



Министерство науки и высшего образования Российской  
Федерации Федеральное государственное автономное  
образовательное учреждение высшего образования  
**«Пермский национальный исследовательский  
политехнический университет»**

Электротехнический факультет  
Кафедра «Информационные технологии и автоматизированные системы»  
направление подготовки: 09.03.04 Программная инженерия

**О Т Ч Е Т**  
по учебной практике

Выполнил студент гр. РИС-19-16

Миннахметов Э.Ю.

(фамилия, имя, отчество)

(подпись)

Проверил:

старший преподаватель кафедры ИТАС Кузнецов Д.Б.

(должность, ФИО руководителя по практической подготовке от кафедры)

(оценка)

(подпись)

(дата)

Пермь 2021

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	3
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ .....	4
1.1 Ознакомление с инструментарием .....	4
1.2 Ознакомление с псевдо-объектным представлением данных в системе МЕТА .....	4
1.3 Анализ поставленной задачи.....	5
2 ТЕХНОЛОГИЯ РЕАЛИЗАЦИИ .....	7
2.1 Настройка индексации сервлета .....	7
2.2 Оформление кода и дизайн веб-интерфейса .....	7
2.3 Выполнение запросов к системе МЕТА .....	12
ЗАКЛЮЧЕНИЕ .....	23
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ .....	24
ПРИЛОЖЕНИЕ А .....	25
ПРИЛОЖЕНИЕ Б .....	26

## **ВВЕДЕНИЕ**

ООО «Комплексные системы» относится ко множеству дочерних компаний ООО «Сириус». Во всех этих компаниях используется собственный продукт – система META. Она позволяет работать с такими базами данных, как PostgreSQL и NoSQL-СУБД Redis. В системе META используется собственная нотация представления данных, называемая Псевдо-объектным представлением данных.

**Цель** практической работы: разработать программный продукт для работы с системой META.

Для исполнения поставленной цели, ее необходимо разбить на следующие **задачи**:

- 1) проанализировать поставленную задачу;
- 2) разработать программную часть.

# 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Суть данного раздела заключается в рассмотрении инструментария, который будет необходим при работе. Затем будут изложены принципы псевдо-объектного представления данных и их реализация в системе META. После чего будут описаны задания, поставленные компанией. Стоит начать с рассмотрения инструментария.

## 1.1 Ознакомление с инструментарием

В качестве операционной системы будет использоваться *Linux* – это корпоративное правило ООО «Комплексные системы» - выбор дистрибутива остается за программистом. В моем случае, будет использоваться *KDE Neon* из вкусовых соображений.

Программный код будет написан на языке *Java* и всем практикантам рекомендуется использовать среду разработки *IntelliJ IDEA*, поскольку в ней был создан проект и иной выбор может повлечь проблемы совместимости. Выбор платной *Ultimate* или бесплатной *Community* версии также остается за программистом. В моем случае, это *Ultimate* по студенческой подписке.

Подключение к серверу будет производиться по *SSH*, но прежде, чем подключиться к серверу, необходимо будет подключиться к *VPN-сети* – данная задача будет выполняться с помощью консольной программы *OpenVPN*.

Таким образом, был представлен набор программного обеспечения, необходимый для комфортной работы. Далее будут рассмотрены принципы *псевдо-объектного представления данных* и их реализация в системе *META*.

## 1.2 Псевдо-объектное представление данных в системе META

Понятие "мета" используется для описания нескольких вещей:

- а) наша основная библиотека доступа к данным - *lib\_meta.jar*;
- б) родной веб-интерфейс доступа к данным (без реализации бизнес-логики);
- в) подход к представлению, хранению, доступу к данным.

Данные лежат в множестве реляционных баз, библиотека `lib_meta.jar` (классы пакета `arpt.meta3.*`) обеспечивает доступ к ним в виде объектов с атрибутами. Каждый объект имеет уникальный идентификатор, сквозной для всех баз, номер типа, дату/время создания, смерти и автора объекта.

— тип объекта — это число от 1 до 999, определяет набор атрибутов для всех объектов типа, параметры прав доступа к объектам, а также расположение объектов типа в множестве реляционных баз;

— уникальный идентификатор объекта (ID объекта) может быть 12- или 18-значным, ID объекта однозначно определяет тип объекта (первые четыре знака минус 1000);

— атрибуты объектов бывают 4 типов: строковый, численный, дата со временем, ссылка на другой объект. Для обращения к атрибутам объекта используется ID атрибута типа, он уникален в контексте типа, ID атрибута типа имеет длину 10 знаков, ID однозначно определяет номер типа, атрибутом объектов которого он является (первые четыре знака минус 1000), и тип атрибута (пятый знак). С атрибутами каждого типа лучше знакомиться на конкретных примерах объектов.

На этом, объяснение того, что такое система МЕТА, будет закончено, поскольку детали реализации являются коммерческой тайной компании, предоставляющей место для практики. Далее будут рассмотрены задачи, которые поставила компания перед командой практикантов.

### **1.3 Анализ поставленной задачи**

Для выполнения задачи потребуется скопировать проект с системы управления версиями Subversion (SVN) с сервера компании, к сети которого будет произведено подключение по VPN. Детали данных действий несут тайный характер, поэтому следует перейти к непосредственному рассмотрению некоторых из задач:

1) Выписать на странице в таблице все ID + Города (тип 5) определенной области. Сделать возможность менять территорию в форме.

2) Сделать форму для создания описания экскурсии (тип 506):

- название 1506410000;
- описание краткое 1506410282;
- регионы 1506923461;
- дополнительно оплачиваются входные билеты 1506223120;
- бронировать у партнера 1506910189 - ссылка на партнера 158 типа, название 1001211;
- Тип (экскурсия - 0, билет - 1, спорт - 2, прокат - 3, услуга - 4, СПА - 5, авиация - 6) 1506310181.

3) Сделать возможность удалять объекты 506 типа в форме - заносим ID объекта, который надо удалить, по кнопке «Удалить» - объект удаляется.

4) Написать интерфейс, чтобы отображать данные из Redis. Зашли в интерфейс, написать откуда взяли данные, если в редисе нет ключа, написать, что заново положили ключ+значение (тип 46) в Redis и положить этот ключ+значение, или просто достали из редиса - написали об этом. В интерфейсе по ID 46 надо отображать следующую информацию: 1000348 Название, 1000350 ID номера, 1046222729 тип стоимости (0/1/2 - С/БНС/НС). Все кладем в редис rev, время жизни ключа = 3 минуты. (ID для теста 104610001184, 104610000865, 104610000863, 104610000831, 104610000807, 104610000783, 104610000561, 104610000529, 104610000495).

5) Добавить любой ключ в Redis и научиться его удалять.

Такова трактовка заданий, предоставленная ООО «Комплексные системы».

**Вывод,** анализ поставленной задачи выполнен. Далее будет изложена разработка приложения для работы с системой META в базах данных PostgreSQL и Redis.

## 2 ТЕХНОЛОГИЯ РЕАЛИЗАЦИИ

В данном разделе будет показан код сервлета на языке Java, который, в зависимости от GET-параметров, будет выполнять запросы к системе META, затем заворачивать их в требуемый формат – это HTML. Прежде всего необходимо прописать индексацию сервлета.

### 2.1 Настройка индексации сервлета

Подобные настройки должны быть расположены в файле web.xml, ниже представлен значащий отрывок из этого файла – его полная часть изложена в Приложении А.

Листинг 2.1 – отрывок из Приложения А, файл web.xml

```
<servlet>
    <servlet-name>EldarServlet</servlet-name>
    <servlet-class>intern.EldarServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>EldarServlet</servlet-name>
    <url-pattern>/eldar</url-pattern>
</servlet-mapping>
```

Здесь указано имя сервлета, класс сервлета и адрес страницы, по которому можно обратиться в браузере и который должен быть дополнен адресом хоста.

Таким образом, был проиндексирован мой сервлет и при обращении к серверу с запросом веб-страницы “/eldar”, будет выводиться результат работы написанного мной сервлета EldarServlet, оформление которого, а также дизайн веб-интерфейса, будут приведены в следующем подразделе.

### 2.2 Оформление кода и дизайн веб-интерфейса

Для ориентации в коде необходима нумерация строк, поэтому код будет прилагаться скриншотами. Изложение пакета и подключения сторонних пакетов библиотек имеет смысл опустить. Ниже приведено определение класса-сервлета

(рис. 2.1) и показаны поля экземпляра, статические поля и статическое определение. Полный код класса находится в Приложении Б.

```
18 public class EldarServlet extends HttpServlet {
19     private PrintWriter out;
20     private ResourceBundle mains;
21     private HttpServletRequest request;
22     private HttpServletResponse response;
23     private final int myId = 1000360;
24     private final String redis = "rev";
25     private final String prefix = "Eldar";
26
27     private static final Map<String, String> typeMap = new HashMap<>();
28     private static final Map<String, String> yesnoMap = new HashMap<>();
29     private static final Map<String, List<String>> pagesMap = new TreeMap<>();
30
31     static {
32         typeMap.put("", "-");
33         typeMap.put("0", "Экскурсия");
34         typeMap.put("1", "Билет");
35         typeMap.put("2", "Спорт");
36         typeMap.put("3", "Прокат");
37         typeMap.put("4", "Услуга");
38         typeMap.put("5", "СПА");
39         typeMap.put("6", "Авиация");
40         typeMap.put("8", "Концерт");
41
42         yesnoMap.put("", "-");
43         yesnoMap.put("1", "Да");
44         yesnoMap.put("0", "Нет");
45
46         pagesMap.put("PostgreSQL", Arrays.asList("Один", "Два",
47             "Три", "Четыре", "Пять", "Шесть", "Семь", "Восемь"));
48         pagesMap.put("Redis", Arrays.asList("Один", "Два", "Три", "Четыре"));
49     }
```

Рисунок 2.1 – Объявление класса-сервлета

В коде объявлены объект потока вывода, обращения к ресурсам, заголовков запроса, заголовков ответа, мой личный идентификатор для работы с системой МЕТА, имя сервера Redis, мой префикс для ключей в Redis, затем словари для типов мероприятий, для утвердительных ответов и для навигации по веб-



интерфейсу.

Далее будет представлен метод обработки GET-запросов (рис. 2.2).

```
64 public void doGet(HttpServletRequest req, HttpServletResponse res) throws IOException {
65     initialize(req, res);
66     try {
67         int task = Integer.parseInt(request.getParameter("task"));
68         if(task < 1 || task > 12) {
69             throw new NumberFormatException();
70         }
71         if(task == 1) {
72             task1();
73         } else if(task == 2) {
74             task2();
75         } else if(task == 3) {
76             task3();
77         } else if(task == 4) {
78             task4();
79         } else if(task == 5) {
80             task5();
81         } else if(task == 6) {
82             task6();
83         } else if(task == 7) {
84             task7();
85         } else if(task == 8) {
86             task8();
87         } else if(task == 9) {
88             task9();
89         } else if(task == 10) {
90             task10();
91         } else if(task == 11) {
92             task11();
93         } else if(task == 12) {
94             task12();
95         }
96     } catch (NumberFormatException e) {
97         printHtml( title: "Hello", text: "<h1>Привет Sirius!</h1>");
98     } catch (Exception e) {
99         printException(e);
100     }
101     out.flush();
102 }
```

Рисунок 2.2 – Метод обработки GET-запросов

Здесь выполняется метод инициализации и выполняется метод задания, соответствующий номеру задания из запроса. Аналогично, ниже будет приведен

метод обработки POST-запросов (рис. 2.3).

```
51 public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException {  
52     initialize(req, res);  
53     String task = request.getParameter("task");  
54     try {  
55         if ("7".equals(task)) {  
56             task7post();  
57         }  
58     } catch (Exception e) {  
59         printException(e);  
60     }  
61     out.flush();  
62 }
```

Рисунок 2.3 – Метод обработки POST-запросов

Здесь также выполняется метод инициализации и выполняется метод задания, однако использование метода запросов POST используется только в 7 задании – здесь это 2 задание, оно будет показано позже. Далее представлен код инициализации (рис. 2.4).

```
104 @ public void initialize(HttpServletRequest req, HttpServletResponse res) throws IOException {  
105     request = req;  
106     response = res;  
107     response.setCharacterEncoding("UTF-8");  
108     request.setCharacterEncoding("UTF-8");  
109     out = res.getWriter();  
110     mains = ResourceBundle.getBundle("intern_main", new Locale("ru", "RU"));  
111 }
```

Рисунок 2.4 – Инициализация

Суть данного метода заключается в присвоении ссылок на объекты заголовков запроса и ответа в поля экземпляра сервлета, а также уточнение кодировки, получение ссылки на объект потока вывода и объект обращения к ресурсам. В ходе выполнения программы могут возникать исключения, которые обрабатываются в методах doGet и doPost, вывод текста исключений показан ниже (рис. 2.5).

```

113 @ public void printException(Exception e) {
114     StackTraceElement[] stack = e.getStackTrace();
115     String[] stackTrace = new String[stack.length];
116     for(int i = 0; i < stack.length; ++i) {
117         stackTrace[i] = stack[i].toString();
118     }
119     printHtml( title: "Exception!", String.format("<h1>%s: %s</h1><div>%s</div>",
120         e.getClass().getName(), e.getMessage(), String.join( delimiter: "<br>", stackTrace)));
121 }

```

Рисунок 2.5 – Вывод текста исключений

Каждое задание и вывод исключений формируют определенный текст, помимо это каждая страница должна иметь название и стили – всем этим пользуется метод вывод HTML кода, который приведен ниже (рис. 2.6).

```

691 public void printHtml(String title, String text) {
692     response.setContentType("text/html; charset=UTF-8");
693     String task = request.getParameter( S: "task");
694     StringBuilder output = new StringBuilder();
695     output.append(String.format("<!doctype html><html><head><title>%s</title><style>" +
696         "table { margin: 0px auto; }" +
697         "table.form tr td:first-child { text-align:right; }" +
698         "table.form tr td:last-child { text-align:left; }" +
699         "table.data { border: solid 1px #ccc; border-spacing: 3px; " +
700         "border-collapse: collapse; margin-bottom:10px; }" +
701         "table.data tr:first-child { font-weight:bold; }" +
702         "table.data td { border: solid 1px #ccc; padding: 5px; }" +
703         "form { margin: 0px auto; margin-bottom:20px; text-align:center; " +
704         "padding:10px; }" +
705         "</style></head>" +
706         "<body><div style=\"text-align:center;margin-bottom:20px;\>" +
707         "<table class=\"form\">", title));
708     int tasks = 0;
709     for(String key : pagesMap.keySet()) {
710         output.append(String.format("<tr>" +
711             "<td style=\"font-weight:bold;color:#072d78;padding-right:15px;\>" +
712             "%s</td><td>", key));
713         List<String> numbers = pagesMap.get(key);
714         for(int i = 1, n = numbers.size(); i <= n; ++i) {
715             if(String.valueOf(tasks + i).equals(task)) {
716                 output.append(String.format("<b style=\"color:grey\">%s</b>%s",
717                     numbers.get(i - 1), i < n ? " | " : ""));
718             } else {
719                 output.append(String.format("<a href=\"?task=%d\">%s</a>%s",
720                     tasks + i, numbers.get(i - 1), i < n ? " | " : ""));
721             }
722         }
723         output.append("</td></tr>");
724         tasks += numbers.size();
725     }
726     output.append(String.format("</table></div>%s</body></html>", text));
727     out.print(output);
728 }

```

Рисунок 2.6 – Вывод HTML-кода

По итогу получаем приложенный ниже веб-интерфейс. Имеются две секции заданий: система META в PostgreSQL и в Redis, - а далее ссылки на сами задания, ниже вывод самого задания (рис. 2.7). В данном случае, шестое задание, которое не будет рассматриваться в отчете, но очень демонстративно в плане дизайна – есть и форма, и таблица – элементы, на которые был упор в стилях CSS.

PostgreSQL

[Один](#) | 
 [Два](#) | 
 [Три](#) | 
 [Четыре](#) | 
 [Пять](#) | 
 [Шесть](#) | 
 [Семь](#) | 
 [Восемь](#)

Redis

[Один](#) | 
 [Два](#) | 
 [Три](#) | 
 [Четыре](#)

Тип

Концерт ▾

Код

Адрес

Категория

- ▾

Контрагент

- ▾

Выбрать

№	Название	Тип	Адрес	Да/Нет	Контрагент
1	(Переименовал) Тест 506/36 - 10	Концерт	Олимпийский проспект, д.1	-	КонтрАг
2	(Переименовал) Тест 506/36 - 20	Концерт	Олимпийский проспект, д.1	-	КонтрАг
3	(Переименовал) Тест 506/36 - 240	Концерт	Олимпийский проспект, д.1	-	КонтрАг
4	(Переименовал) Тест 506/36 - 30	Концерт	Олимпийский проспект, д.1	-	КонтрАг
5	(Переименовал) Тест 506/36 - 360	Концерт	Олимпийский проспект, д.1	-	КонтрАг
6	(Переименовал) Тест 506/36 - 40	Концерт	Олимпийский проспект, д.1	-	КонтрАг
7	(Переименовал) Тест 506/36 - Базис	Концерт	Олимпийский проспект, д.1	-	КонтрАг

Рисунок 2.7 – Вывод шестого задания в браузере

Таким образом, было показано оформление Java-кода, показаны стили CSS и пример вывода HTML-кода с его визуализацией в браузере. В следующем подразделе будут рассматриваться задания.

## 2.3 Выполнение запросов к системе META

Первым на очереди будет первое задание, в котором демонстрируется выборка данных из базы – в данном случае, выборка областей, а затем соответствующих выбранной области городов (рис. 2.8).

```

123 public void task1() throws Exception {
124     Map<String, String> regions = getRegions( countryId: "100410000050");
125     Obb filter = Obb.createFilter( tid: 5);
126     String regionId = request.getParameter( s: "region");
127     if(regionId == null || regionId.equals("")) {
128         regionId = "100518301512";
129     }
130     Obb.addCondition(filter, atid: 1005101368, Obb.ComparisonType.EQ, regionId);
131     Obb[] cities = Obb.getSrcObs(mains, filter, m: 0, entr: 0);
132     List<String> regionIds = intern.Utills.getKeysSortedByValue(regions, bDescending: false);
133     StringBuilder text = new StringBuilder("<form action=\"#\\" method=\"get\">" +
134         "<table class=\"form\"><tr><td>Перион</td><td><select name=\"region\">";
135     for(String key : regionIds) {
136         text.append(String.format("<option% s value=\"%s\">%s</option>",
137             key.equals(regionId) ? " selected" : "", key, regions.get(key)));
138     }
139     text.append("</select></td></tr></table><input type=\"submit\" value=\"Выбрать\">" +
140         "<input type=\"hidden\" name=\"task\" value=\"1\"></form>");
141     Arrays.sort(cities, Comparator.comparing((Obb ob) -> ob.getAt( id_at: "1000098")));
142     int i = 0;
143     text.append("<table class=\"data\"><tr><td>№</td><td>ID</td><td>Город</td></tr>");
144     for(Obb city : cities) {
145         text.append(String.format("<tr><td>%d</td><td>%s</td><td>%s</td></tr>",
146             ++i, city.id, Obb.getAt(city, id_at: "1000098")));
147     }
148     text.append("</table>");
149     printHtml( title: "Города России", text.toString());
150 }

```

Рисунок 2.8 – Выборка городов по области

Здесь извлекаются области из базы, создается фильтр с типом городов, затем он дополняется условием с идентификатором выбранной области. Ниже приведен метод выборки областей (рис. 2.9).

```

152 public Map<String, String> getRegions(String countryId) throws Exception {
153     Obb filter = Obb.createFilter( tid: 5);
154     Obb.addCondition(filter, atid: 1000004, Obb.ComparisonType.EQ, countryId);
155     Obb.addCondition(filter, atid: 1005101368, Obb.ComparisonType.NEQ, val: "");
156     Obb[] cities = Obb.getSrcObs(mains, filter, m: 0, entr: 0);
157     Map<String, String> regions = new TreeMap<>();
158     for(Obb city : cities) {
159         String regionId = Obb.getAt(city, id_at: 1005101368);
160         if(!regions.containsKey(regionId)) {
161             regions.put(regionId, Obb.getZn(mains, regionId, id_oba: 1000098, type_oba: 4));
162         }
163     }
164     return regions;
165 }

```

Рисунок 2.9 – Выборка областей

Здесь извлекаются все регионы России и ими заполняется словарь для быстрого доступа к названию по идентификатору. Ниже представлен вывод в браузере (рис. 2.10).

Регион Московская область ▼

Выбрать

№	ID	Город
1	100510397181	Балашиха
2	100510397187	Бронницы
3	100510520109	Видное
4	100510397196	Волоколамск
5	100510397199	Воскресенск

Рисунок 2.10 – Вывод первого задания в браузере

Далее будет представлено задание с созданием нового объекта системы МЕТА и его последующая загрузка в базу (рис. 2.11).

```

400 public void task7() throws Exception {
401     Map<String, String> regions = new HashMap<>();
402     Map<String, String> partners = new HashMap<>();
403     regionsAndPartners(regions, partners);
404
405     StringBuilder text = new StringBuilder();
406     text.append("<form method=\"post\" action=\"#\"><table class=\"form\"> +
407         "<tr><td>Название</td><td><input type=\"text\" name=\"name\"></td></tr> +
408         "<tr><td>Описание</td><td><textarea name=\"desc\"></td></tr> +
409         "<tr><td>Регион</td><td><select name=\"region\">");
410     List<String> regionIds = intern.Utills.getKeysSortedByValue(regions, bDescending: false);
411     for(String regionId : regionIds) {
412         text.append(String.format("<option value=\"%s\">%s</option>",
413             regionId, regions.get(regionId)));
414     }
415     text.append("</select></td></tr> +
416         "<tr><td>Доп. оплата</td><td><input type=\"text\" name=\"cost\"></td></tr> +
417         "<tr><td>Бронирование у партнера</td><td><select name=\"partner\">");
418     for(String partnerId : partners.keySet()) {
419         text.append(String.format("<option value=\"%s\">%s</option>",
420             partnerId, partners.get(partnerId)));
421     }
422     text.append("</select></td></tr> +
423         "<tr><td>Тип</td><td><select name=\"type\">");
424     for(String typeId : typeMap.keySet()) {
425         text.append(String.format("<option value=\"%s\">%s</option>",
426             typeId, typeMap.get(typeId)));
427     }
428     text.append("</select></td></tr> +
429         "</table><input type=\"hidden\" name=\"task\" value=\"7\"> +
430         "<input type=\"submit\" value=\"Создать\"></form>");
431     printDescs(text, regions, partners);
432 }

```

Рисунок 2.11 – Гипертекст формы создания объекта

Суть показанного метода в выводе формы в браузер, а также выводе всех объектов данного типа, реализованном в другом методе. Результат работы метода можно наблюдать ниже (рис. 2.12).

Название: 1 сентябре в Политехе

Описание: Тестовое описание

Регион: Пермь

Доп. оплата: 100

Бронирование у партнера: тестовый Сириус

Тип: Концерт

Создать

Рисунок 2.12 – Форма создания объекта

Здесь показана форма, дополнительно заполненная для создания и записи объекта в систему МЕТА. Результатом нажатия на кнопку «Создать» будет дальнейший вывод в таблице нового объекта (рис. 2.13).

№	ID	Название	Описание	Регион	Доп. оплаты	Бронирование у партнера	Тип
1	150610003700	1 сентябре в Политехе	Тестовое описание	Пермь	100	тестовый Сириус	Концерт
2	150610003660	Агрызок	Агрыз - город мечта	null	1	тестовый Сириус	Билет
3	150610003560	title	description	null	0	тестовый Сириус	Экскурсия

Рисунок 2.13 – Новый объект

Теперь же следует перейти к рассмотрению того, как происходит обработка POST-запроса, создание объекта и его загрузка в систему МЕТА. Данный код представлен ниже (рис. 2.14).



```

434 public void task7post() throws Exception {
435     String name = request.getParameter( S: "name");
436     String desc = request.getParameter( S: "desc");
437     String region = request.getParameter( S: "region");
438     String cost = request.getParameter( S: "cost");
439     String partner = request.getParameter( S: "partner");
440     String type = request.getParameter( S: "type");
441
442     name = name == null ? "" : name;
443     desc = desc == null ? "" : desc;
444     region = region == null ? "-" : region;
445     cost = !NumberUtils.isNumber(cost) ? "" : cost;
446     partner = partner == null ? "-" : partner;
447     type = !typeMap.containsKey(type) ? "-" : type;
448
449     if(!name.equals("") && !desc.equals("") && !region.equals("-") &&
450         !cost.equals("") && !partner.equals("-") && !type.equals("-")) {
451         Obb ob = new Obb( id_type: 506);
452         ob.id_user = myId;
453         Obb.addAt(ob, id_at: "1506410000", name);
454         Obb.addAt(ob, id_at: "1506410282", desc);
455         Obb.addAt(ob, id_at: "1506923461", region);
456         Obb.addAt(ob, id_at: "1506223120", cost);
457         Obb.addAt(ob, id_at: "1506910189", partner);
458         Obb.addAt(ob, id_at: "1506310181", type);
459         Obb.addOb(mains, ob);
460     }
461     response.sendRedirect( S: "/eldar?task=7");
462 }

```

Рисунок 2.14 – Создание объекта и его загрузка в базу

Здесь создается объект, получает свои атрибуты из POST-запроса, отправляется в базу данных и выполняется перенаправление с GET-запросом на страницу с заданием. Далее будут показано извлечение из базы регионов и партнеров, необходимых для выполнения задания (рис. 2.15).



```

481 @
482
483 public void regionsAndPartners(Map<String, String> regions,
484                               Map<String, String> partners) throws Exception {
485     Obb filter = Obb.createFilter( tid: 5);
486     Obb.addCondition(filter, atid: 1000004, Obb.ComparisonType.EQ, val: "100410000050");
487     Obb.addCondition(filter, atid: 1005101368, Obb.ComparisonType.NEQ, val: "");
488     Obb[] cities = Obb.getSrcObs(mains, filter, m: 0, entr: 0);
489     Obb[] prtns = Obb.getSrcObs(mains, Obb.createFilter( tid: 158), m: 0, entr: 0);
490     regions.put("", "-");
491     partners.put("", "-");
492     for(Obb city : cities) {
493         regions.put(city.id, city.getAt( id_at: "1000098"));
494     }
495     for(Obb partner : prtns) {
496         partners.put(partner.id, partner.getAt( id_at: "1001211"));
497     }
498 }

```

Рисунок 2.15 – Извлечение регионов и партнеров

Как видно, все очень просто и не требует детального рассмотрения. Теперь же следует приступить к выполнению следующего задания, код которого представлен ниже (рис. 2.16).

```

464
465 public void task8() throws Exception {
466     String descId = request.getParameter( s: "did");
467     descId = descId == null ? "" : descId;
468     if(!descId.equals("")) {
469         Obb.delOb(mains, descId, myId);
470     }
471     Map<String, String> regions = new HashMap<>();
472     Map<String, String> partners = new HashMap<>();
473     regionsAndPartners(regions, partners);
474     StringBuilder text = new StringBuilder();
475     text.append("<form method=\"get\" action=\"#\>" +
476             "ID <input type=\"text\" name=\"did\">" +
477             "<input type=\"submit\" value=\"Удалить\">" +
478             "<input type=\"hidden\" name=\"task\" value=\"8\"></form>");
479     printDescs(text, regions, partners);
480 }

```

Рисунок 2.16 – Удаление объекта из системы МЕТА

Выполнение данного кода выведет в браузер форму для удаления (рис. 2.17).



ID

Рисунок 2.17 – Форма удаления

Далее будет представлено задание с загрузкой объекта системы МЕТА в базу данных на оперативной памяти Redis (рис. 2.18).

```
520 public void task9() throws Exception {
521     long time = 100000000L;
522     int expire = 180;
523     String[] types = new String[]{"C", "БНС", "НС"};
524     String costId = Util.s2s(request.getParameter("id"));
525     String foundOrCreated = null;
526     Obb ob = null;
527     if(!costId.equals("")) {
528         ob = Obb.fromBytes(Ob3.get(mains, redis, redis, (prefix + costId).getBytes()));
529         if(ob == null) {
530             ob = Obb.getOb(mains, costId);
531             if(ob == null) {
532                 foundOrCreated = "<div style=\"text-align:center;color:red;\"> " +
533                     "Искомый объект не существует в базе PostgreSQL</div>";
534             } else {
535                 Ob3.puts(mains, redis, (prefix + costId).getBytes(), Obb.toBytes(ob),
536                     time, nx: false, expire, oper: "");
537                 foundOrCreated = "<div style=\"text-align:center;color:blue;\"> " +
538                     "Объект изъят из базы PostgreSQL и записан в REDIS</div>";
539             }
540         } else {
541             foundOrCreated = "<div style=\"text-align:center;color:green;\">Объект найден</div>";
542         }
543     }
544     String text = String.format("<form method=\"get\" action=\"#\"> " +
545         "ID <input type=\"text\" name=\"id\" value=\"%s\"> " +
546         "<input type=\"submit\" value=\"Найти\"> " +
547         "<input type=\"hidden\" name=\"task\" value=\"9\"> " +
548         "</form>%s", costId, foundOrCreated == null ? "" : foundOrCreated, ob == null ? "" :
549         String.format("<table class=\"data\"> " +
550             "<tr><td>Название</td><td>ID номера</td><td>Тип стоимости</td></tr> " +
551             "<tr><td>%s</td><td>%s</td><td>%s</td></tr> " +
552             "</table>",
553             ob.getAt(id_at: 1000348),
554             ob.getAt(id_at: 1000350),
555             types[Integer.parseInt(ob.getAt(id_at: 1046222729))]);
556     printHtml(title: "Redis - Один", text);
557 }
```

Рисунок 2.18 – Загрузка объекта в Redis

Вывод задания представляет собой форму с возможностью ввода ID. Если его ввести нажать на кнопку «Найти», будет выведены данный найденного объекта в таблице (рис. 2.19).

ID  Найти

Объект изъят из базы PostgreSQL и записан в REDIS

Название	ID номера	Тип стоимости
СТАНДАРТ 2-Х МЕСТН. ВВ	102110001662	НС

Рисунок 2.19 – Форма и таблицы с данными найденного объекта

Далее будет рассмотрено добавление, поиск и удаление ключей из Redis (рис. 2.20 и рис. 2.21).

```

559 public void task10() throws Exception {
560     long time = 1000000000L;
561     int expire = 1;
562     String action = request.getParameter("action");
563     action = action == null || action.equals("") ? "find" : action;
564     String key = Util.s2s(request.getParameter("key"));
565     String value = null;
566     if(action.equals("find") && !key.equals("")) {
567         byte[] bytes = Ob3.get(mains, redis, redis, (prefix + key).getBytes());
568         value = bytes == null ? "" : new String(bytes, StandardCharsets.UTF_8);
569     } else if (action.equals("create")) {
570         value = request.getParameter("value");
571         Ob3.puts(mains, redis, (prefix + key).getBytes(), value.getBytes());
572     } else if (action.equals("delete")) {
573         //Ob3.del(mains, redis, prefix + key, redis);
574         byte[] bytes = Ob3.get(mains, redis, redis, (prefix + key).getBytes());
575         value = bytes == null ? "" : new String(bytes, StandardCharsets.UTF_8);
576         Ob3.puts(mains, redis, (prefix + key).getBytes(), value.getBytes(),
577                 time, nx: false, expire, oper: "");
578     }

```

Рисунок 2.20 – Первая часть кода последнего задания

```

579 String text = String.format("<form method=\"get\" action=\"#\"><table class=\"form\">" +
580     "<tr><td>Ключ</td><td><input type=\"text\" name=\"key\"></td></tr>" +
581     "<tr><td>Значение</td><td><input type=\"text\" name=\"value\"></td></tr>" +
582     "</table>" +
583     "<input type=\"submit\" value=\"Создать\">" +
584     "<input type=\"hidden\" name=\"task\" value=\"10\">" +
585     "<input type=\"hidden\" name=\"action\" value=\"create\">" +
586     "</form>%s" +
587     "<form method=\"get\" action=\"#\">" +
588     "Ключ <input type=\"text\" name=\"key\"> " +
589     "<input type=\"submit\" value=\"Найти\">" +
590     "<input type=\"hidden\" name=\"task\" value=\"10\">" +
591     "<input type=\"hidden\" name=\"action\" value=\"find\">" +
592     "</form>%s" +
593     "<form method=\"get\" action=\"#\">" +
594     "Ключ <input type=\"text\" name=\"key\"> " +
595     "<input type=\"submit\" value=\"Удалить\">" +
596     "<input type=\"hidden\" name=\"task\" value=\"10\">" +
597     "<input type=\"hidden\" name=\"action\" value=\"delete\">" +
598     "</form>%s",
599     action.equals("create") ? "<p style=\"text-align:center;color:green\">[" +
600     key + " : " + value + "] создано</p>" : "<br>",
601     !key.equals("") && action.equals("find") ?
602         !value.equals("")
603             ? "<p style=\"text-align:center;color:green\">[" +
604             key + " : " + value + "]</p>"
605             : "<p style=\"text-align:center;color:red\">Объект с ключом " +
606             key + " не найден.</p>"
607         : "<br>",
608     action.equals("delete")
609         ? !value.equals("")
610             ? "<p style=\"text-align:center;color:green\">[" +
611             key + " ] удален</p>"
612             : "<p style=\"text-align:center;color:red\">Объект с ключом " +
613             key + " не найден.</p>"
614         : "<br>");
615 printHtml( title: "Redis - Два", text);
616 }

```

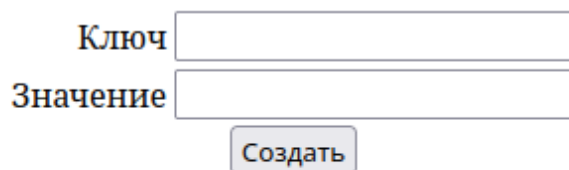
Рисунок 2.21 – Вторая часть кода последнего задания

Вывод метода в браузер представляет из себя 3 формы с созданием, поиском и удалением ключей. Ниже представлена первая из них (рис. 2.22).

Ключ	<input type="text" value="Hello"/>
Значение	<input type="text" value="Polytech"/>
<input type="button" value="Создать"/>	

Рисунок 2.22 – Форма создания ключа

После нажатия на кнопку «Создать» будет выведено сообщение об успешном создании ключа (рис. 2.23).



The form consists of two input fields. The first field is labeled 'Ключ' (Key) and the second is labeled 'Значение' (Value). Below the 'Значение' field is a button labeled 'Создать' (Create).

[Hello : Polytech] создано

Рисунок 2.23 – Сообщение о созданном ключе

Следующая форма позволяет искать ключи (рис. 2.24).



The form consists of an input field labeled 'Ключ' (Key) containing the text 'Hello', and a button labeled 'Найти' (Find).

Рисунок 2.24 – Форма поиска ключей

В случае успешного нахождения, выводится ключ и значение (рис. 2.25).



The form consists of an empty input field labeled 'Ключ' (Key) and a button labeled 'Найти' (Find).

[Hello : Polytech]

Рисунок 2.25 – Сообщение с найденным ключом

В противном случае (рис. 2.26):



The form consists of an empty input field labeled 'Ключ' (Key) and a button labeled 'Найти' (Find).

Объект с ключом Hellowin не найден.

Рисунок 2.26 – Сообщение о несуществовании ключа

Последняя форма позволяет удалять ключи (рис. 2.27).



Рисунок 2.27 – Форма удаления ключа

В случае успешного удаления, выводится соответствующее сообщение (рис. 2.28).



[Hello] удален

Рисунок 2.28 – Сообщение об успешном удалении ключа

В противном случае (рис. 2.29):



Объект с ключом Hellowin не найден.

Рисунок 2.29 – Сообщение о несуществовании ключа

Таким образом, были показаны запросы к системе META в реляционной базе данных PostgreSQL и базе данных NoSQL Redis, работающей в оперативной памяти.

**Вывод,** в данном разделе была показана реализация поставленной задачи, а именно, реализация приложения для работы с системой META в PostgreSQL и Redis.

## **ЗАКЛЮЧЕНИЕ**

В процессе выполнения практического проекта были использованы среда разработки IntelliJ IDEA Ultimate, язык программирования Java.

Были выполнены следующие задачи: проведён анализ предметной области, разработка визуализации программы, реализация программы для работы с таблицами баз данных.

Цель данной практической работы, а именно, разработка программного продукта для работы с системой META выполнена.

## СПИСОК ИСТОЧНИКОВ

- 1) Шилдт, Герберт. Java. Полное руководство, 10-е изд. : Пер. с англ. — СПб. : ООО «Диалектика», 2020. — 1488 с. : ил. — Парал. тит. англ.
- 2) Моргунов, Е. П. PostgreSQL. Основы языка SQL: учеб. пособие / Е. П. Моргунов; под ред. Е. В. Рогова, П. В. Лузанова. — СПб.: БХВ-Петербург, 2018. — 336 с.: ил.
- 3) Хомоненко А. Д. Базы данных: Учебник для высших учебных заведений/ В.М. Цыганков, М. Г. Мальцев, под ред. проф. А. Д. Хомоненко— СПб.: КОРОНА принт, 2002 — 672 с.
- 4) Дейт К. Дж. Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом “Вильямс”, 2005. — 1328 с.



## ПРИЛОЖЕНИЕ А

### Листинг А – файл web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app>
  <servlet>
    <servlet-name>InokovaServlet</servlet-name>
    <servlet-class>intern.InokovaServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>InokovaServlet</servlet-name>
    <url-pattern>/inokovaServlet</url-pattern>
  </servlet-mapping>

  <servlet>
    <servlet-name>EldarServlet</servlet-name>
    <servlet-class>intern.EldarServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>EldarServlet</servlet-name>
    <url-pattern>/eldar</url-pattern>
  </servlet-mapping>
</web-app>
```

## ПРИЛОЖЕНИЕ Б

### Листинг Б – файл EldarServlet.java

```
package intern;

import appt.meta3.Ob0;
import appt.meta3.Ob3;
import appt.meta3.Obb;
import appt.meta3.Util;
import org.apache.commons.lang3.math.NumberUtils;

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.nio.charset.StandardCharsets;
import java.util.*;
import java.util.function.BiConsumer;

public class EldarServlet extends HttpServlet {
    private PrintWriter out;
    private ResourceBundle mains;
    private HttpServletRequest request;
    private HttpServletResponse response;
    private final int myId = 1000360;
    private final String redis = "rev";
    private final String prefix = "Eldar";

    private static final Map<String, String> typeMap = new HashMap<>();
    private static final Map<String, String> yesnoMap = new HashMap<>();
    private static final Map<String, List<String>> pagesMap = new TreeMap<>();

    static {
        typeMap.put("", "-");
        typeMap.put("0", "Экскурсия");
        typeMap.put("1", "Билет");
        typeMap.put("2", "Спорт");
        typeMap.put("3", "Прокат");
        typeMap.put("4", "Услуга");
        typeMap.put("5", "СПА");
        typeMap.put("6", "Авиация");
        typeMap.put("8", "Концерт");

        yesnoMap.put("", "-");
        yesnoMap.put("1", "Да");
        yesnoMap.put("0", "Нет");

        pagesMap.put("PostgreSQL", Arrays.asList("Один", "Два",
            "Три", "Четыре", "Пять", "Шесть", "Семь", "Восемь"));
        pagesMap.put("Redis", Arrays.asList("Один", "Два", "Три", "Четыре"));
    }

    public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException {
        initialize(req, res);
        String task = request.getParameter("task");
        try {
```

```

        if ("7".equals(task)) {
            task7post();
        }
    } catch (Exception e) {
        printException(e);
    }
    out.flush();
}

```

```

public void doGet(HttpServletRequest req, HttpServletResponse res) throws IOException {
    initialize(req, res);
    try {
        int task = Integer.parseInt(request.getParameter("task"));
        if(task < 1 || task > 12) {
            throw new NumberFormatException();
        }
        if(task == 1) {
            task1();
        } else if(task == 2) {
            task2();
        } else if(task == 3) {
            task3();
        } else if(task == 4) {
            task4();
        } else if(task == 5) {
            task5();
        } else if(task == 6) {
            task6();
        } else if(task == 7) {
            task7();
        } else if(task == 8) {
            task8();
        } else if(task == 9) {
            task9();
        } else if(task == 10) {
            task10();
        } else if(task == 11) {
            task11();
        } else if(task == 12) {
            task12();
        }
    } catch (NumberFormatException e) {
        printHtml("Hello", "<h1>Привет Sirius!</h1>");
    } catch (Exception e) {
        printException(e);
    }
    out.flush();
}

```

```

public void initialize(HttpServletRequest req, HttpServletResponse res) throws IOException {
    request = req;
    response = res;
    response.setCharacterEncoding("UTF-8");
    request.setCharacterEncoding("UTF-8");
    out = res.getWriter();
    mains = ResourceBundle.getBundle("intern_main", new Locale("ru", "RU"));
}

```

```

public void printException(Exception e) {
    StackTraceElement[] stack = e.getStackTrace();
    String[] stackTrace = new String[stack.length];
}

```

```

    for(int i = 0; i < stack.length; ++i) {
        stackTrace[i] = stack[i].toString();
    }
    printHtml("Exception!", String.format("<h1>%s: %s</h1><div>%s</div>",
        e.getClass().getName(), e.getMessage(), String.join("<br>", stackTrace)));
}

public void task1() throws Exception {
    Map<String, String> regions = getRegions("100410000050");
    Obb filter = Ob0.createFilter(5);
    String regionId = request.getParameter("region");
    if(regionId == null || regionId.equals("")) {
        regionId = "100518301512";
    }
    Ob0.addCondition(filter, 1005101368, Ob0.ComparisonType.EQ, regionId);
    Obb[] cities = Ob0.getSrcObs(mains, filter, 0, 0);
    List<String> regionIds = intern.Utills.getKeysSortedByValue(regions, false);
    StringBuilder text = new StringBuilder("<form action=\\\"#\\\" method=\\\"get\\\">" +
        "<table class=\\\"form\\\"><tr><td>Регион</td><td><select name=\\\"region\\\">" +
    for(String key : regionIds) {
        text.append(String.format("<option%s value=\\\"%s\\\">%s</option>",
            key.equals(regionId) ? " selected" : "", key, regions.get(key)));
    }
    text.append("</select></td></tr></table><input type=\\\"submit\\\" value=\\\"Выбрать\\\">" +
        "<input type=\\\"hidden\\\" name=\\\"task\\\" value=\\\"1\\\"></form>");
    Arrays.sort(cities, Comparator.comparing((Obb ob) -> ob.getAt("1000098")));
    int i = 0;
    text.append("<table class=\\\"data\\\"><tr><td>№</td><td>ID</td><td>Город</td></tr>");
    for(Obb city : cities) {
        text.append(String.format("<tr><td>%d</td><td>%s</td><td>%s</td></tr>",
            ++i, city.id, Ob0.getAt(city, "1000098")));
    }
    text.append("</table>");
    printHtml("Города России", text.toString());
}

public Map<String, String> getRegions(String countryId) throws Exception {
    Obb filter = Ob0.createFilter(5);
    Ob0.addCondition(filter, 1000004, Ob0.ComparisonType.EQ, countryId);
    Ob0.addCondition(filter, 1005101368, Ob0.ComparisonType.NEQ, "");
    Obb[] cities = Ob0.getSrcObs(mains, filter, 0, 0);
    Map<String, String> regions = new TreeMap<>();
    for(Obb city : cities) {
        String regionId = Ob0.getAt(city, 1005101368);
        if(!regions.containsKey(regionId)) {
            regions.put(regionId, Ob0.getZn(mains, regionId, 1000098, 4));
        }
    }
    return regions;
}

public void task2() throws Exception {
    String countryId = request.getParameter("country");
    if(countryId == null || countryId.equals("")) {
        countryId = "100410000050";
    }
    Obb filter = Ob0.createFilter(5);
    Ob0.addCondition(filter, 1000004, Ob0.ComparisonType.EQ, countryId);
    Ob0.addCondition(filter, 1000101, Ob0.ComparisonType.EQ, "Да");
    Obb[] cities = Ob0.getSrcObs(mains, filter, 0, 0);
    Obb[] countries = Ob0.getSrcObs(mains, Ob0.createFilter(4), 0, 0);

```

```

Map<String, String> regions = getRegions(countryId);
Map<String, Set<String>> distribution = regionDistribute(cities);
List<String> regionIds = intern.Utils.getKeysSortedByValue(regions, false);
Arrays.sort(countries, Comparator.comparing((Obb ob) -> ob.getAt("1000000")));
StringBuilder text = new StringBuilder();
text.append("<form action=\"#\" method=\"get\"><table class=\"form\"><tr><td>\" +
    \"Страна</td><td><select name=\"country\">");
for(Obb country : countries) {
    text.append(String.format("<option%s value=\"%s\">%s</option>",
        country.id.equals(countryId) ? " selected" : "",
        country.id, country.getAt("1000000")));
}
text.append("</select></td></tr></table><input type=\"submit\" value=\"Выбрать\">\" +
    "<input type=\"hidden\" name=\"task\" value=\"2\"></form>");
if(regionIds.isEmpty()) {
    text.append("<h4 style=\"text-align:center;color:grey;\">\" +
        \"Данная страна не поддерживает разбиение на регионы</h4>");
} else {
    text.append("<table class=\"data\"><tr><td>№</td><td>Область</td><td>Города</td></tr>");
    int i = 0;
    for (String key : regionIds) {
        if (regions.containsKey(key) && distribution.containsKey(key)) {
            text.append(String.format("<tr><td>%d</td><td>%s</td><td>%s</td></tr>",
                ++i, regions.get(key),
                String.join(", ", distribution.get(key))));
        }
    }
    text.append("</table>");
}
printHtml("Города России", text.toString());
}

public Map<String, Set<String>> regionDistribute(Obb[] cities) {
    Map<String, Set<String>> distribution = new TreeMap<>();
    for(Obb city : cities) {
        String regionId = Ob0.getAt(city, 1005101368);
        if(!distribution.containsKey(regionId)) {
            distribution.put(regionId, new TreeSet<>());
        }
        distribution.get(regionId).add(Ob0.getAt(city, 1000098));
    }
    return distribution;
}

public void task3() throws Exception {
    response.setContentType("application/json; charset=UTF-8");
    String birthday = request.getParameter("bd");
    if(birthday == null || birthday.equals("")) {
        birthday = "01.01.1990";
    }
    Obb filter = Ob0.createFilter(23);
    Ob0.addCondition(filter, 1000152, Ob0.ComparisonType.GT, birthday);
    Obb[] tourists = Ob0.getSrcObs(mains, filter, 0, 0);
    Arrays.sort(tourists, Comparator.comparing(
        (Obb ob) -> ob == null ? "" : ob.getAt("1000144")));

    out.print("[");
    int i = 0;
    for(Obb tourist : tourists) {
        out.printf("{ \"id\": \"%s\", \"nm\": \"%s %s %s\", \"bd\": \"%s\", \"trs\": [",
            tourist == null ? "null" : tourist.id,

```

```

        Ob0.getAt(tourist, 1000144),
        Ob0.getAt(tourist, 1000146),
        Ob0.getAt(tourist, 1000147),
        Ob0.getAt(tourist, 1000152));
String[] tours = Ob0.getAt(tourist, 1023422081).split("#");
int j = 0;
for(String tour : tours) {
    out.printf("\t%s\t%s", tour, ++j < tours.length ? "," : "");
}
out.printf("\t\t%s", ++i < tourists.length ? "," : "");
}
out.print("\n");
}

public void task4() throws Exception {
    String agentName = request.getParameter("agent");
    String categoryName = request.getParameter("category");
    if(agentName == null || agentName.equals("")) {
        agentName = "КонтрАг";
    }
    if(categoryName == null || categoryName.equals("")) {
        categoryName = "Концерты";
    }
    Obb filter = Ob0.createFilter(36);
    Ob0.addCondition(filter, new int[]{1036922797, 1317100000},
        Ob0.ComparisonType.EQ, agentName);
    Ob0.addCondition(filter, new int[]{1036900082, 1162100000},
        Ob0.ComparisonType.EQ, categoryName);
    Obb[] costs = Ob0.getSrcObs(mains, filter, 0, 0);
    Obb[] agents = Ob0.getSrcObs(mains, Ob0.createFilter(317), 0, 0);
    Obb[] categories = Ob0.getSrcObs(mains, Ob0.createFilter(162), 0, 0);
    Arrays.sort(costs, (Obb left, Obb right) -> right.data_n.compareTo(left.data_n));
    costs = Arrays.copyOfRange(costs, Math.max(0, costs.length - 50), costs.length);
    Arrays.sort(costs, Comparator.comparing((Obb ob) -> ob.getAt("1036423021")));
    Arrays.sort(agents, Comparator.comparing((Obb ob) -> ob.getAt("1317100000")));
    StringBuilder text = new StringBuilder();
    text.append("<form action=\"\t#\t\" method=\"\tget\t\"><table class=\"\tform\t\"><tr><td>\" +
        "Контрагент</td><td><select name=\"\tagent\t\">");
    for(Obb agent : agents) {
        String an = agent.getAt("1317100000");
        text.append(String.format("<option%s>%s</option>",
            an.equals(agentName) ? " selected" : "", an));
    }
    text.append("</select></td></tr><tr><td>Категория</td><td><select name=\"\tcategory\t\">");
    for(Obb category : categories) {
        String cn = category.getAt("1162100000");
        text.append(String.format("<option%s>%s</option>",
            cn.equals(categoryName) ? " selected" : "", cn));
    }
    text.append("</select></td><tr></table><input type=\"\tsubmit\t\" value=\"\tВыбрать\t\">\" +
        "<input type=\"\thidden\t\" name=\"\task\t\" value=\"\t4\t\"></form>\" +
        "<table class=\"\tdata\t\"><tr><td>№</td><td>Название</td>\" +
        "<td>Категория</td><td>Адрес</td></tr>");
    int i = 0;
    for(Obb cost : costs) {
        text.append(String.format("<tr><td>%d</td><td>%s</td><td>%s</td><td>%s</td></tr>",
            ++i, cost.getAt("1036423021"),
            yesnoMap.get(cost.getAt("1036200042")),
            cost.getAt("1036410028")));
    }
    printHtml("Эксперсии", text.append("</table>").toString());
}

```

```
}
```

```
public void task5() throws Exception {
    Obb[] costs = Ob0.getSrcObs(mains, Ob0.createFilter(36), 0, 0);
    List<Obb> list = Arrays.asList(costs);
    List<Obb> array = new ArrayList<>(list);
    List<Obb> linked = new LinkedList<>(list);
    StringBuilder text = new StringBuilder();
    BiConsumer<List<Obb>, String> consumer = (List<Obb> lst, String listType) -> {
        int i = 0;
        text.append("<div>");
        long time = System.currentTimeMillis();
        for(Obb item : lst) {
            text.append(String.format("[%d : %s]%s", ++i, item.id, i == lst.size() ? "" : ", "));
        }
        time = System.currentTimeMillis() - time;
        text.append(String.format("<br>%s - %d ms</div>", listType, time));
    };
    consumer.accept(array, "ArrayList");
    consumer.accept(linked, "LinkedList");
    printHtml("Временное сравнение", text.toString());
}
```

```
public void task6() throws Exception {
    String type = request.getParameter("type");
    String code = request.getParameter("code");
    String address = request.getParameter("address");
    String category = request.getParameter("category");
    String agentId = request.getParameter("agent");

    type = NumberUtils.isNumber(type) && Integer.parseInt(type) >= 0
        && Integer.parseInt(type) <= 8 && Integer.parseInt(type) != 7 ? type : "";
    code = code == null ? "" : code;
    address = address == null ? "" : address;
    category = category == null || !(category.equals("1") || category.equals("0")) ? "" : category;
    agentId = agentId == null ? "" : agentId;

    Obb filter = Ob0.createFilter(36);
    if(!type.equals("")) {
        Ob0.addCondition(filter, 1036200042, Ob0.ComparisonType.EQ, type);
    }
    if(!code.equals("")) {
        Ob0.addCondition(filter, 1036423021, Ob0.ComparisonType.EQ, code);
    }
    if(!address.equals("")) {
        Ob0.addCondition(filter, 1036410028, Ob0.ComparisonType.EQ, address);
    }
    if(!category.equals("")) {
        Ob0.addCondition(filter, 1162200125, Ob0.ComparisonType.EQ, category);
    }
    if(!agentId.equals("")) {
        Ob0.addCondition(filter, 1036922797, Ob0.ComparisonType.EQ, agentId);
    }
    Obb[] costs = Ob0.getSrcObs(mains, filter, 0, 0);
    Obb[] agents = Ob0.getSrcObs(mains, Ob0.createFilter(317), 0, 0);
    Arrays.sort(agents, Comparator.comparing((Obb ob) -> ob.getAt("1317100000")));

    Map<String, String> agentMap = new HashMap<>();
    agentMap.put("", "-");
    for(Obb agent : agents) {
        agentMap.put(agent.id, agent.getAt("1317100000"));
    }
}
```



```

}

StringBuilder text = new StringBuilder();
text.append("<form action=\"#{\" method=\"get\"><table class=\"form\"><tr><td>\" +
    \"Тип</td><td><select name=\"type\">\"");
for(String key : typeMap.keySet()) {
    text.append(String.format("<option%s value=\"%s\">%s</option>\",
        type.equals(key) ? \" selected\" : \"\", key, typeMap.get(key)));
}
text.append("</select></td></tr><tr><td>Код</td>\" +
    \"<td><input name=\"code\" type=\"text\" value=\"\"></td></tr>\" +
    \"<tr><td>Адрес</td><td><input name=\"address\" type=\"text\" value=\"\">\" +
    \"</td></tr><tr><td>Категория</td><td><select name=\"category\">\"");
for(String key : yesnoMap.keySet()) {
    text.append(String.format("<option%s value=\"%s\">%s</option>\",
        category.equals(key) ? \" selected\" : \"\", key, yesnoMap.get(key)));
}
text.append("</select></td></tr><tr><td>Контрагент</td><td><select name=\"agent\">\"");
for(String key : agentMap.keySet()) {
    text.append(String.format("<option%s value=\"%s\">%s</option>\",
        agentId.equals(key) ? \" selected\" : \"\", key, agentMap.get(key)));
}
text.append("</select></td></tr></table><input type=\"submit\" value=\"Выбрать\">\" +
    \"<input type=\"hidden\" name=\"task\" value=\"6\"></form>\" +
    \"<table class=\"data\"><tr><td>№</td><td>Название</td><td>Тип</td>\" +
    \"<td>Адрес</td><td>Да/Нет</td><td>Контрагент</td></tr>\"");
Arrays.sort(costs, Comparator.comparing((Obb ob) -> ob.getAt("1036423021")));
int i = 0;
for(Obb cost : costs) {
    text.append(String.format("<tr><td>%d</td><td>%s</td><td>%s</td>\" +
        \"<td>%s</td><td>%s</td><td>%s</td></tr>\",
        ++i, cost.getAt("1036423021"),
        typeMap.get(cost.getAt("1036200042")),
        cost.getAt("1036410028"),
        yesnoMap.get(cost.getAt("1162200125")),
        agentMap.get(cost.getAt("1036922797"))));
}
printHtml("Экскурсии", text.append("</table>").toString());
}

```

```

public void task7() throws Exception {
    Map<String, String> regions = new HashMap<>();
    Map<String, String> partners = new HashMap<>();
    regionsAndPartners(regions, partners);

    StringBuilder text = new StringBuilder();
    text.append("<form method=\"post\" action=\"#{\"><table class=\"form\">\" +
        \"<tr><td>Название</td><td><input type=\"text\" name=\"name\"></td></tr>\" +
        \"<tr><td>Описание</td><td><textarea name=\"desc\"></textarea></td></tr>\" +
        \"<tr><td>Регион</td><td><select name=\"region\">\"");
    List<String> regionIds = intern.Utils.getKeysSortedByValue(regions, false);
    for(String regionId : regionIds) {
        text.append(String.format("<option value=\"%s\">%s</option>\",
            regionId, regions.get(regionId)));
    }
    text.append("</select></td></tr>\" +
        \"<tr><td>Доп. оплата</td><td><input type=\"text\" name=\"cost\"></td></tr>\" +
        \"<tr><td>Бронирование у партнера</td><td><select name=\"partner\">\"");
    for(String partnerId : partners.keySet()) {
        text.append(String.format("<option value=\"%s\">%s</option>\",
            partnerId, partners.get(partnerId)));
    }
}

```



```

    }
    text.append("</select></td></tr>" +
        "<tr><td>Тип</td><td><select name=\"type\">");
    for(String typeId : typeMap.keySet()) {
        text.append(String.format("<option value=\"%s\">%s</option>",
            typeId, typeMap.get(typeId)));
    }
    text.append("</select></td></tr>" +
        "</table><input type=\"hidden\" name=\"task\" value=\"7\">" +
        "<input type=\"submit\" value=\"Создать\"></form>");
    printDescs(text, regions, partners);
}

```

```

public void task7post() throws Exception {
    String name = request.getParameter("name");
    String desc = request.getParameter("desc");
    String region = request.getParameter("region");
    String cost = request.getParameter("cost");
    String partner = request.getParameter("partner");
    String type = request.getParameter("type");

    name = name == null ? "" : name;
    desc = desc == null ? "" : desc;
    region = region == null ? "-" : region;
    cost = !NumberUtils.isNumber(cost) ? "" : cost;
    partner = partner == null ? "-" : partner;
    type = !typeMap.containsKey(type) ? "-" : type;

    if(!name.equals("") && !desc.equals("") && !region.equals("-") &&
        !cost.equals("") && !partner.equals("-") && !type.equals("-")) {
        Obb ob = new Obb(506);
        ob.id_user = myId;
        Ob0.addAt(ob, "1506410000", name);
        Ob0.addAt(ob, "1506410282", desc);
        Ob0.addAt(ob, "1506923461", region);
        Ob0.addAt(ob, "1506223120", cost);
        Ob0.addAt(ob, "1506910189", partner);
        Ob0.addAt(ob, "1506310181", type);
        Ob0.addOb(mains, ob);
    }
    response.sendRedirect("/eldar?task=7");
}

```

```

public void task8() throws Exception {
    String descId = request.getParameter("did");
    descId = descId == null ? "" : descId;
    if(!descId.equals("")) {
        Ob0.delOb(mains, descId, myId);
    }
    Map<String, String> regions = new HashMap<>();
    Map<String, String> partners = new HashMap<>();
    regionsAndPartners(regions, partners);
    StringBuilder text = new StringBuilder();
    text.append("<form method=\"get\" action=\"#\>" +
        "ID <input type=\"text\" name=\"did\">" +
        "<input type=\"submit\" value=\"Удалить\">" +
        "<input type=\"hidden\" name=\"task\" value=\"8\"></form>");
    printDescs(text, regions, partners);
}

```

```

public void regionsAndPartners(Map<String, String> regions,

```

```

        Map<String, String> partners) throws Exception {
    Obb filter = Ob0.createFilter(5);
    Ob0.addCondition(filter, 1000004, Ob0.ComparisonType.EQ, "100410000050");
    Ob0.addCondition(filter, 1005101368, Ob0.ComparisonType.NEQ, "");
    Obb[] cities = Ob0.getSrcObs(mains, filter, 0, 0);
    Obb[] prtns = Ob0.getSrcObs(mains, Ob0.createFilter(158), 0, 0);
    regions.put("", "-");
    partners.put("", "-");
    for(Obb city : cities) {
        regions.put(city.id, city.getAt("1000098"));
    }
    for(Obb partner : prtns) {
        partners.put(partner.id, partner.getAt("1001211"));
    }
}

public void printDescs(StringBuilder text, Map<String, String> regions,
        Map<String, String> partners) throws Exception {
    Obb[] desc = Ob0.getSrcObs(mains, Ob0.createFilter(506), 0, 0);
    text.append("<table class=\"data\"><tr><td>№</td><td>ID</td><td>Название</td>" +
        "<td>Описание</td><td>Регион</td>" +
        "<td>Доп. оплаты</td><td>Бронирование у парнера</td><td>Тип</td></tr>");
    int i = 0;
    for(Obb desc : desc) {
        text.append(String.format("<tr><td>%d</td><td>%s</td><td>%s</td><td>%s</td>" +
            "<td>%s</td><td>%s</td><td>%s</td><td>%s</td></tr>",
            ++i, (desc.id_user == myId ? "<b style=\"color:green;\">" + desc.id + "<b>" : desc.id),
            desc.getAt("1506410000"),
            desc.getAt("1506410282"),
            regions.get(desc.getAt("1506923461")),
            desc.getAt("1506223120"),
            partners.get(desc.getAt("1506910189")),
            typeMap.get(desc.getAt("1506310181"))));
    }
    text.append("</table>");
    printHtml("Описания экскурсий", text.toString());
}

public void task9() throws Exception {
    long time = 100000000L;
    int expire = 180;
    String[] types = new String[]{"C", "БНС", "НС"};
    String costId = Util.s2s(request.getParameter("id"));
    String foundOrCreated = null;
    Obb ob = null;
    if(!costId.equals("")) {
        ob = Ob0.fromBytes(Ob3.get(mains, redis, redis, (prefix + costId).getBytes()));
        if(ob == null) {
            ob = Ob0.getOb(mains, costId);
            if(ob == null) {
                foundOrCreated = "<div style=\"text-align:center;color:red;\">" +
                    "Искомый объект не существует в базе PostgreSQL</div>";
            } else {
                Ob3.puts(mains, redis, (prefix + costId).getBytes(), Ob0.toBytes(ob),
                    time, false, expire, "");
                foundOrCreated = "<div style=\"text-align:center;color:blue;\">" +
                    "Объект изъят из базы PostgreSQL и записан в REDIS</div>";
            }
        } else {
            foundOrCreated = "<div style=\"text-align:center;color:green;\">Объект найден</div>";
        }
    }
}

```

```

}
String text = String.format("<form method=\\"get\\" action=\\\\"#\\">" +
    "ID <input type=\\"text\\" name=\\"id\\" value=\\\\"%s\\">" +
    "<input type=\\"submit\\" value=\\"Найти\\">" +
    "<input type=\\"hidden\\" name=\\"task\\" value=\\"9\\">" +
    "</form>%s%s", costId, foundOrCreated == null ? "" : foundOrCreated, ob == null ? "" :
    String.format("<table class=\\"data\\">" +
        "<tr><td>Название</td><td>ID номера</td><td>Тип стоимости</td></tr>" +
        "<tr><td>%s</td><td>%s</td><td>%s</td></tr>" +
        "</table>",
        ob.getAt(1000348),
        ob.getAt(1000350),
        types[Integer.parseInt(ob.getAt(1046222729))]);
printHtml("Redis - Один", text);
}

```

```

public void task10() throws Exception {
    long time = 100000000L;
    int expire = 1;
    String action = request.getParameter("action");
    action = action == null || action.equals("") ? "find" : action;
    String key = Util.s2s(request.getParameter("key"));
    String value = null;
    if(action.equals("find") && !key.equals("")) {
        byte[] bytes = Ob3.get(mains, redis, redis, (prefix + key).getBytes());
        value = bytes == null ? "" : new String(bytes, StandardCharsets.UTF_8);
    } else if (action.equals("create")) {
        value = request.getParameter("value");
        Ob3.puts(mains, redis, (prefix + key).getBytes(), value.getBytes());
    } else if (action.equals("delete")) {
        //Ob3.del(mains, redis, prefix + key, redis);
        byte[] bytes = Ob3.get(mains, redis, redis, (prefix + key).getBytes());
        value = bytes == null ? "" : new String(bytes, StandardCharsets.UTF_8);
        Ob3.puts(mains, redis, (prefix + key).getBytes(), "".getBytes(),
            time, false, expire, "");
    }
    String text = String.format("<form method=\\"get\\" action=\\\\"#\\"><table class=\\"form\\">" +
        "<tr><td>Ключ</td><td><input type=\\"text\\" name=\\"key\\"></td></tr>" +
        "<tr><td>Значение</td><td><input type=\\"text\\" name=\\"value\\"></td></tr>" +
        "</table>" +
        "<input type=\\"submit\\" value=\\"Создать\\">" +
        "<input type=\\"hidden\\" name=\\"task\\" value=\\"10\\">" +
        "<input type=\\"hidden\\" name=\\"action\\" value=\\"create\\">" +
        "</form>%s" +
        "<form method=\\"get\\" action=\\\\"#\\">" +
        "Ключ <input type=\\"text\\" name=\\"key\\">" +
        "<input type=\\"submit\\" value=\\"Найти\\">" +
        "<input type=\\"hidden\\" name=\\"task\\" value=\\"10\\">" +
        "<input type=\\"hidden\\" name=\\"action\\" value=\\"find\\">" +
        "</form>%s" +
        "<form method=\\"get\\" action=\\\\"#\\">" +
        "Ключ <input type=\\"text\\" name=\\"key\\">" +
        "<input type=\\"submit\\" value=\\"Удалить\\">" +
        "<input type=\\"hidden\\" name=\\"task\\" value=\\"10\\">" +
        "<input type=\\"hidden\\" name=\\"action\\" value=\\"delete\\">" +
        "</form>%s",
        action.equals("create") ? "<p style=\\"text-align:center;color:green\\">[" +
            key + " : " + value + "]" создано</p>" : "<br>",
        !key.equals("") && action.equals("find") ?
        !value.equals("")
        ? "<p style=\\"text-align:center;color:green\\">["

```

```

        + key + " : " + value + "]/p>"
        : "<p style=\"text-align:center;color:red\">Объект с ключом "
        + key + " не найден.</p>"
        : "<br>",
    action.equals("delete")
    ? !value.equals("")
    ? "<p style=\"text-align:center;color:green\">["
    + key + "]" удален</p>"
    : "<p style=\"text-align:center;color:red\">Объект с ключом "
    + key + " не найден.</p>"
    : "<br>");
    printHtml("Redis - Два", text);
}

public void task11() throws Exception {
    Obb[] obs = Ob0.getSrcObs(mains, Ob0.createFilter(36), 0, 0);
    List<String> result = new ArrayList<>(obs.length);
    String action = Util.s2s(request.getParameter("action"));
    action = action.equals("") ? "postgre" : action;
    long time = -1;
    if(action.equals("postgre")) {
        time = catchTime() -> {
            for(Obb ob : obs) {
                result.add(Ob0.getOb(mains, ob.id).id);
            }
        };
    } else if(action.equals("redis")) {
        time = catchTime() -> {
            for(Obb ob : obs) {
                Obb temp = Ob0.fromBytes(Ob3.get(mains, redis, redis, (prefix + ob.id).getBytes()));
                if(temp != null) {
                    result.add(temp.id);
                } else {
                    result.clear();
                    break;
                }
            }
        };
    } else if(action.equals("in_redis")) {
        long time0 = 1000000000L;
        int expire = 3600;
        for(Obb ob : obs) {
            Ob3.puts(mains, redis, (prefix + ob.id).getBytes(), Ob0.toBytes(ob),
                time0, false, expire, "");
        }
    } else {
        throw new Exception("Плохой аргумент");
    }
    StringBuilder text = new StringBuilder();
    text.append("<div style=\"text-align:center;\"><p> +
        "<a href=\"?task=11&action=postgre\">Выбрать из PostgreSQL</a> | " +
        "<a href=\"?task=11&action=redis\">Выбрать из Redis</a> | " +
        "<a href=\"?task=11&action=in_redis\">Заполнить Redis</a></p>");
    text.append(time == -1
        ? String.format("<p style=\"color:green\">%d объектов добавлено в Redis.</p>",
            obs.length)
        : !result.isEmpty()
        ? String.format("<p>Время выборки из %s: %d мс</p>",
            action.equals("postgre") ? "PostgreSQL" : "Redis", time)
        : "<h1 style=\"text-align:center;\"> +
        "<a href=\"?task=11&action=in_redis\">Заполните</a> Redis</h1>");
}

```

```

text.append("</div>");
if(!result.isEmpty()) {
    text.append(String.join(" ", result));
}
printHtml("Redis - Три", text.toString());
}

public long catchTime(Procedure procedure) throws Exception {
    long time = System.currentTimeMillis();
    procedure.run();
    return System.currentTimeMillis() - time;
}

public interface Procedure {
    void run() throws Exception;
}

public void task12() throws Exception {
    printHtml("Redis - Четыре",
        "<div style=\"border:1px solid #ccc; width:900px; margin:0px auto; padding:15px;\"> +
        \"Целесообразно применять Redis в онлайн-магазинах для корзины, \" +
        \"в онлайн-играх по типу шахмат для хранения ходов и состояния шахматной доски, \" +
        \"в стриминговых платформах для буферизации видеопотоков...\" +
        "</div>");
}

public void printHtml(String title, String text) {
    response.setContentType("text/html; charset=UTF-8");
    String task = request.getParameter("task");
    StringBuilder output = new StringBuilder();
    output.append(String.format("<!doctype html><html><head><title>%s</title><style>\" +
        \"table { margin: 0px auto; }\" +
        \"table.form tr td:first-child { text-align:right; }\" +
        \"table.form tr td:last-child { text-align:left; }\" +
        \"table.data { border: solid 1px #ccc; border-spacing: 3px;\" +
        \"border-collapse: collapse; margin-bottom:10px; }\" +
        \"table.data tr:first-child { font-weight:bold; }\" +
        \"table.data td { border: solid 1px #ccc; padding: 5px; }\" +
        \"form { margin: 0px auto; margin-bottom:20px; text-align:center;\" +
        \"padding:10px; }\" +
        "</style></head>\" +
        "<body><div style=\"text-align:center;margin-bottom:20px;\">\" +
        "<table class=\"form\">\", title));
    int tasks = 0;
    for(String key : pagesMap.keySet()) {
        output.append(String.format("<tr>\" +
            "<td style=\"font-weight:bold;color:#072d78;padding-right:15px;\">\" +
            \"%s</td><td>\", key));
        List<String> numbers = pagesMap.get(key);
        for(int i = 1, n = numbers.size(); i <= n; ++i) {
            if(String.valueOf(tasks + i).equals(task)) {
                output.append(String.format("<b style=\"color:grey\">%s</b>%s\",
                    numbers.get(i - 1), i < n ? \" | \" : ""));
            } else {
                output.append(String.format("<a href=\"?task=%d\">%s</a>%s\",
                    tasks + i, numbers.get(i - 1), i < n ? \" | \" : ""));
            }
        }
        output.append("</td></tr>");
        tasks += numbers.size();
    }
}

```

```
    output.append(String.format("</table></div>%s</body></html>", text));  
    out.print(output);  
  }  
}
```