РЕФЕРАТ

Расчётно-пояснительная записка содержит 40 с., 6 рис., 2 табл., 17 ист.

Целью работы: создание базы данных для отслеживания посетителей в сети магазинов.

Ключевые слова: базы данных, PostgreSQL, реляционная модель, OLAP, OLTP.

В данной работе проводится изучение принципов работы баз данных.

Объектом исследования является модель представления данных в посетителей в сетях магазинов.

Содержание

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

OLAP — online analytical processing — интекративная аналитическая обработка.

OLTP — online transaction processing — транзакционная система.

БД — база данных.

СУБД — система управления базами данных.

ВВЕДЕНИЕ

В настоящее время сетевые магазины предоставляют своим клиентам широкий ассортимент товаров и услуг, а также внедряют новые технологии для улучшения качества обслуживания. Одной из таких технологий является отслеживание посетителей в сети магазинов, которое позволяет собирать информацию о перемещениях клиентов внутри магазина и анализировать ее для бизнес целей.

Целью работы: создание базы данных для отслеживания посетителей в сетях магазинов.

Для достижения поставленной цели, необходимо решить следующие задачи:

1. формализовать задачу и определить необходимый функционал;
2. описать структуру объектов БД;
3. выбрать СУБД для хранения данных;
4. спроектировать и реализовать программу для обработки заявок, которая будет взаимодействовать с описанной базой данных;
5. провести исследование времени обработки операций от количества запросов в СУБД.

1 Аналитическая часть

В данном разделе проведен анализ предметной области, формализированы данные, а также проведен анализ существующих решений.

1.1 Анализ предметной области

Термин "база данных"не имеет точного определения, но стоит отметить несколько из них.

База данных [1] — это совокупность данных, хранимых в упорядоченной форме, с целью обеспечения доступа к этим данным и их использования каким-либо организационными или прикладными процессам.

База данных — это самодокументирования собрание интегрированных записей.

- Запись — это события, которые надо где-то хранить;
- интегрированных — записи, которые имеют некоторую структуру.

Также необходимо определиться с типом базы данных. Всего существует два основных применения баз данных

1. **OLAP** [2] — это метод обработки данных, который используется для анализа больших объемов данных.
2. **OLTP** [3] — это метод обработки транзакций, который используется для выполнения операция в режиме реального времени.

Из этих данных определений следует, что для поставленной задачи больше подойдет метод OLTP, так как для обработки посетителей в магазине, требуется обработка в реальном времени.

Для выполнения курсовой работы, также необходимо выбрать систему управления базами данных.

СУБД — это приложение обеспечивающее создание, хранение, обновление и поиск информации.

У систем управления базами данных существует классификация:

1.2 Классификация СУБД

1.2.1 По модели данных

1.2.1.1 Дореляционные

1. **Инвертированные списки** (файлы). БД на основе инвертированных списков представляет собой совокупность файлов, содержащих записи (таблиц). Для записей в файле определен некоторый порядок, диктуемый физической организацией данных. Для каждого файла может быть определено произвольное число других упорядочений на основании значений некоторых полей записей (инвертированных списков). Обычно для этого используются индексы. В такой модели данных отсутствуют ограничения целостности как таковые. Все ограничения на возможные экземпляры БД задаются теми программами, которые работают с БД. Одно из немногих ограничений, которое все-таки может присутствовать — это ограничение, задаваемое уникальным индексом.
2. **Иерархическая модель** данных подразумевает что элементы, организованные в структуры, объединены иерархической или древовидной связью. В таком представлении родительский элемент может иметь несколько дочерних, а дочерний — только один родительский.
3. **Сетевые** — могут быть представлены в виде графа; логика выборки зависит от физической организации данных.

1.2.1.2 Реляционные

В отличие от вышеописанных, в данной модели не существует физических отношений между сущностями. Хранение информации осуществляется в виде таблиц (отношений), состоящих из рядов и столбцов. Отношение имеет

имя, которое отличает его от имён всех других отношений.

Существует несколько типов реляционных моделей:

- **Структурный** — данные — набор отношений.
- **Целостностный** — отношения (таблицы) отвечают определенным условиям целостности.
- **Манипуляционный** — манипулирования отношениями осуществляется средствами реляционной алгебры и/или реляционного исчисления.

1.2.2 По архитектуре организации хранения данных

1. **Локальные** — все части локальной СУБД размещаются на одном компьютере.
2. **Распределенные** — части СУБД могут размещаться на 2-х и более компьютерах.

1.2.3 По способу доступа к БД

1. **Файл-серверные** — при работе с базой, данные отправляются приложению, которое с ней работает, вне зависимости от того, сколько их нужно. Все операции — на стороне клиента. Файловый сервер периодически обновляется тем же клиентом.
2. **Клиент-серверные** — вся работа на сервере, по сети передаются результаты запросов, гораздо меньше информации. Обеспечивается безопасность данных, потому что все происходит на стороне сервера.
3. **Встраиваемые** — библиотека, которая позволяет унифицированным образом хранить; большие объемы данных на локальной машине. Доступ к данным может происходить через SQL либо через особые функции СУБД. Встраиваемые СУБД быстрее обычных клиент-серверных и не требуют установки сервера, поэтому востребованы в локальном ПО,

которое имеет дело с большими объемами данных.

4. **Сервисно-ориентированные** — БД является хранилищем сообщений, промежуточных состояний, метаинформации об очередях сообщений и сервисах;
5. Прочие — пространственная, временная и пространственно-временная.

1.3 Пользователи системы

В системе присуствуют три уровня пользователей.

1. **Сотрудник магазина** — пользователь, обладающим возможностями только просматривать сущность посетителей.
2. **Охрана** — пользователь, обладающий возможностями просматривать сущности: камер, украденных товаров, полок, посетителей.
3. **Администратор** — пользователь, обладающий возможностями изменения сущностей и полей базы данных, также есть доступен просмотр всех сущностей.

На рисунке 1 представлена диаграмма использования приложения.

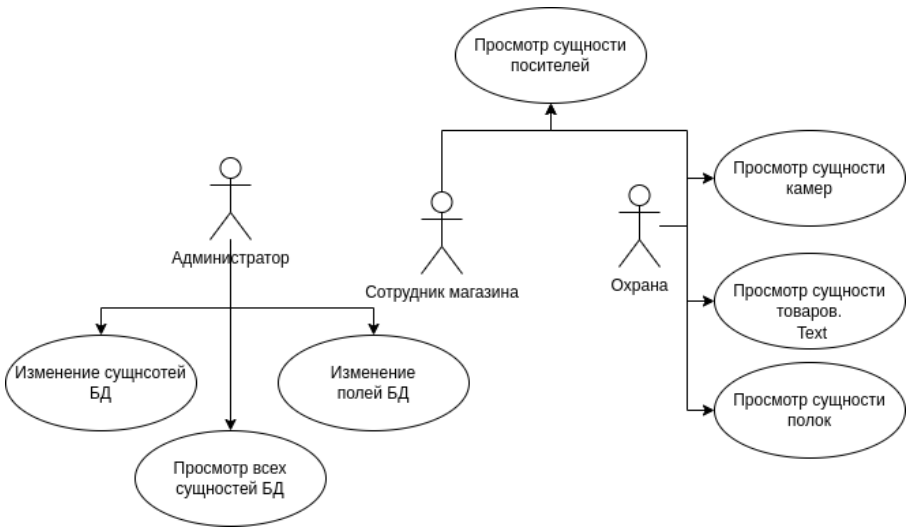


Рисунок 1 – Use-case диаграмма

1.4 Формализация данных

База данных состоит из нескольких таблиц:

1. таблица посетителей Visitor;
2. таблица камер Camera;
3. таблица полок Shelf;
4. талица товаров Product;
5. таблица сетей магазинов Chain store;

На рисунке представлена ER-диаграмма сущностей в нотации Чена.

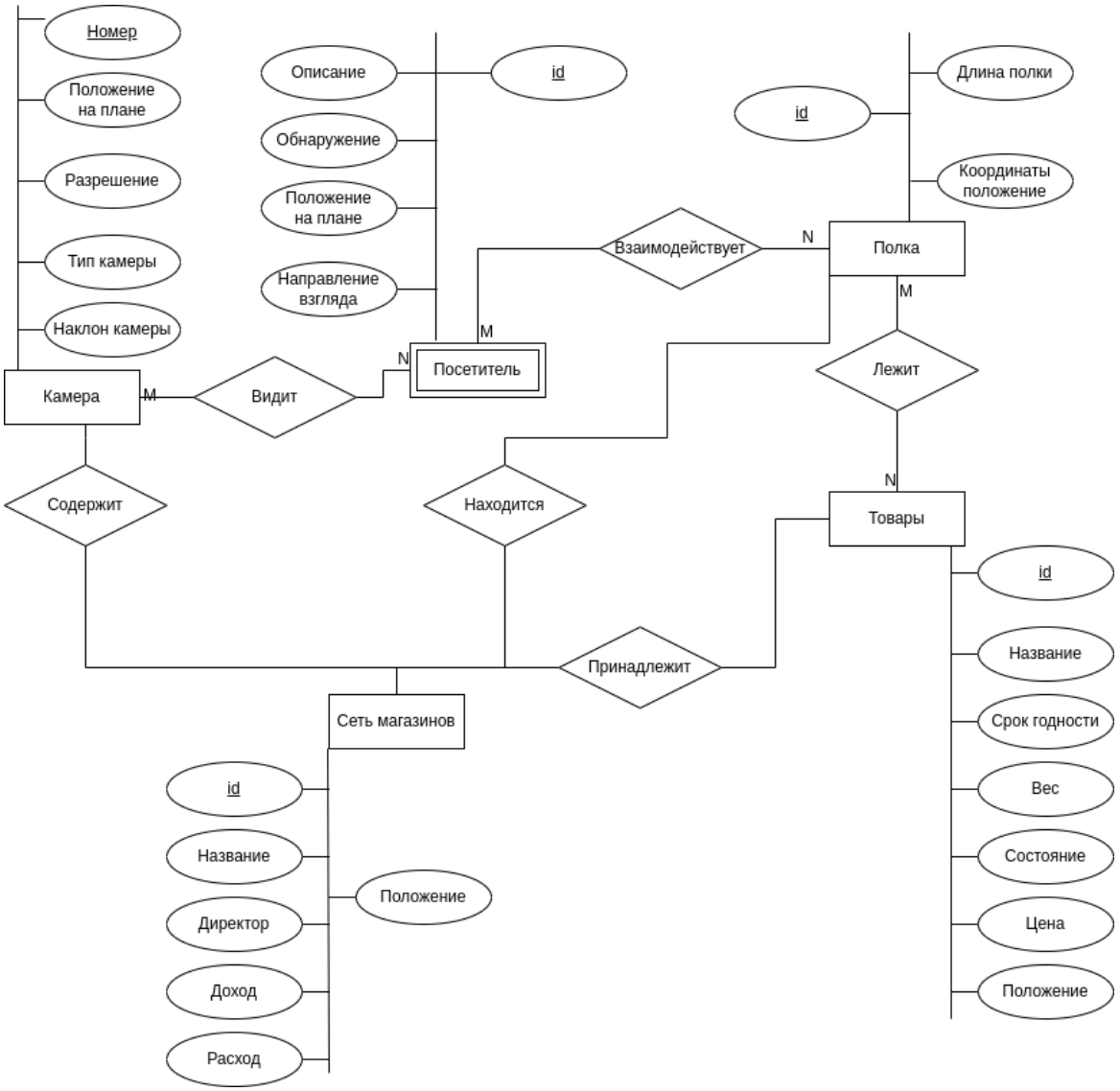


Рисунок 2 – ER-диаграмма в нотации Чена

1.5 Анализ существующих решений

Среди уже имеющихся проектов, решающих поставленную задачу, были выделены 3 аналога. Сравнение проводилось по ряду критериев, а именно наличие определения местоположения, не требующее дополнительного оборудования, определение характеристик посетителей.

В таблице 1 представлено сравнение по вышеупомянутым критериям.

Таблица 1 – Существующие решения поставленной задачи

Название проекта	**Местоположение**	**Доп. оборудование**	**Характеристики**
Антивор [4]	Определение местоположения может проводится при помощи дополнительного оборудования	Требуется большое количество дополнительного оборудования для работы системы	Не предусмотренно
Воролов [5]	Нет	Оборудование не образует систему	Не предусмотренно
Navigine [6]	Определение с помощью дополнительного оборудования	Браслет и стационарный датчик	С помощью браслета

Из таблицы можно сделать вывод, что каждая система требует большое количество дополнительного оборудования. Также не во всех присутствует определение характеристик посетителей.

Создаваемое программное обеспечение предоставляет дополнительный функционал, без использования дополнительного оборудования.

Такими особенностями являются:

1. определение местоположения на основе положения камер;

2. задание характеристик посетителей;
3. проверка взаимодействия посетителя и товара.

1.6 Выбор модели базы данных

В данном проекте будет использоваться структурная реляционная модель данных, так как в рамках проектах она обладает следующими преимуществами:

1. изложение информации осуществляется с помощью таблиц;
2. позволяет работать со структурированными данными;
3. имеет возможность произвольного доступа к записям сущностей;
4. исключает дублирование, при помощи реализации связи между отношениями посредством внешнего ключа.

Вывод

В данном разделе рассмотрены ролевые модели системы, конкретизированны хранимые данные и их связь между собой, построенны соответствующие диграммы. Также представлено анализ существующих решений. Был осуществлен выбор модели данных.

2 Конструкторская часть

В данном разделе рассматриваются этапы проектирования базы данных, спроектированы триггер и функция.

2.1 Формализация сущностей системы

На основе выделенных ранее сущностей 1 спроектированы таблицы базы данных.

2.1.1 Таблица Visitor

Содержит информацию о посетителях магазинов, которые включает следующие поля:

1. id — индетификатор пользователя, который является первичным ключом;
2. description — описание пользователя, вещественный тип;
3. location — положение пользователя на плане магазина, символьный тип который состои из двух вещественных чисел;
4. view — вектор взгляда, символьный тип, состоящий из шести вещественных чисел, которые задают вектор взгляда;
5. detection — описание пользователя, символьный тип, состоящий из 4 вещественных чисел, которые задают прямоугольник обнаружения посетителя.

2.1.2 Таблица Camera

Содержит информацию о камерах в магазине, которые включает следующая поля:

1. id — номер камеры в магазине, первичный ключ;

2. location — положение пользователя на плане магазина, символьный тип который состои из трех вещественных чисел;
3. resolution — разрешение камеры, символьный тип состоящий из двух вещественных чисел;
4. rotation — наклон камеры, символьный тип. Является матрицей состоящих из 9 вещественных чисел;
5. type — тип камеры, символьный тип.

2.1.3 Таблица Shelf

Содержит информацию о полках в магазине, которые включает следующая поля:

1. id — номер полки в магазине, первичный ключ;
2. location — положение пользователя на плане магазина, символьный тип который состои из трех вещественных чисел;
3. length — длина полки, вещественный тип.

2.1.4 Таблица Product

Содержит информацию о товарах в магазине, которые включает следующая поля:

1. id — номер товара в магазине, первичный ключ;
2. location — положение товара на плане магазина, символьный тип который состои из трех вещественных чисел;
3. name — имя товара, символьный тип;
4. dataEnd — срок годности, тип дата;
5. weight — вес товара, вещественный тип;
6. status — статус товара, целочисленный тип;
7. price — цена товара, вещественный тип.

2.1.5 Таблица ChainStore

Содержит информацию о сетях магазинов, которые включает следующая поля:

1. id — номер магазина, первичный ключ;
2. location — положение магазина в городе, символьный тип который состои из двух вещественных чисел (широта и долгота);
3. name — имя магазина, символьный тип;
4. nameDir — имя директора магазина, символьный тип;
5. income — доход магазина, вещественный тип;
6. consumption — расходы магазина, вещественный тип.

2.2 Ролевая модель

Для обеспечения работы пользователей с системой управления базами данных, выделена следующая ролевая модель.

2.2.1 Сотрудник Employee

1. SELECT — над таблицей Visitor.

2.2.2 Охрана Security

1. SELECT — над таблицей Visitor;
2. SELECT — над таблицами Camera, Product, Shelf.

2.2.3 Администратор Administrator

1. все права над таблицами Visitor;
2. все права над таблицами Camera, Product, Shelf;

3. все права над таблицами ChainStore.

2.3 Разработка триггера и функции

В СУБД предусмотрена функция, которая проверяет местоположение пользователя относительно выхода. Данная функция возвращает статус посетителя (находится внутри или вне магазина).

Также системе представлен ALTER триггер, который оповещает систему о том, что посетитель вышел из магазина.

Вывод

В данном разделе были формализированны сущности системы, представлен рисунок диаграммы сущности системы, выделены ролевые модели, спроектированы триггер и функция.

3 Технологическая часть

В данном разделе выбирается СУБД, средства реализации приложения, описаны создание базы данных, триггера, функции и ролей, а также спроектирован пользовательский интерфейс.

3.1 Выбор СУБД

Существует множество различных СУБД, работающих на основе реляционной модели, каждая из которых имеет свои сильные и слабые стороны. Среди самых распространенных [7] выделяют MySQL [8], PostgreSQL [9] и SQLite [10]. Рассмотрим особенности каждой из них.

1. MySQL. Среди достоинств данной СУБД можно выделить высокую безопасность и масштабируемость, поддержку большей части функционала SQL. Однако, несмотря на перечисленные положительные аспекты, MySQL не сопровождается бесплатной технической поддержкой.
2. PostgreSQL. В рамках использования этой СУБД имеется возможность помимо встроенного SQL использовать различные дополнения, отличается поддержкой форматов csv и json, но оперирует большим объемом ресурсов.
3. SQLite. Очевидными достоинствами является компактность базы данных, которая состоит из одного файла, и переносимость. Но данная СУБД совершенно не подходит для больших БД, а также не поддерживает управление пользователями.

При реализации проекта использован PostgreSQL, поскольку эта СУБД обладает достаточным набором инструментов для поставленной задачи.

3.2 Требования к программе

Программное обеспечение должно удовлетворять следующим требованиям:

1. возможно подключение к бд;
2. программа позволяет определять время запросов;
3. возможно создание программного интерфейса для работы с бд.

3.3 Средства реализации

Для реализации ПО был выбран язык программирования Python[11].

В данном языке есть все требующиеся инструменты для данной курсовой работы.

В качестве среды разработки была выбрана среда VS Code[12], запуск происходил через команду python back.py.

3.4 Создание базы данных

В соответствии с выбранной СУБД и спроектированной базой данных было осуществлено создание БД и ее сущностей представлено в листинге 1.

Листинг 1 – Создание БД

```
class Visitor(BASE):
    __tablename__ = 'Visitor'
    id = Column(Integer, Identity(always=True), primary_key=True)
    description = Column(Text)
    location = Column(Text)
    view = Column(Text)
    detection = Column(Text)

class Camera(BASE):
    __tablename__ = 'Camera'
    id = Column(Integer, Identity(always=True), primary_key=True)
    location = Column(Text)
```

```
13      resolution = Column(Text)
14      rotation = Column(Text)
15      cam_type = Column(Text)
16  class CameraVisitor(BASE):
17      __tablename__ = 'CameraVisitor'
18      id_vis = Column(Integer, ForeignKey('Visitor.id'), primary_key=True)
19      id_cam = Column(Integer, ForeignKey('Camera.id'), primary_key=True)
20
21  class Shelf(BASE):
22      __tablename__ = 'Shelf'
23      id = Column(Integer, Identity(always=True), primary_key=True)
24      location = Column(Text)
25      length = Column(Float)
26  class ShelfVisitor(BASE):
27      __tablename__ = 'ShelfVisitor'
28      id_shelf = Column(Integer, ForeignKey('Shelf.id'), primary_key=True)
29      id_cam = Column(Integer, ForeignKey('Camera.id'), primary_key=True)
30  class Product(BASE):
31      __tablename__ = 'Product'
32      id = Column(Integer, Identity(always=True), primary_key=True)
33      name = Column(Text)
34      location = Column(Text)
35      dataEnd = Column(Date)
36      weight = Column(Float)
37      status = Column(Integer)
38      price = Column(Integer)
39
40  class ShelfProduct(BASE):
41      __tablename__ = 'ShelfProduct'
42      id_shelf = Column(Integer, ForeignKey('Shelf.id'), primary_key=True)
43      id_product = Column(Integer, ForeignKey('Product.id'), primary_key=True)
44
45  class ChainStore(BASE):
46      __tablename__ = 'ChainStore'
47      id = Column(Integer, Identity(always=True), primary_key=True)
48      name = Column(Text)
49      location = Column(Text)
50      nameDir = Column(Text)
51      income = Column(Float)
52      consumption = Column(Float)
```

3.5 Создание триггера

В предыдущем разделе был спроектирован триггер AFTER. Код его создания представлен в листинге 2

Листинг 2 – Реализация триггера AFTER

```
CREATE TRIGGER check_location_trigger
AFTER UPDATE ON "Visitor"
FOR EACH ROW
EXECUTE FUNCTION check_location();
```

Для этого триггера была написана соответствующая функция. Код функции представлен в листинге 3.

Листинг 3 – Реализация функции

```
CREATE FUNCTION check_location() RETURNS TRIGGER AS $$
BEGIN
    IF NEW.location = '0␣0' THEN
    RAISE NOTICE 'New␣visitor␣added␣with␣location:␣%', NEW.location;
    Return NEW;
    ELSE
    RETURN NULL;
    END IF;
END;
$$ LANGUAGE plpgsql;
```

3.6 Создание ролей и выделение им прав

В конструкторском разделе была разработана ролевая модель, в которой выделены следующие роли:

1. employee — сотрудник;
2. security — охрана;
3. administrator — администратор.

Соответствующий этой ролевой модели сценарий создания ролей и выделения им прав представлен на листинге 4.

Листинг 4 – Создание ролей и выделение им прав

```
CREATE ROLE employee LOGIN PASSWORD 'postgres';
GRANT SELECT ON TABLE "Visitor" TO employee;

CREATE ROLE security LOGIN PASSWORD 'postgres';
GRANT SELECT ON TABLE "Visitor", "Product", "Shelf" TO security;

CREATE ROLE administrator LOGIN PASSWORD 'postgres';
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO administrator;
```

3.7 Интерфейс программы

Для работы с БД был разработан интерфейс взаимодействия в виде API [13]. Для реализации API была использована библиотека fastapi [14].

В программного интерфейсе реализованы методы: добавления, чтения, обновления и удаления для каждой таблицы. Методы представлены на рисунке 3.

Для использования чтения необходимо указать количество элементов, которые будут выведены. Чтобы добавить элемент в таблицу необходимо указать, необходимые поля в таблице. Для обновления и удаления элемента таблицы, необходимо указать его номер.

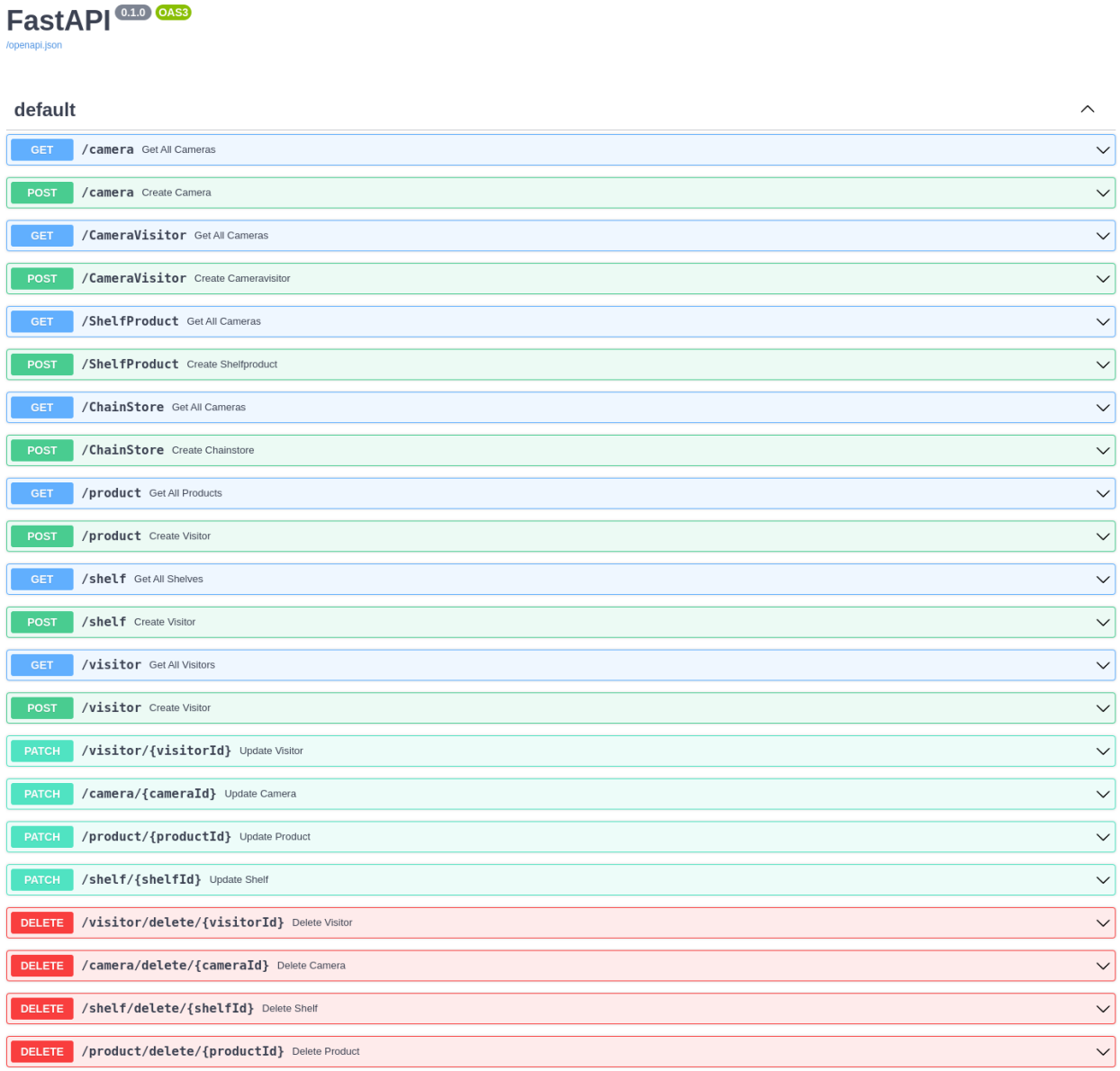


Рисунок 3 – Программный интерфейс

Вывод

В данном разделе выбраны СУБД и средств реализации, описано создание БД, триггера, ролей с выделением прав. Также представлен пользовательский интерфейс.

4 Исследовательская часть

В данном разделе произведено постановка задачи исследования и представлены результаты данного исследования.

4.1 Технические характеристики

Тестирование выполнялось на устройстве со следующими техническими характеристиками:

- операционная система Pop!_OS 22.04 LTS [15] Linux [16];
- оперативная память 16 Гбайт;
- процессор AMD® Ryzen 7 2700 eight-core processor × 16 [17].

4.2 Демонстрация работы

На рисунке 4 демонстрируется запрос, который показывает каких посетителей видит камера.

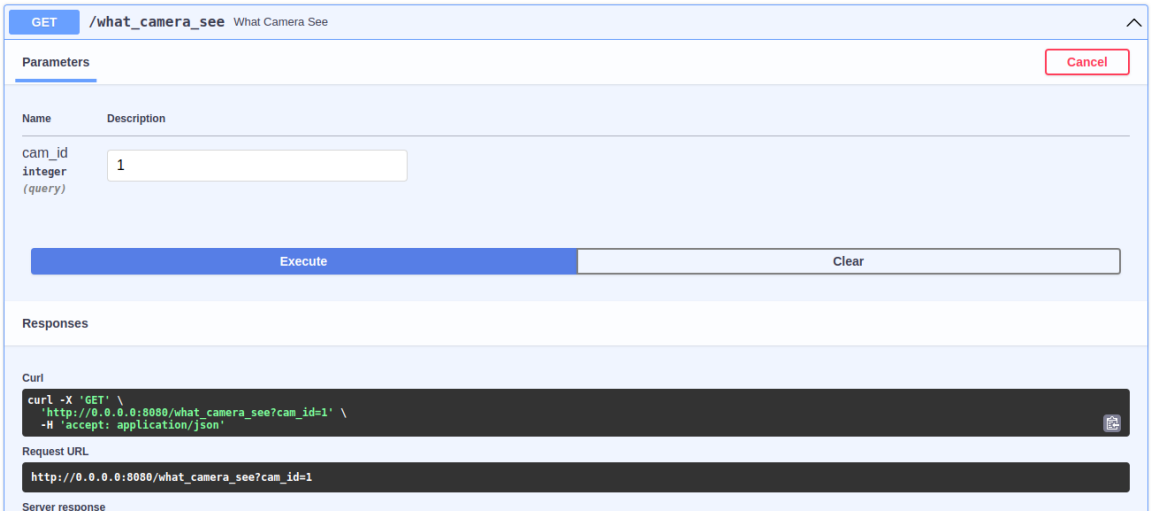


Рисунок 4 – Демонстрация запроса

На рисунке 5 демонстрируется результат запроса, из которого видно, что камера видит трех посетителей.

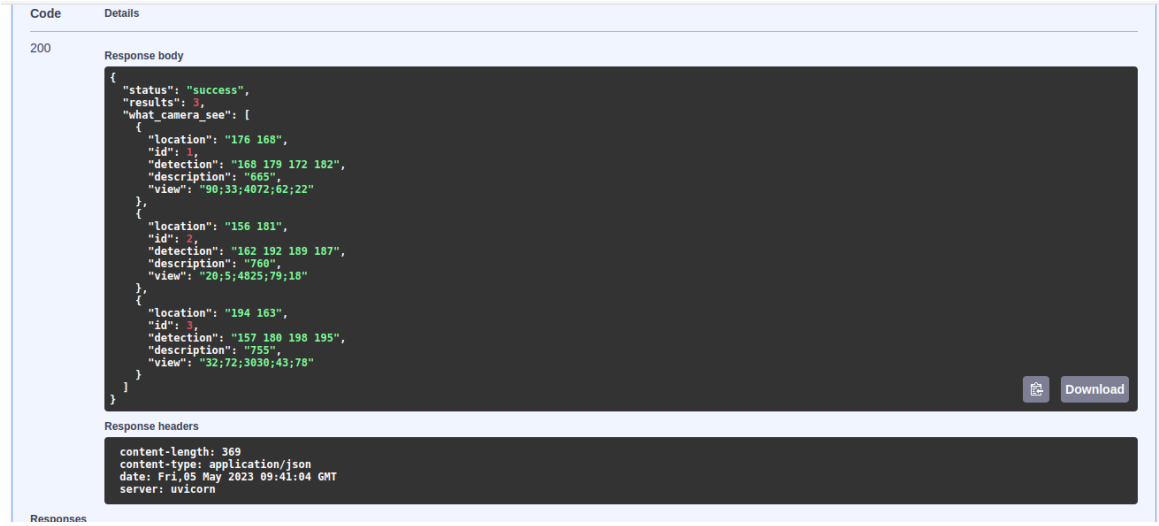


Рисунок 5 – Демонстрация результата запроса

4.3 Постановка исследования

Целью является исследование времени обработки операции от количества запросов в СУБД.

4.4 Результаты исследования

В таблице 2 продемонстрировано пользовательское время программы при разном количестве полей таблиц.

Таблица 2 – Время работы программы при разном количестве полей

Количетво полей	Время в с.
100	0.006
1000	0.009
10000	0.097
100000	1.170
1000000	11.118

На рисунке 6 представлено время запроса к БД.

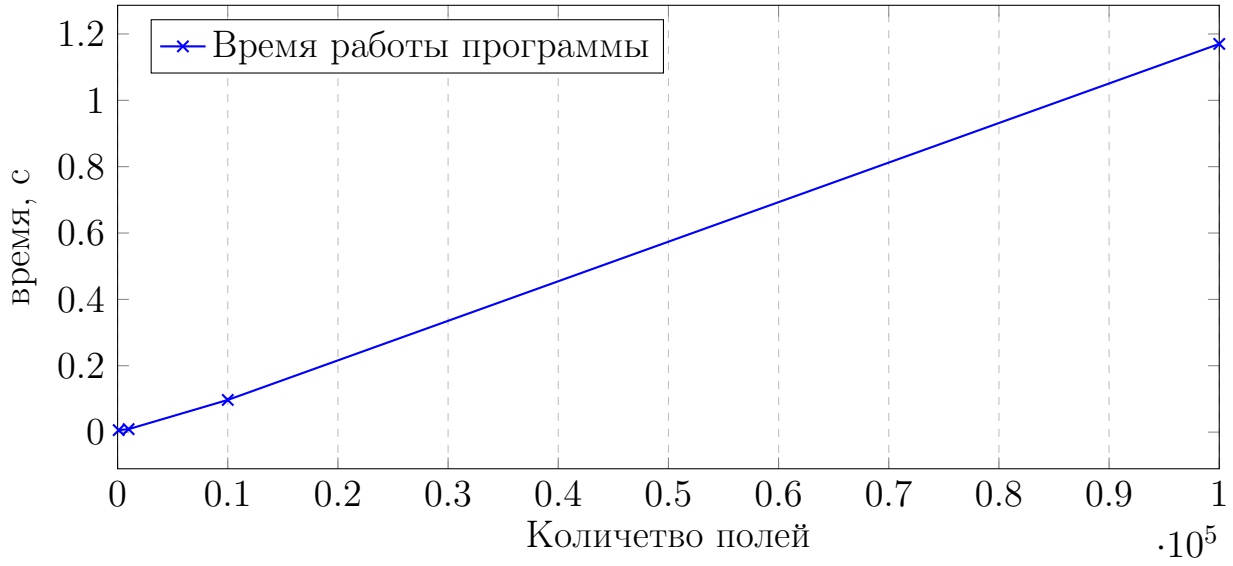


Рисунок 6 – Результаты время запроса к БД

Из результатов исследования можно сделать вывод, что зависимость времени запроса от количества полей линейно.

Вывод

В данном разделе постановлена задачи исследования и представлены результаты данного исследования.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы были выполнены следующее задачи:

1) формализована задача и определен необходимый функционал;
2) описана структуру объектов БД;
3) выбрана СУБД для хранения данных;
4) спроектирована и реализована программу для обработки заявок, которая будет взаимодействовать с описанной базой данных;
5) проведено исследование времени обработки операций от количества запросов в СУБД.

Цель курсового проекта достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Роб, П., Коронелл, К. Базы данных: концепции, технологии, применение. - БХВ-Петербург: Вильямс, 2004. - 15 с.

[2] С. Я. Архипенков, Д. В. Голубев, О. Б. Максименко К. Хранилища данных - Москва: Диалог-МИФИ, 2002. - 105 с.

[3] С. Я. Архипенков, Д. В. Голубев, О. Б. Максименко К. Хранилища данных - Москва: Диалог-МИФИ, 2002. - 106 с.

[4] Антивор [Электронный ресурс]. — Режим доступа: `https://antivor.ru/`, свободный (дата обращения: 31.03.2023).

[5] Антивор [Электронный ресурс]. — Режим доступа: `https://vorolov.ru/schetchiki-posetitelej/`, свободный (дата обращения: 31.03.2023).

[6] Navigine [Электронный ресурс]. — Режим доступа: `https://nvgn.ru/platform/tracking/`, свободный (дата обращения: 31.03.2023).

[7] LearnSQL — The Most Popular Databases for 2022 [Электронный ресурс] — Режим доступа: `https://learnsql.com/blog/most-popular-databases-2022/`, свободный – (дата обращения: 01.05.2023).

[8] MySQL [Электронный ресурс] – Режим доступа: `https://www.mysql.com/`, свободный – (дата обращения: 01.05.2023).

[9] П. Лузанов, Е. Рогов, И. Лёвшин. Postgres: первое знакомство - Москва: Диалог-МИФИ, 2002. - 90 с.

[10] SQLite [Электронный ресурс] — Режим доступа: `https://www.sqlite.org/`, свободный — (дата обращения: 01.05.2023).

[11] Python [Электронный ресурс] — Режим доступа: https://www.python.org/, свободный — (дата обращения: 01.05.2023).

[12] Vscode [Электронный ресурс] — Режим доступа: https://code.visualstudio.com/insiders/, свободный — (дата обращения: 01.05.2023).

[13] API [Электронный ресурс] — Режим доступа: https://www.ibm.com/topics/api, свободный — (дата обращения: 01.05.2023).

[14] FastAPI [Электронный ресурс] — Режим доступа: https://fastapi.tiangolo.com/, свободный — (дата обращения: 01.05.2023).

[15] PopOs [Электронный ресурс] — Режим доступа: https://pop.system76.com/, свободный — (дата обращения: 01.05.2023).

[16] Linux [Электронный ресурс] — Режим доступа: https://www.linux.org/pages/download/, свободный — (дата обращения: 01.05.2023).

[17] AMD [Электронный ресурс] — Режим доступа: https://www.amd.com/en/support, свободный — (дата обращения: 01.05.2023).

ПРИЛОЖЕНИЕ А

Создание базы данных для отслеживания посетителей в сетях магазинов

Студент: Ковель Александр Денисович ИУ7-66Б
Научный руководитель: Степанов Валерий Павлович

Москва, 2023 г.

Цели и задачи

Цель курсового проекта - создание базы данных для отслеживания посетителей в сетях магазинов.

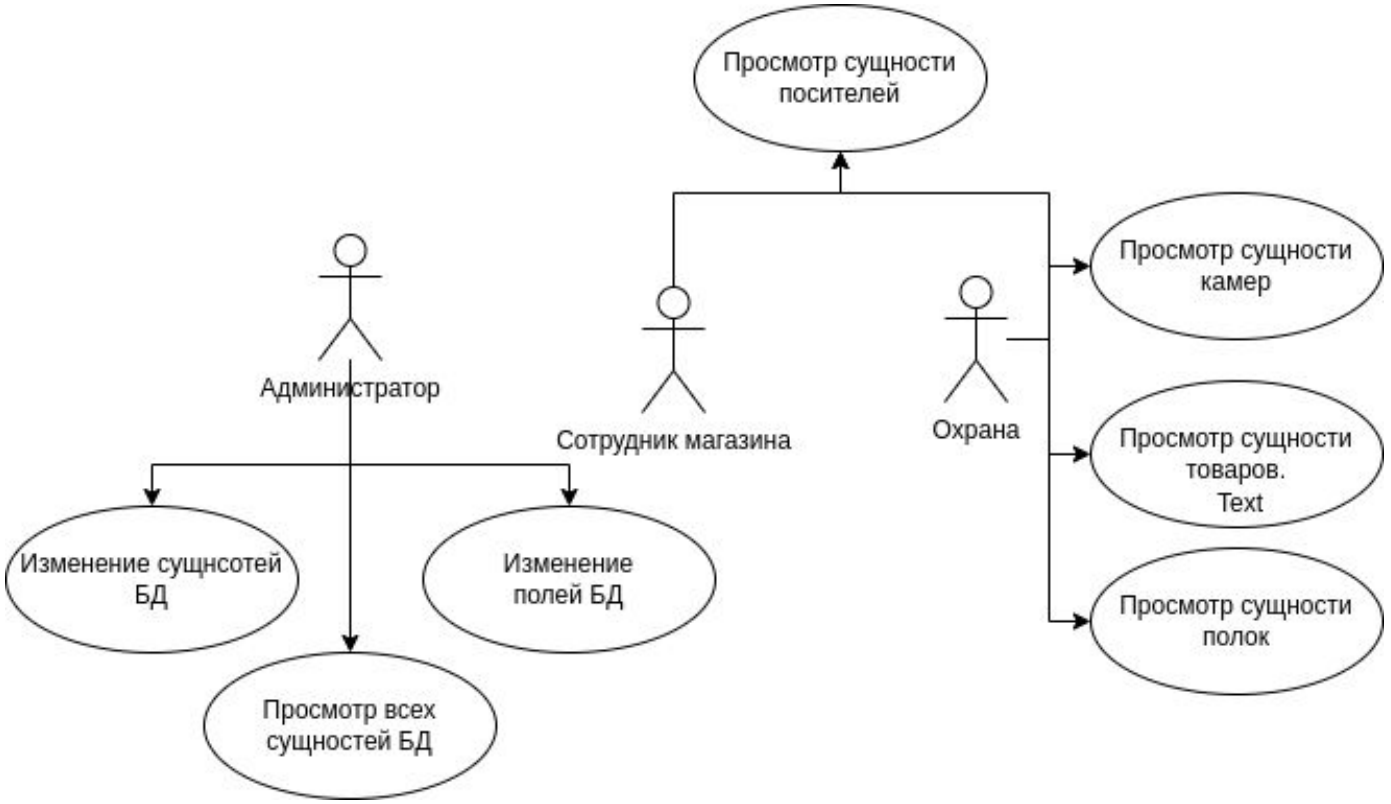
Задачи

- формализовать задачу и определить необходимый функционал;
- описать структуру объектов БД;
- выбрать СУБД для хранения данных;
- спроектировать и реализовать программу для обработки заявок, которая будет взаимодействовать с описанной базой данных;
- провести исследование времени обработки операций от количества запросов в СУБД.

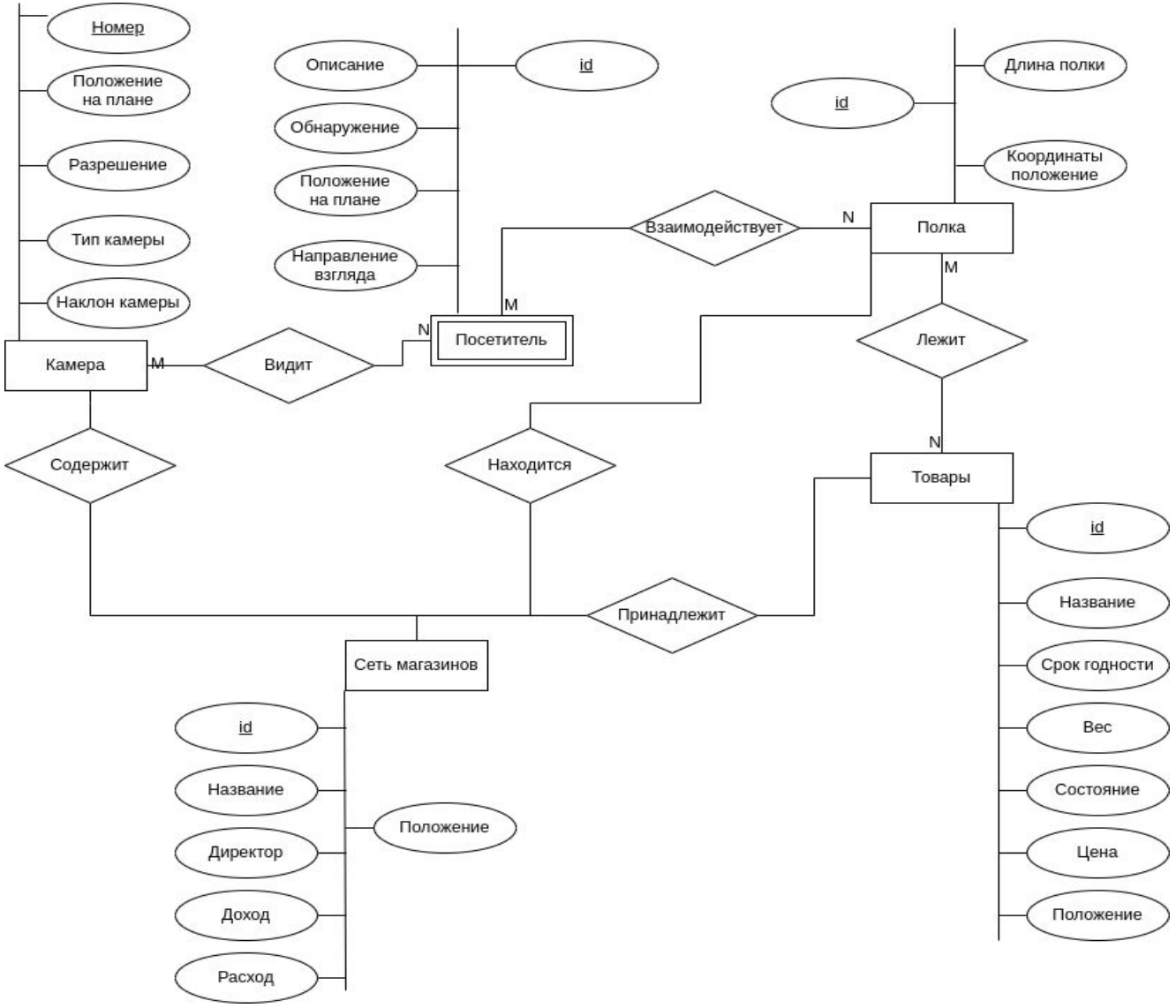
Пользователи системы

3 вида пользователей:

- Сотрудник
- Охрана
- Администратор



ER диаграмма



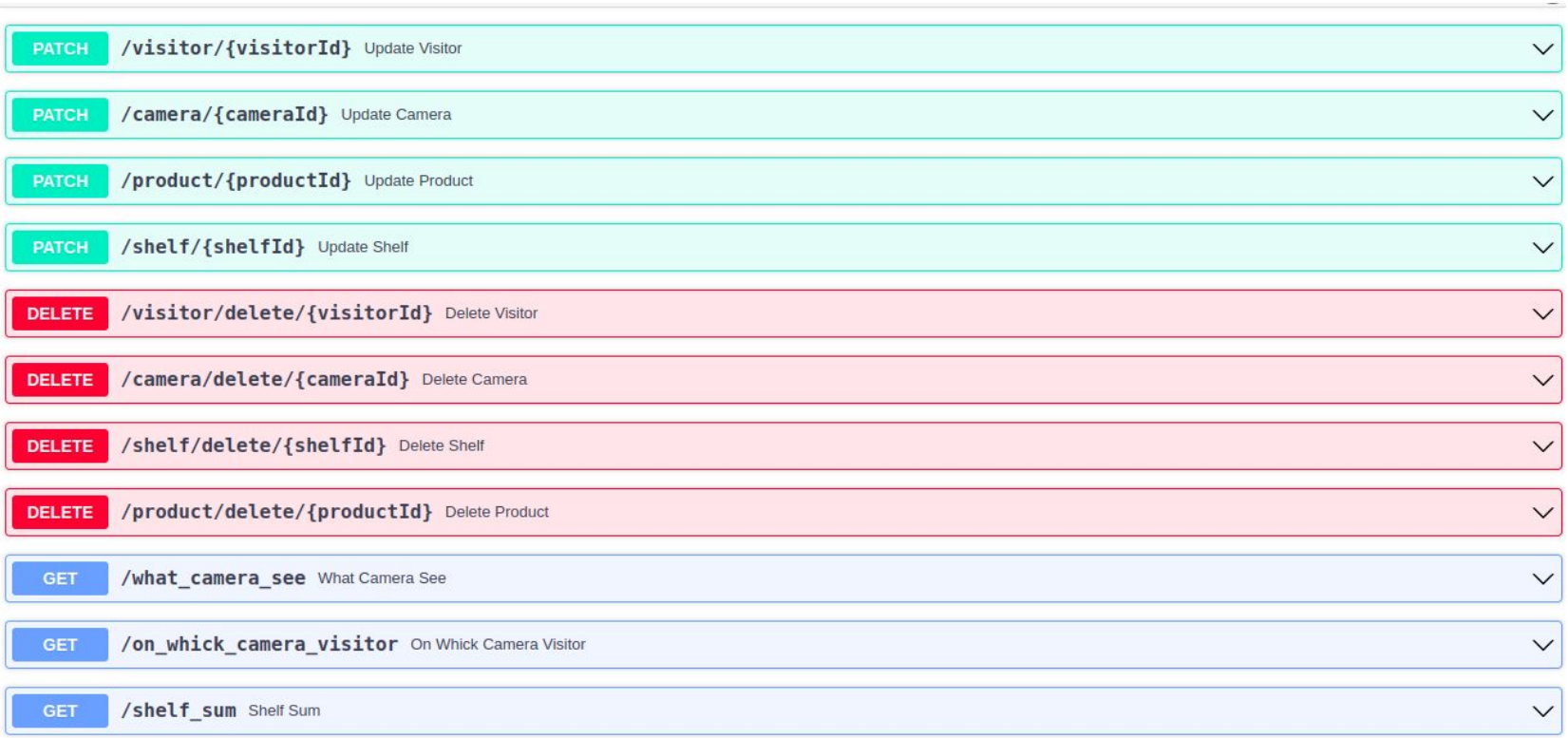
Формализация сущностей

1. Таблица Visitor (id, description, location, view, detection).
2. Таблица Camera (id, location, resolution, rotation, type).
3. Таблица Shelf (id, location, length).
4. Таблица Product (id, location, name, dataEnd, weight, status, price).
5. Таблица ChainStore (id, location, name, nameDir, income, consumption).

Выбор СУБД

1. MySql - не сопровождается бесплатно
2. PostgreSql
3. Sqlite - нет контроля пользователей

Программный интерфейс



Демонстрация работы

Найти всех посетителей которых видит камера.

Code Details

200

Response body


```
{
  "status": "success",
  "results": 3,
  "what_camera_see": [
    {
      "location": "176 168",
      "id": 1,
      "detection": "168 179 172 182",
      "description": "665",
      "view": "90;33;4072;62;22"
    },
    {
      "location": "156 181",
      "id": 2,
      "detection": "162 192 189 187",
      "description": "760",
      "view": "20;5;4825;79;18"
    },
    {
      "location": "194 163",
      "id": 3,
      "detection": "157 180 198 195",
      "description": "755",
      "view": "32;72;3030;43;78"
    }
  ]
}
```


Download

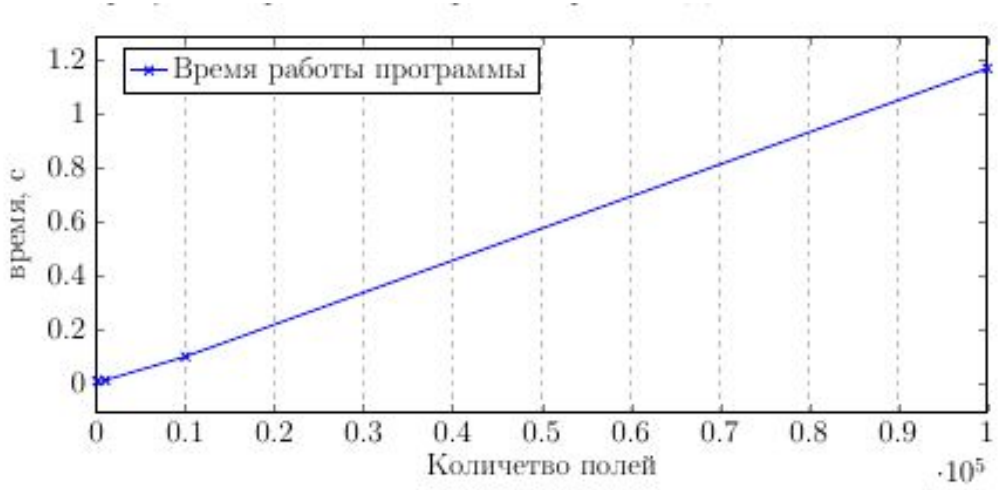
Response headers

```
content-length: 369
content-type: application/json
date: Fri,05 May 2023 09:41:04 GMT
server: uvicorn
```

Responses

Анализ работы программы

Время работы программы растет линейно.



Заключение

В рамках курсового проекта были:

- формализована задача и определен необходимый функционал;
- описана структуру объектов БД;
- выбрана СУБД для хранения данных;
- спроектирована и реализована программу для обработки заявок, которая будет взаимодействовать с описанной базой данных;
- проведено исследование времени обработки операций от количества запросов в СУБД.