МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

по дисциплине «Введение в нереляционные системы управления базами данных»

Тема: Сервис управления сетью коммерческих автоматизированных складов.

Студенты гр. 1303 Новак П. И

Королева П. А.

Иевлев Е.А.

Преподаватель Заславский М.М.

Санкт-Петербург
2024

ЗАДАНИЕ

Студенты

Новак П. И.

Королева П. А.

Иевлев Е. А.

Группа 1303

Тема проекта: Разработка сервиса управления сетью коммерческих автоматизированных складов.

Исходные данные:

Необходимо реализовать сервиса управления сетью коммерческих автоматизированных складов для СУБД (ArangoDB).

Содержание пояснительной записки:

«Содержание»

«Введение»

«Качественные требования к решению»

«Сценарий использования»

«Модель данных»

«Разработка приложения»

«Вывод»

«Приложение»

Предполагаемый объем пояснительной записки:

Не менее 10 страниц.

Дата выдачи задания:

Дата сдачи реферата:

Дата защиты реферата:

Студенты гр. 4303	_____	Новак П. И.
	_____	Королева П. А.
	_____	Иевлев Е. А.
Преподаватель	_____	Заславский М.М.

АННОТАЦИЯ

В рамках данного курса предполагалось разработать какой-либо сервис в команде на одну из поставленных тем. Была выбрана тема создания сервиса управления сетью коммерческих автоматизированных складов для СУБД (ArangoDB). В приложении было важно реализовать эффективное управление данными о складе, товарах, заказах и клиентских запросах. Основной упор - повышение производительности складских операций, оптимизация процессов хранения и автоматизация учёта. Найти исходный код и всю дополнительную информацию можно по ссылке: https://github.com/moevm/nosql2h24-store.

ANNOTATION

Within the framework of this course it was supposed to develop some service in a team on one of the assigned topics. The topic of creating a service for managing a network of commercial automated warehouses for DBMS (ArangoDB) was chosen. In the application it was important to realize efficient management of data about warehouse, goods, orders and customer requests. The main focus is to increase the productivity of warehouse operations, optimize storage processes and automate accounting. Find the source code and all additional information at: https://github.com/moevm/nosql2h24-store.

Оглавление

1. Введение

Цель работы – создать сервис управления сетью коммерческих автоматизированных складов, который позволит автоматизировать процессы учёта, хранения и обработки заказов. Система должна быть направлена на повышение эффективности операций и оптимизацию использования складских ресурсов.

2. Качественные требования к решению

Требуется разработать приложение с использованием СУБД – ArangoDB, с возможностью массового импорта-экспорта.

3. Сценарии использования

Макеты UI

1. Диаграмма Use Case (Рис. 1).

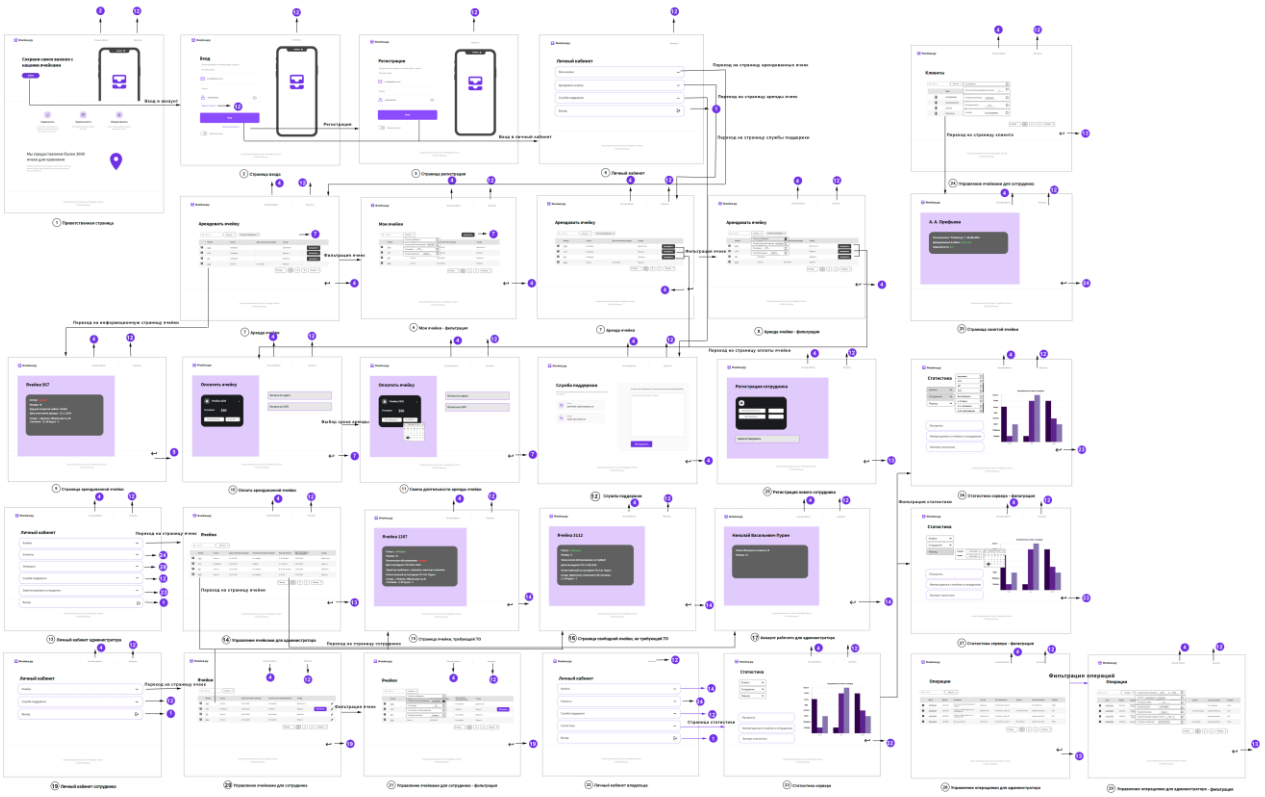

Рисунок 1. Диаграмма Use Case.

2. Импорт данных (массовая загрузка пользователем данных в программу) (Рис. 2).

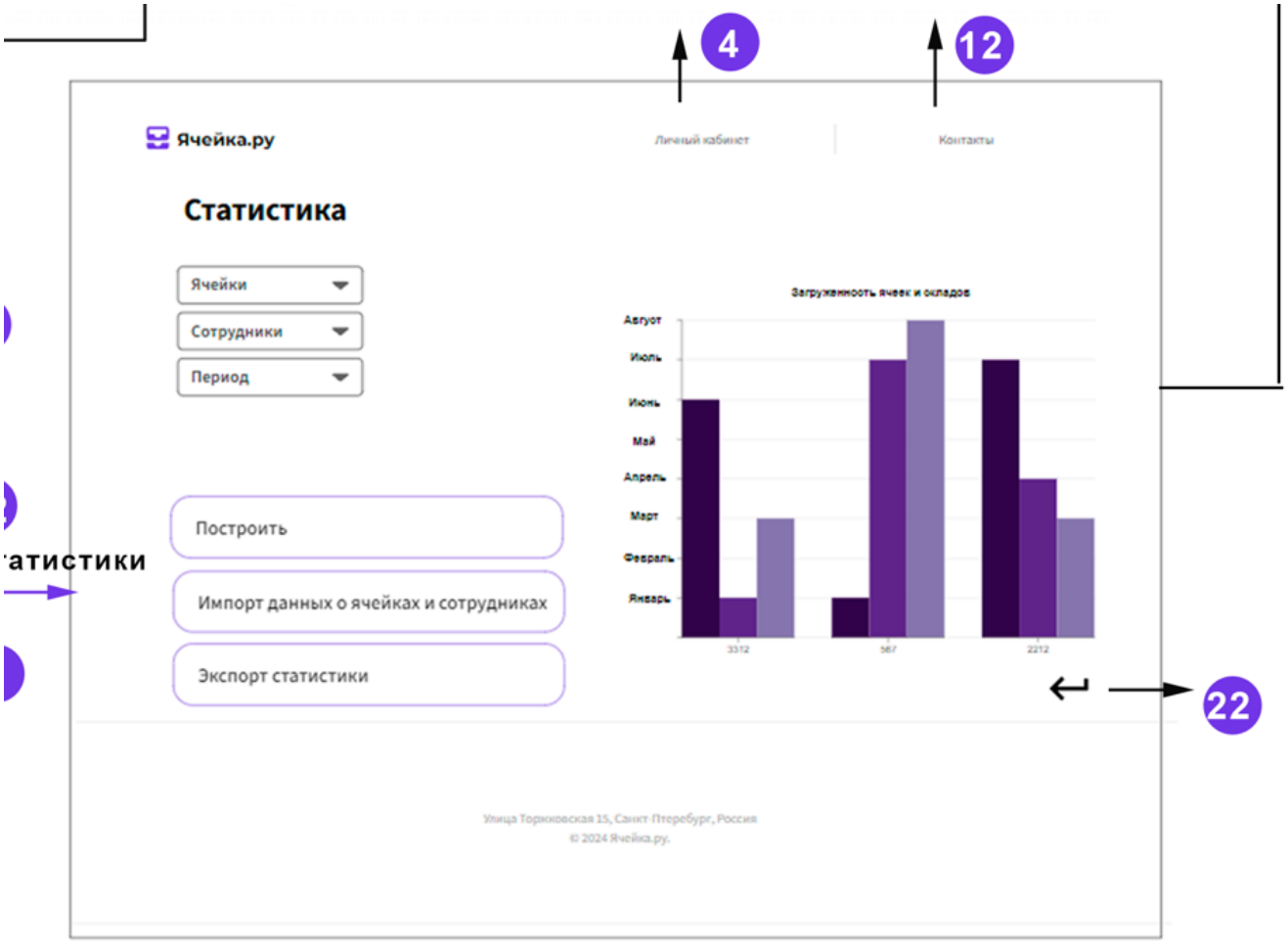

Рисунок 2. Массовый импорт данных

3. Импорт данных (загрузка пользователем данных в программу вручную) (Рис. 3).

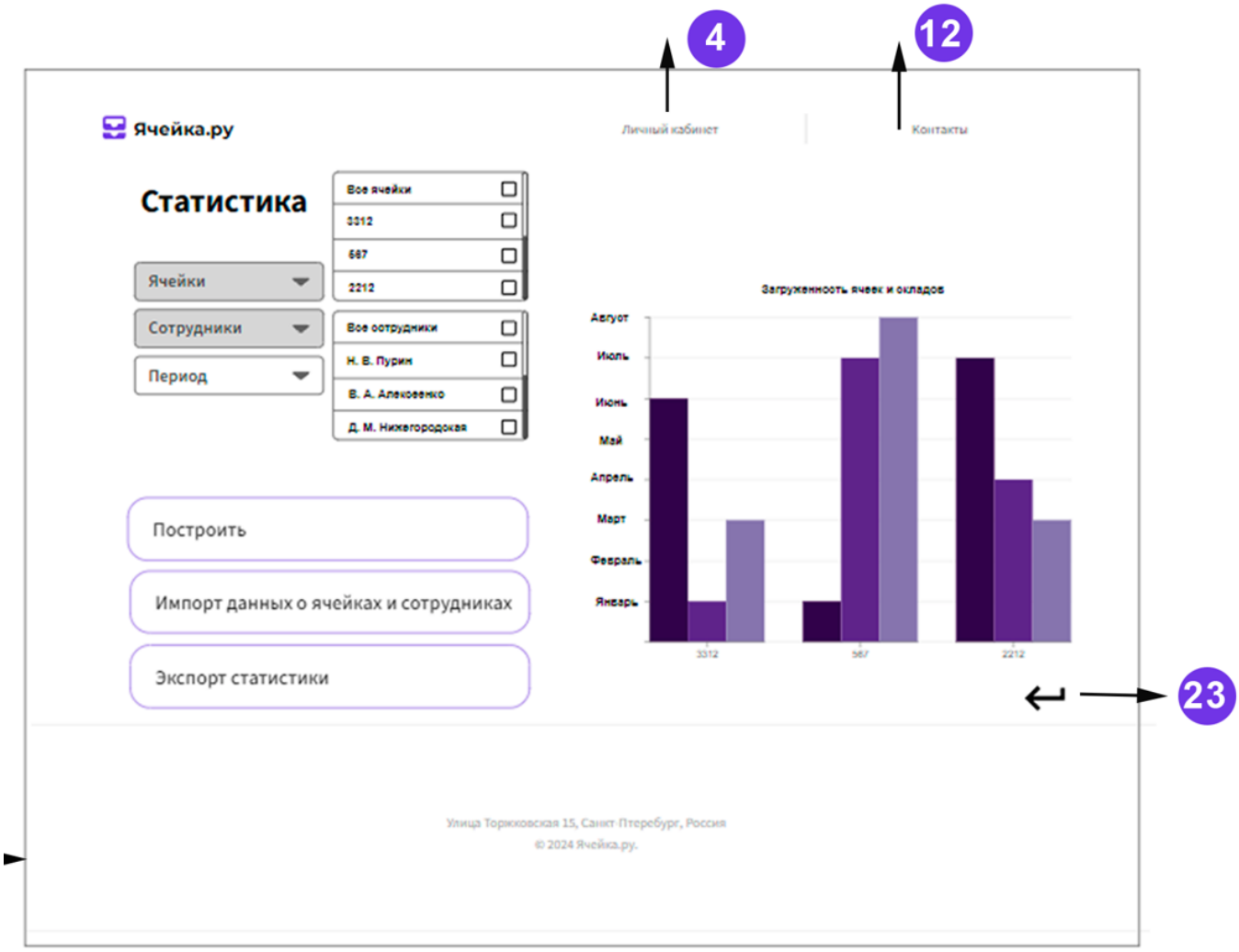

Рисунок 3. Загрузка пользователем данных в программу вручную.

4. Представление данных (Рис. 4).

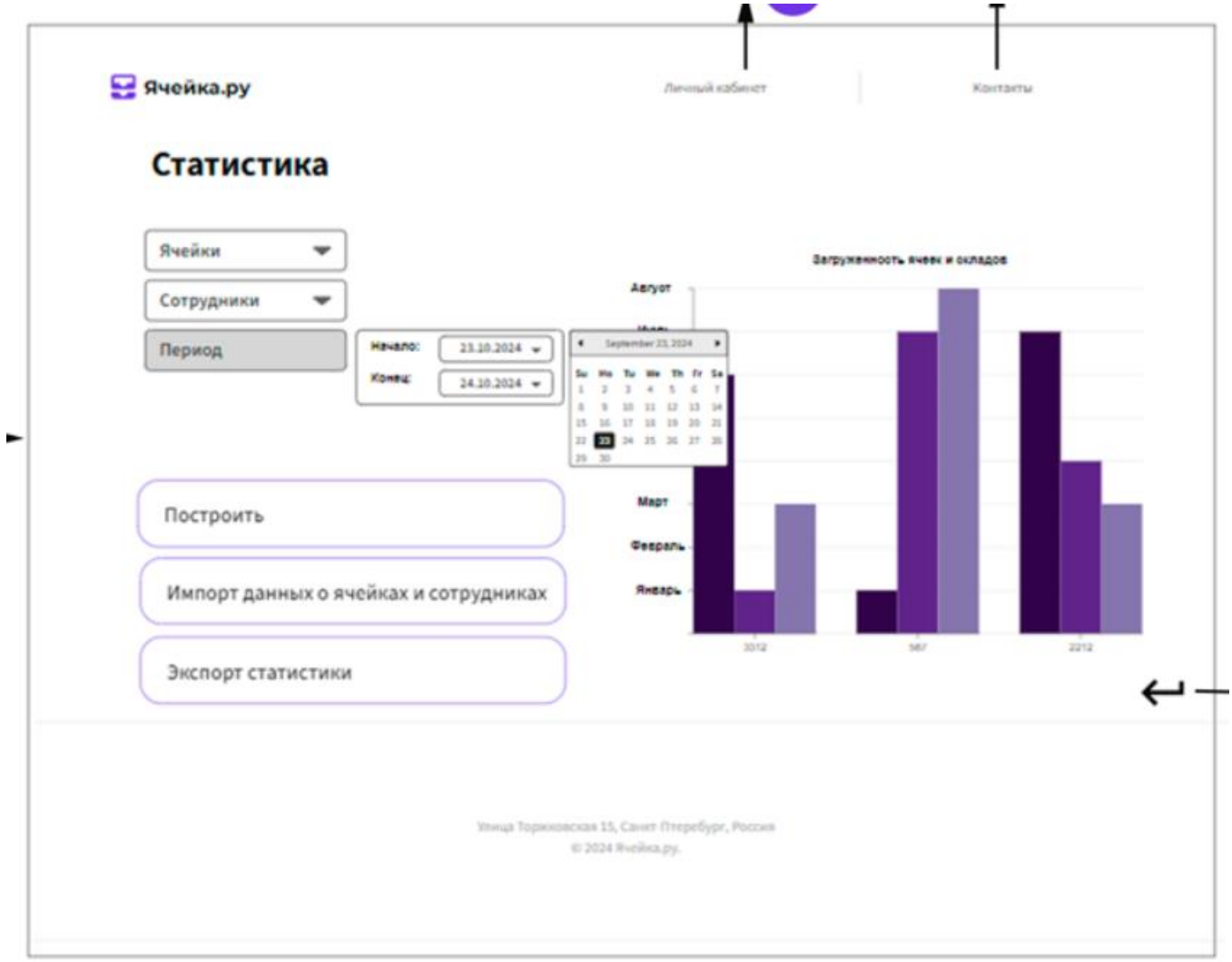

Рисунок 4. Представление данных.

5. Анализ данных (Рис. 5).

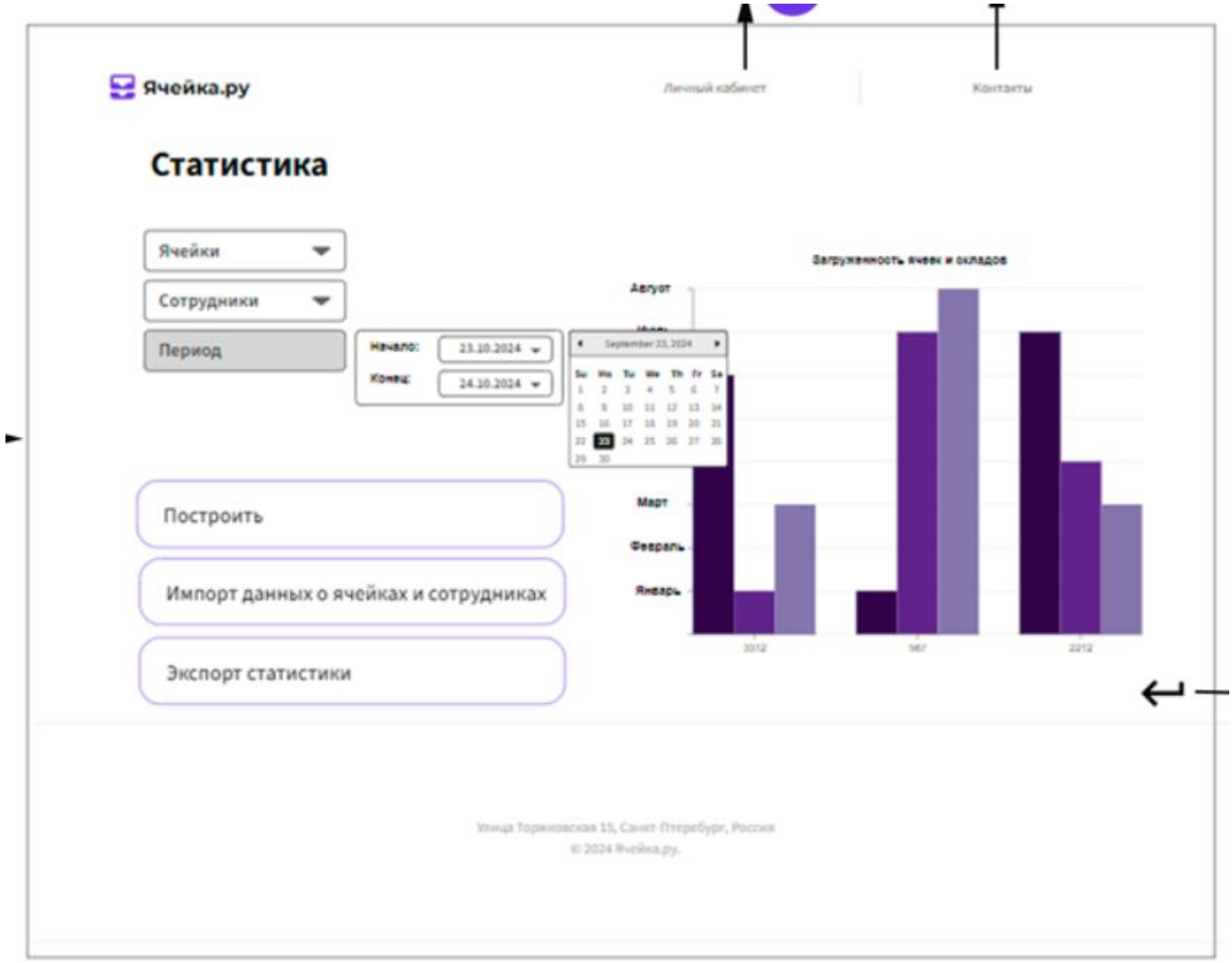

Рисунок 5. Анализ данных.

6. Экспорт данных (Рис. 6).

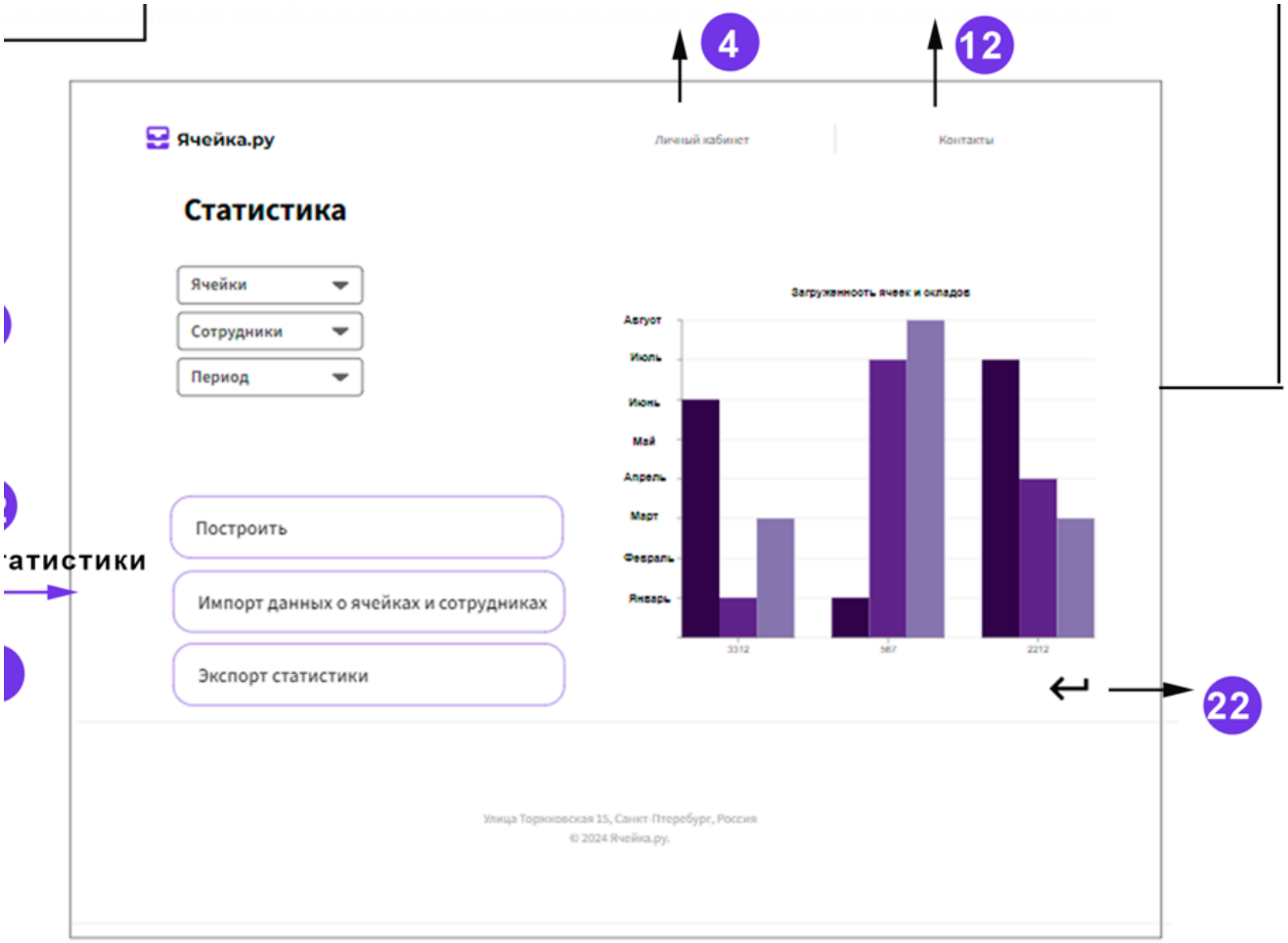

Рисунок 6. Экспорт данных.

Описание сценариев использования

В системе существуют следующие роли: пользователь, администратор, сотрудник, владелец. Владелец входит на свою страницу в системе. Он заходит на вкладку “Статистика” и оттуда имеет возможность анализировать, импортировать и экспортировать данные.

Для того, чтобы импортировать данные владелец должен:

1. Нажать на кнопку “Импорт данных о ячейках и сотрудниках”;
2. Импортировать данные.

Для того, чтобы экспортировать данные владелец должен:

1. Проставить требуемые для экспорта фильтры: по номерам ячеек, по сотрудникам или по периоду.
2. Нажать на кнопку “Экспорт статистики”;
3. Экспортировать данные.

Для того, чтобы анализировать данные владелец должен:

1. Проставить требуемые для экспорта фильтры: по номерам ячеек, по сотрудникам или по периоду.
2. Нажать на кнопку “Построить статистику”;
3. Проанализировать данные по построенной статистике.

Вывод о преобладающих операциях

Основываясь на приведенных сценариях использования, для нашего решения будут в равной мере будут преобладать операции чтения и записи, так как расширение склада с ячейками и штата сотрудников (добавление новых данных в систему) происходит так же активно, как и сгружение статистики по аренде ячеек и активности работы сотрудников.

4. Модель данных

Нереляционная модель данных

ARANGODB

Модель данных включает перечень коллекций: “Ячейки”, “Пользователи”, “События”, “Склады”.

Описание назначений коллекций, типов данных и сущностей

Исходя из тех же соображений, что и в аналогичном пункте для реляционной модели, будем считать длину строк n = 150.

Основные сущности реализованы как коллекции-документы:

- Склад

```
"WAREHOUSE" :
[
    {
        "id" : "WAREHOUSE_ID_1",
        "address" : string "SOME_ADDRESS",
        "capacity" : number "SOME_CELLS_NUMBER",
        "chiefId" : number "SOME_USER_ID",
        "cells" : array
        [
            "SOME_CELL_ID_1",
            "SOME_CELL_ID_2",
            ... ,
            "SOME_CELL_ID_N"
        ]
    },
    ... ,
    {
        "id" : "WAREHOUSE_ID_N",
        "address" : string "SOME_ADDRESS",
        "capacity" : number "SOME_CELLS_NUMBER",
        "chiefId" : number "SOME_USER_ID",
        "cells" : array
        [
            "SOME_CELL_ID_1",
            "SOME_CELL_ID_2",
            ... ,
            "SOME_CELL_ID_N"
        ]
    }
]
```

- number warehouseID (8)
- text address (n)
- number capacity (8)
- number chiefID (8)
- array<number cellID (8)> cells (100)

    - Ячейка

```
"CELL" :
[
    {
        "cellId" : "CELL_ID_1",
        "warehouseId" : "CURRENT_WAREHOUSE_ID",
        "cellNum" : number "SOME_CELL_NUM",
        "tierNum" : number "SOME_TIER_NUM",
        "isFree" : bool "TRUE_OR_FALSE",
        "endOfRent" : number "YY-MM-DD",
        "tariffPerDay" : number "SOME_PRICE",
        "size" : number "SOME_SQUARE_METRES",
        "listOfEventIds" : array
        [
            "SOME_EVENT_ID_1",
            "SOME_EVENT_ID_2",
            ... ,
            "SOME_EVENT_ID_N"
        ]
    },
    ... ,
    {
        "cellId" : "CELL_ID_N",
        "warehouseId" : "CURRENT_WAREHOUSE_ID",
        "cellNum" : number "SOME_CELL_NUM",
        "tierNum" : number "SOME_TIER_NUM",
        "isFree" : bool "TRUE_OR_FALSE",
        "endOfRent" : number "YY-MM-DD",
        "tariffPerDay" : number "SOME_PRICE",
        "size" : number "SOME_SQUARE_METRES",
        "listOfEventIds" : array
        [
            "SOME_EVENT_ID_1",
            "SOME_EVENT_ID_2",
```

- number cellID (8)
- number warehouseID (8)
- number cellNum (8)
- number tierNum (8)
- bool isFree (1)
- number endOfRent (8)
- number tariffPerDay (8)
- number size (8)
- array<number eventID (8)> listOfEventIds (3)

- Пользователь

Информация о пользователях системы, к которым относятся клиенты сервиса, работники и администрация.

```
"USER" :
[
    {
        "id" : "USER_ID_1",
        "NameSurnamePatronymic" : string "SOME_NSP",
        "role" : string "SOME_ROLE_NUM",
        "login" : string "LOGIN_1",
        "password" : string "PSWD_1",
        "birthday" : number "YY-MM-DD",
        "regDate" : number "YY-MM-DD",
        "editDate" : number "YY-MM-DD",
        "indebtedness" : number "SOME_DEBT"
    },
    ... ,
    {
        "id" : "USER_ID_N",
        "NameSurnamePatronymic" : string "SOME_NSP",
        "role" : string "SOME_ROLE_NUM",
        "login" : string "LOGIN_N",
        "password" : string "PSWD_N",
        "birthday" : number "YY-MM-DD",
        "regDate" : number "YY-MM-DD",
        "editDate" : number "YY-MM-DD",
        "indebtedness" : number "SOME_DEBT"
    }
]
```

- number userID (8)
- string NameSurnamePatronymic (n)
- string role (n)
- string login (n)
- string password (n)
- number birthday (8)
- number regDate (8)
- number editDate (8)
- number indebtedness (8)

    - Событие

    Связи между основными сущностями - события - реализованы как коллекции-дуги:

```
"EVENT" :
[
    {
        "eventId" : "EVENT_ID_1",
        "cellId" : "SOME_CELL_ID",
        "userId" : "SOME_USER_ID",
        "action" : string "SOME_ACTION",
        "dateAndTime" : number "YY-MM-DD_HH:MM:SS",
        "description" : string "SOME_DISCRIPTION"
    },
    ... ,
    {
        "eventId" : "EVENT_ID_N",
        "cellId" : "SOME_CELL_ID",
        "userId" : "SOME_USER_ID",
        "action" : string "SOME_ACTION",
        "dateAndTime" : number "YY-MM-DD_HH:MM:SS",
        "description" : string "SOME_DISCRIPTION"
    }
]
```

- number eventID (8)
- number cellID (8)
- number userID (8)
- string action (n)
- number dateAndTime (8)
- string description (n)

Графическое представление (Рис. 11):

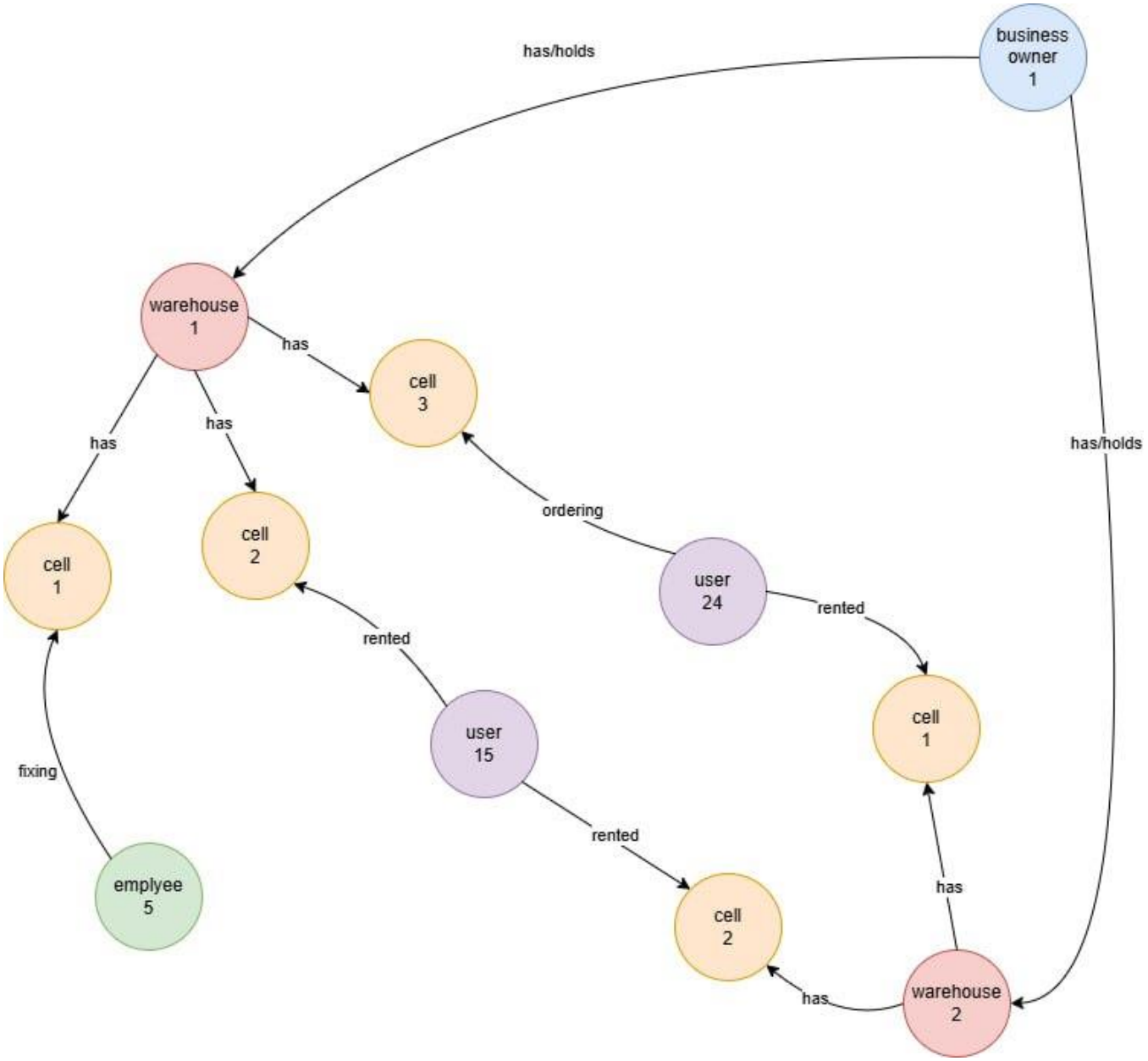

Рисунок 11. Графическое представление модели данных ArangoDB.

Оценка удельного объема информации, хранимой в моделях

Оценка объема информации.

Рассчитаем размер каждой сущности:

- Склад: 3*8 + 150 + 8*100 = 974 байт
- Ячейка: 8*7 + 1 + 8*3 = 73 байта
- Пользователь: 8*5 + 150*4 = 640 байта
- Событие: 8*4 + 150*2 = 332 байта

Для расчета формулы, определим, что на каждом складе должно быть как минимум:

- 100 ячеек

- каждую ячейку нужно хотя бы раз обслужить, забронировать и освободить => 300 событий
- 3 рабочих, 1 директор склада, 100 клиентов, 1 сис.админ => 105 пользователей

*владелец бизнеса не включен в формулу, т.к. он всегда 1.

Формула, зависящая от количества складов s:
$V(s)=(974+81*100+332*300+640*105)*s=175874*s$

Избыточность данных.

- warehouseID, cheifID и array cells в сущности Склад: 158 байт
- cellID, warehouseID и array listOfEventIds в сущности Ячейка: 41 байта
- userID в сущности Пользователь: 632 байта
- eventID, cellID и userID в сущности Событие: 308 байт

За вычетом избыточных полей, объем коллекции будет составлять:
$Vчист(s)=(158+41*100+308*300+632*105)*s=163018*s$

Отношение между фактическим объемом базы и чистым объемом составляет:
$V(s)Vчист(s)=175874*s163018*s=1.079$

Зависимость избыточности от кол-ва складов можно выразить формулой:
$V(s)-Vчист(s)=175874*s-163018*s=12856*s$

Запросы к модели

Авторизация:

```
FOR user IN USER
  FILTER user.userID == @userID && password == @password
  RETURN user
```

Регистрация:

```
INSERT { NameSurnamePatronymic : @FIO, login : @login, password : @password,
       date : DATE_NOW(), indebtedness : 0.00 }
  INTO USER
```

Аренда ячейки:

```
INSERT { eventID : @eventID, cellID : @cellID, userID : @userID,
       action : @action, dateAndTime : DATE_HOUR(DATE_NOW()),
       description : @description }
  INTO EVENT

UPDATE { cellID : @cellID }
  WITH { isFree : false, endOfRent : @endOfRent }
  IN CELL
```

Обслуживание ячейки:

```
INSERT ( "cellId" : @cellId, "userId" : @userId, "action" : 'service', "dateAndTime" : DATE_HOUR(DATE_NOW()),
  INTO EVENT
```

Поиск складов, где все ячейки заняты

```
FOR w IN WAREHOUSE
  LET num = (
    FOR c in CELL
      FILTER c.isFree == false
      WITH COUNT INTO number
      RETURN number
  )
  FILTER LENGTH(w.cells) == num
  RETURN { warehouseID : w.id }
```

Импорт данных о складах:

```
arangoimport --file "dataset.csv" --type csv --collection "Warehouse"
```

Экспорт данных о складах:

```
arangoexport --type csv --collection Event
```

Реляционная модель данных

ARANGODB

Модель данных включает перечень коллекций: “Ячейки”, “Пользователи”, “События”, “Склады”.

Описание назначений коллекций, типов данных и сущностей

Для дальнейшего расчета объема памяти, будем считать длину строк n = 150, т.к. строки фигурируют в полях с именем пользователя, логином, паролем, типом события, описанием события, адресом и названием склада, которые не требуют большой длины.

- Склад

Хранение информации о складе, на котором расположены ячейки. Поля:

int id_storage (4)
text Адрес (n)
text Название (n)
int Вместимость (4)
int id_user (4) - id ответственного за склад работника

- Ячейка

Хранение информации о характеристиках и состоянии ячейки.

int id_cell (4)
int id_storage (4)
int Номер ячейки (4)
int Номер яруса (4)
bool Свободна (1)
datetime Время окончания аренды (8)
char Размер (1)
float Тариф (8)

- Событие

Информация об изменении состояния ячейки, например: забронирована, освобождена, обслужена, выявлена неполадка.

int id_event (4)
int id_cell (4)
int id_user (4)
text Тип события (n)
text Описание (n)
datetime Время (8)

- Пользователь

  Информация о пользователях системы, к которым относятся клиенты сервиса, работники и администрация.

  int id_user (4)
  text ФИО (n)
  text Роль (n)

text Логин (n)
text Пароль (n)
datetime Дата регистрации (8)
float Задолженность (4)
date Дата рождения (3)
datetime Дата обновления профиля (8)

Графическое представление (Рис. 11):

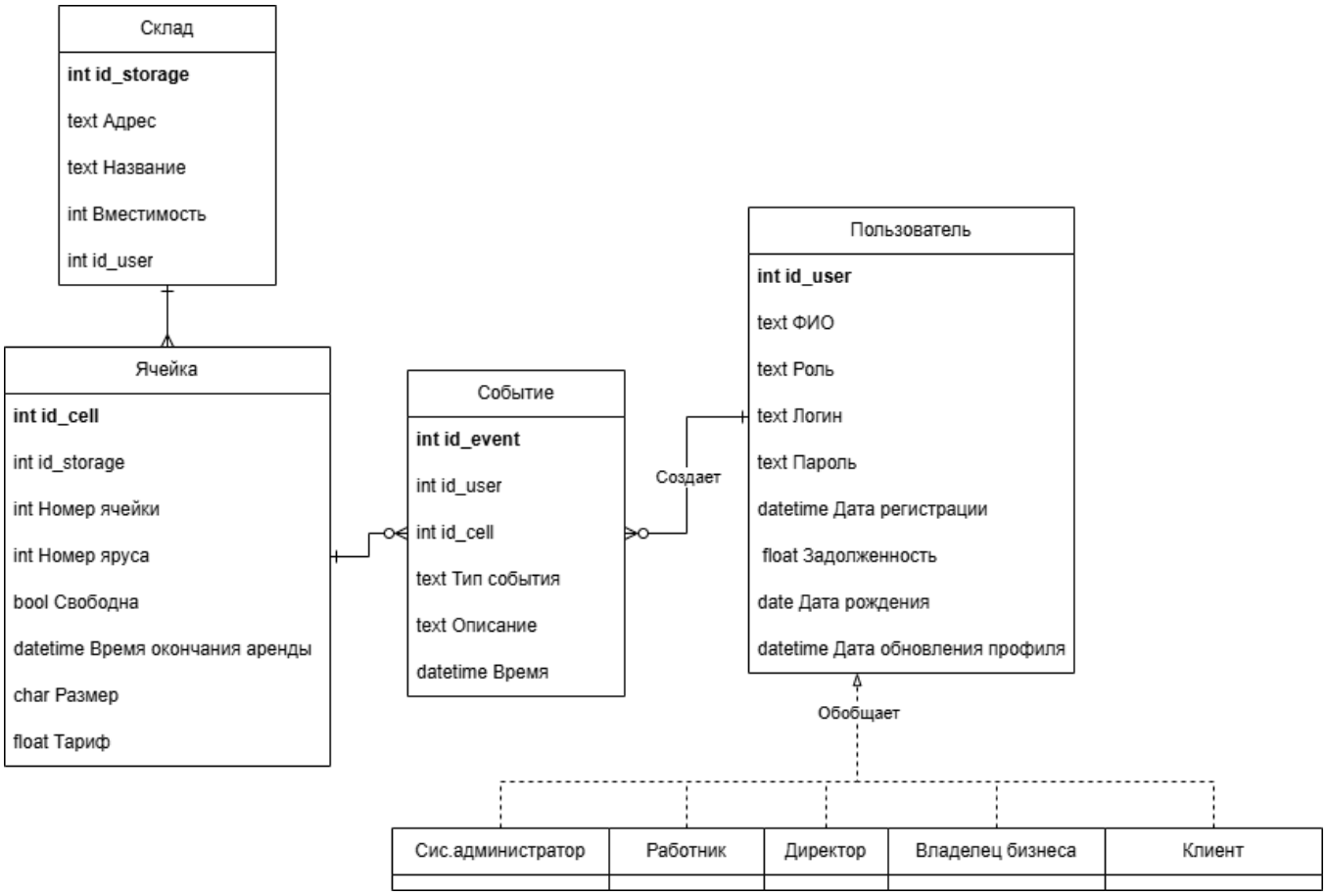

Рисунок 11. Графическое представление модели данных ArangoDB.

Оценка удельного объема информации, хранимой в моделях

Оценка объема информации.

Рассчитаем размер каждой сущности:

- Склад: 4*3 + 150*2 = 312 байт
- Ячейка: 4*3 + 2 + 2*8 = 30 байт
- Событие: 4*3 + 150*2 + 8 = 320 байт
- Пользователь: 4*2 + 150*4 + 8*2 + 3 = 627 байт

Для расчета формулы, определим, что на каждом складе должно быть как минимум:

- 100 ячеек
- каждую ячейку нужно хотя бы раз обслужить, забронировать и освободить => 300

событий
- 3 рабочих, 1 директор склада, 100 клиентов, 1 сис.админ => 105 пользователей

*владелец бизнеса не включен в формулу, т.к. он всегда 1.

Формула, зависящая от количества складов s:
$V(s)=(312+30*100+320*300+627*105)*s=165147*s$

Избыточность данных.

Избыточные поля в модели:
- id_storage, id_user в сущности Склад
- id_storage, id_cell в сущности Ячейка
- id_event, id_cell, id_user в сущности Событие
- id_user в сущности Пользователь

За вычетом избыточных полей, объем коллекции будет составлять:
$Vчист(s)=(304+22*100+312*300+623*105)*s=162044*s$

Отношение между фактическим объемом базы и чистым объемом составляет:
$V(s)Vчист(s)=165147*s162044*s=1.019$

Зависимость избыточности от кол-ва складов можно выразить формулой:
$V(s)-Vчист(s)=165147*s-162044*s=3103*s$

Запросы к модели

Авторизация:

```
SELECT role
  FROM User
 WHERE id = @id
   AND password = @password
```

Регистрация:

```
INSERT INTO User(FIO, login, password, created_at, arear)
     VALUES (@FIO, @login, @password, NOW(), 0.00)
```

Аренда ячейки:

```
INSERT INTO Event(id_cell, id_user, type_action, descript, created_at)
     VALUES (@id_cell, @id_user, "reserve", null, NOW())

UPDATE Cell
   SET is_free = false, date_free = @date_free
 WHERE id = @id_cell;
```

Обслуживание ячейки:

```
INSERT INTO Event(id_cell, id_user, type_action, descript, created_at)
     VALUES (@id_cell, @id_user, 'service', null, NOW())
```

Поиск складов, где все ячейки заняты

```
SELECT Storage.id_storage
FROM Storage
JOIN Cell ON Storage.id_storage = Cell.id_storage
GROUP BY Storage.id_storage
HAVING COUNT(c.id_cell) = COUNT(CASE WHEN c.isfree = false THEN 1 END);
```

Импорт данных о складах:

```
COPY Storage FROM '/dataset.scv' DELIMITER ',' CSV HEADER;
```

Экспорт данных о складах:

```
COPY Event TO '/export.csv' DELIMITER ',' CSV HEADER;
```

Сравнение моделей

Реляционная модель имеет меньший удельный объем информации, чем нереляционная. Это связано с тем, что в нереляционной модели данных типы данных занимают больше места в целом, также некоторые сущности хранят массивы ID других сущностей.

Удельный объем;

Entity\DB	SQL	NoSQL
Warehouse	312	974
Cell	30	81
Event	320	332
User	627	640

Запросы по отдельным юзкейсам:

Запросы\DB	SQL(количество)	NoSQL(количество)
Авторизация	1	1
Регистрация	1	1
Аренда ячейки	2	2

Количество задействованных коллекций:

Запросы\DB	SQL(количество)	NoSQL(количество)
Авторизация	1	1
Регистрация	1	1
Аренда ячейки	2	2

Вывод

На основе приведённого выше анализа и оценки, можно заключить следующее для данного проекта:

По удельному объёму нереляционная модель превосходит реляционную
Модели имеют одинаковые показатели по количеству задействованных юзкейсов и коллекций.
Для реализуемой системы логичнее выбрать SQL систему, т.к. она имеет меньший удельный вес, при прочих равных условиях.

Разработанное приложение

Краткое описание

Back-end представляет из себя C#-приложение.

Front-end – это web-приложение, которое использует API back-end приложения и отображает данные удобным образом для пользователя.

Использованные технологии

БД: ArangoDB

Back-end: C#

Front-end: HTML, CSS, JavaScript, React.

Вывод

Результаты

В ходе работы был разработан сервис управления сетью коммерческих автоматизированных складов, который позволяет пользователям удобно и быстро добавлять, удалять и редактировать их.

Будущее развитие решения

Планируется разработка нативных приложений для OS Windows и MacOS.

1. Приложения

Документация по сборке и развертыванию приложения

1. Скачать проект из репозитория (указан в ссылках на приложение)
2. Запустить React
3. Открыть приложение в браузере по адресу.

2. Используемая литература

1. Документация ArangoDB: https://docs.arangodb.com/manual/
2. Репозиторий: https://github.com/moevm/nosql2h24-store