean

public Properties jpaProperties() {

return new Properties();

}

@Bean

@Autowired

public EntityManagerFactory entityManagerFactory(DataSource dataSource, Properties jpaProperties) {

HibernateJpaVendorAdapter vendorAdapter = new HibernateJpaVendorAdapter();

LocalContainerEntityManagerFactoryBean factory = new LocalContainerEntityManagerFactoryBean();

factory.setJpaVendorAdapter(vendorAdapter);

factory.setPackagesToScan("com.github.durakin.isdlabs.lab3");

factory.setDataSource(dataSource);

factory.setJpaProperties(jpaProperties);

Окончание листинга 5

factory.afterPropertiesSet();

return factory.getObject();

}

@Bean

@Autowired

public PlatformTransactionManager transactionManager(EntityManagerFactory entityManagerFactory) {

JpaTransactionManager txManager = new JpaTransactionManager();

txManager.setEntityManagerFactory(entityManagerFactory);

return txManager;

}

}

Листинг 6 – Код класса ConsoleInput

package com.github.durakin.isdlabs.lab3;

import java.io.BufferedReader;

import java.util.function.Function;

public class ConsoleInput {

public static Integer CheckedIntInput(String message, BufferedReader reader, Function<Integer, Boolean> checker) {

while (true) {

try {

System.out.print(message);

var number = Integer.parseInt(reader.readLine());

if (!checker.apply(number)) {

throw new Exception("Out of required range");

}

return number;

} catch (Exception e) {

System.err.println(e.getMessage());

}

}

}

public static Integer CheckedIntInput(String message, BufferedReader reader) {

return CheckedIntInput(message, reader, (integer) -> true);

}

public static Double CheckedDoubleInput(String message, BufferedReader reader, Function<Double, Boolean> checker) {

Окончание листинга 6

while (true) {

try {

System.out.print(message);

var number = Double.parseDouble(reader.readLine());

if (!checker.apply(number)) {

throw new Exception("Out of required range");

}

return number;

} catch (Exception e) {

System.err.println(e.getMessage());

}

}

}

public static Double CheckedDoubleInput(String message, BufferedReader reader) {

return CheckedDoubleInput(message, reader, (aDouble -> true));

}

public static String CheckedStringInput(String message, BufferedReader reader, Function<String, Boolean> checker) {

while (true) {

try {

System.out.print(message);

var string = reader.readLine();

if (!checker.apply(string)) {

throw new Exception("String is empty");

}

return string;

} catch (Exception e) {

System.err.println(e.getMessage());

}

}

}

public static String CheckedStringInput(String message, BufferedReader reader) {

return CheckedStringInput(message, reader, (string) -> string != null && !string.isBlank());

}

}

Листинг 7 – Код класса Program

package com.github.durakin.isdlabs.lab3;

import com.github.durakin.isdlabs.lab3.entity.Apparel;

import com.github.durakin.isdlabs.lab3.service.ApparelService;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class Program {

public static void main(String[] args) {

AnnotationConfigApplicationContext context = new AnnotationConfigApplicationContext(SpringConfiguration.class);

var reader = new BufferedReader(new InputStreamReader(System.in));

var service = context.getBean("apparelServiceImpl", ApparelService.class);

boolean lastMenuIteration = false;

while (!lastMenuIteration) {

System.out.println("" +

"1. Print info about all apparels" + '\n' +

"2. Print info about single apparel by ID" + '\n' +

"3. Print apparels with price lower than requested" + '\n' +

"4. Delete element by index" + '\n' +

"5. Edit element" + '\n' +

"6. Add element" + '\n' +

"7. Quit" + '\n'

);

var cmdNumber = ConsoleInput.CheckedIntInput("Enter required command number (1-7) ", reader,

(input) -> 1 <= input && input <= 7);

switch (cmdNumber) {

case 1 -> {

var data = service.findAll();

for (var i : data) {

System.out.println(i);

}

}

Продолжение листинга 7

case 2 -> {

var data = service.findById(ConsoleInput.CheckedIntInput("Enter required apparel ID ", reader));

System.out.println(data != null ? data : "Element not found ");

}

case 3 -> {

var data = service.findByMaxPrice(ConsoleInput.CheckedDoubleInput("Enter required price ", reader));

System.out.println("Found " + data.size() + " elements");

for (var i : data) {

System.out.println(i);

}

}

case 4 -> {

var result = service.delete(ConsoleInput.CheckedIntInput("Enter id of the element to delete ", reader));

System.out.println(result ? "Successfully deleted " : "Element not found ");

}

case 5 -> {

var id = ConsoleInput.CheckedIntInput("Enter id of the element to edit ", reader);

var element = service.findById(id);

if (element == null) {

System.out.println("Element not found");

break;

}

Double newPrice = null;

Integer newInStock = null;

var lastSubMenuIteration = false;

while (!lastSubMenuIteration) {

System.out.println("Element\n" + element);

System.out.println((newPrice == null ? "" : "New price will be " + newPrice + '\n') + (newInStock == null ? "" : "New number in stock will be " + newInStock + '\n'));

System.out.println("1. Edit price\n" +

"2. Edit number in stock\n" +

"3. Finish editing\n" +

"4. Cancel editing\n"

);

var subCmdNumber = ConsoleInput.CheckedIntInput("Enter required command number (1-4) ", reader,

(input) -> 1 <= input && input <= 4);

Продолжение листинга 7

switch (subCmdNumber) {

case 1 -> newPrice = ConsoleInput.CheckedDoubleInput("Enter new apparel's price", reader);

case 2 -> newInStock = ConsoleInput.CheckedIntInput("Enter new number of apparels in stock", reader);

case 3 -> lastSubMenuIteration = true;

case 4 -> {

newInStock = null;

newPrice = null;

}

}

}

System.out.println(service.update(id, newInStock, newPrice) ? "Updated" : "Something went wrong");

}

case 6 -> {

var apparel = new Apparel();

apparel.setApparelType(ConsoleInput.CheckedStringInput("Enter apparel's type ", reader));

var sex = ConsoleInput.CheckedStringInput("Enter apparel's sex (M, F or Uni) ", reader, Program::ApparelsSexChecker);

apparel.setSex(sex.length() == 1 ? sex : null);

var size = ConsoleInput.CheckedStringInput("Enter apparel's size (- for non-sized items) ", reader);

apparel.setSize(!size.equals("-") ? size : null);

var inStock = ConsoleInput.CheckedIntInput("Enter number of items in stock ", reader, (integer -> integer > 0));

apparel.setInStock(inStock);

var price = ConsoleInput.CheckedDoubleInput("Enter price of the item ", reader, (aDouble -> aDouble >= 0));

apparel.setPrice(price);

var id = service.add(apparel);

System.out.println("Apparel added " + service.findById(id));

}

case 7 -> lastMenuIteration = true;

}

}

System.out.println();

context.close();

System.exit(0);

Окончание листинга 7

}

private static boolean ApparelsSexChecker(String string) {

return string.equals("M") || string.equals("F") || string.equals("Uni");

}

}

* 1. Демонстрация работы приложения

Далее на рисунках приведены примеры работы программы.



Рисунок 3 – Вывод всех записей в таблице

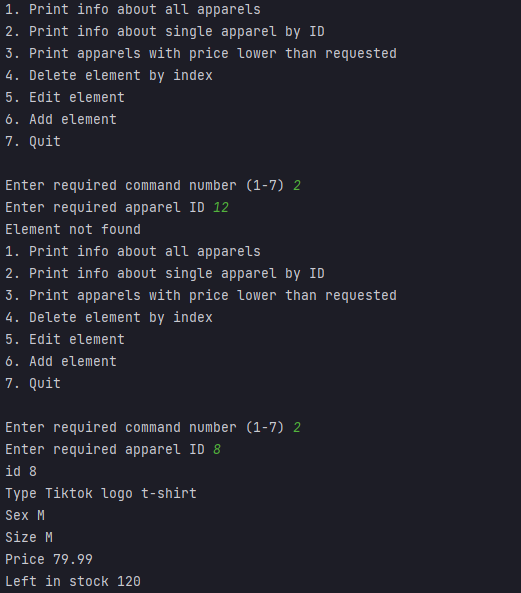


Рисунок 4 – Вывод записи по ID

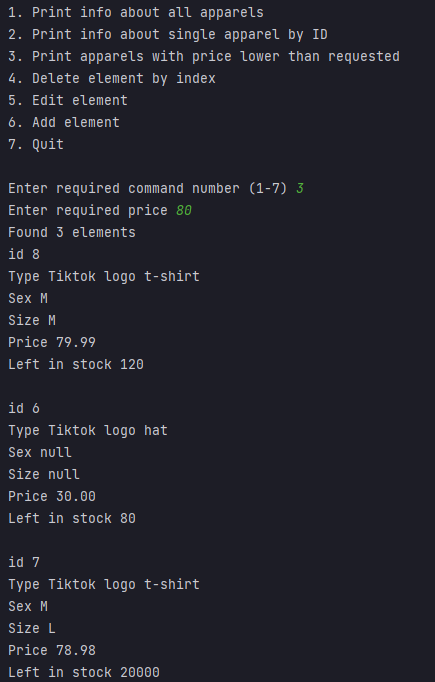


Рисунок 5 – Вывод записей по условию (дешевле заданной цены)

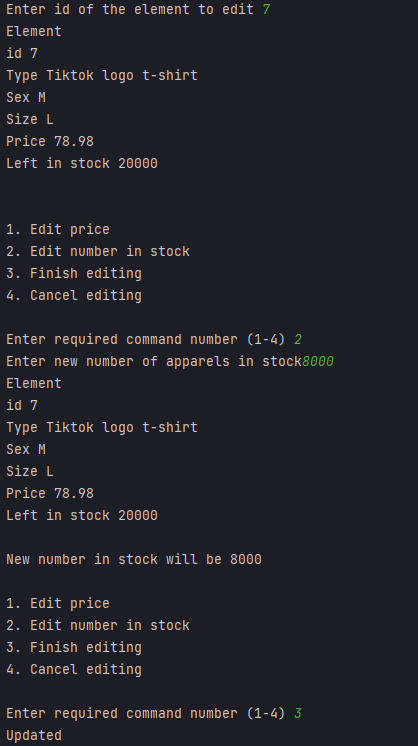


Рисунок 6 – Редактирование записи по ID

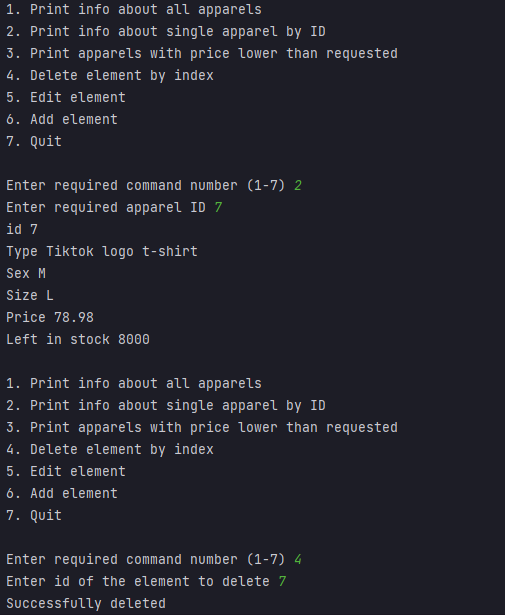


Рисунок 7 – Удаление записи по ID

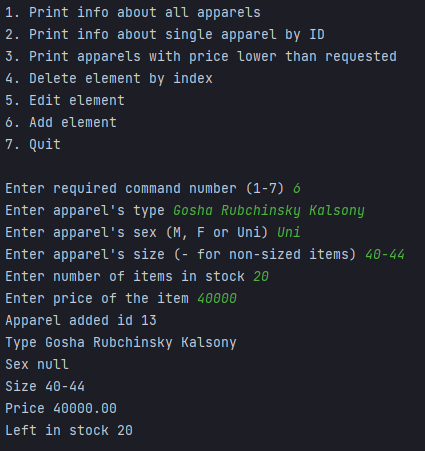


Рисунок 8 – Добавление записи в таблицу

1. Вывод

Было произведено ознакомление с механизмами работы с базами данных в Spring Framework. Была создана БД в PostgreSQL и реализовано приложение в соответствии с заданием.