Государственное бюджетное профессиональное образовательное учреждение

города Москвы

«Московский государственный колледж электромеханики и

информационных технологий»

(ГБПОУ МГКЭИТ)

Отчет ВКР

Проектирование базы данных инвентаризация

Выполнил студент группы 3ИП-11-19

Насонов Д.С.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc88816885)

[1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 4](#_Toc88816886)

[1.1 Исследование предметной области 4](#_Toc88816887)

[1.2 Анализ существующих СУБД 4](#_Toc88816888)

[2 ПРАКТИЧЕСКАЯ ЧАСТЬ 5](#_Toc88816889)

[2.1 Формирование требований 5](#_Toc88816890)

[2.2 Проектирование базы данных 5](#_Toc88816891)

[2.3 Разработка базы данных 6](#_Toc88816892)

[3 ОХРАНА ТРУДА И ТЕХНИКА БЕЗОПАСНОСТИ 8](#_Toc88816893)

[ЗАКЛЮЧЕНИЕ 9](#_Toc88816894)

[СПИСОК ЛИТЕРАТУРЫ 10](#_Toc88816895)

[ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД ОБЪЕКТОВ БАЗЫ ДАННЫХ 11](#_Toc88816896)

# ВВЕДЕНИЕ

Целью ВКР является –

Актуальность заключается в

Объектом исследования является

Предметом исследования является …

Результатом окончания ВКР будет …

Какие задачи включает в себя работа над ВКР:

(Содержимое предыдущих практик вставляем в соответствующие разделы отчёта. НЕ БЕЗДУМНО, а аккуратно и (со вкусом))

# 1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

## Исследование предметной области

**1.1.1 Анализ основных понятий.**

Согласно теме исследования, основными понятиями являются:

**Инвентаризация** – это периодическая проверка наличия и состояния материальных ценностей, принадлежащих кому-либо (учреждению, организации, предприятию и т.п.) путем подсчета, описи.   
**Сотрудник**- человек занимающийся подготовкой участка магазина.

**Ревизор**- человек являющимся главным при проверке.

**Цели и задачи инвентаризации:**

1. Выявление фактического наличия основных средств, товарно-материальных ценностей и денежных средств, ценных бумаг, а также объемов незавершенного производства в натуре.
2. Проверка реальной стоимости товарных ценностей.
3. Выявление товарно-материальных ценностей, потерявших свое первоначальное качество, не отвечающих стандартам качества, техническим условиям.

**1.1.2. Анализ основных объектов**

Таблица 1 — описание объекта товар.

|  |  |
| --- | --- |
| **Название характеристики** | **Описание** |
| Цвет | Черная, белая, цветная |
| Направление | Спорт, походы, повседневная |
| Пол | Мужская, Женская |
| Состав | Хлопок, синтетика, нейлон |

**1.1.3. Анализ основных действий с объектами:**

-Товары

1) носить

2) вешать

3) класть

4) надевать на манекены

5) складывать

**1.1.4. Анализ основных участников предметной области:**

**Сотрудник**– это человек, в обязанности которого входит проверка товаров в зоне сканирования, подготовка к инвентаризации.

**Ревизор** – это человек, в обязанности которого входит выбор участка магазина для инвентаризации, сканирование товаров в выбранном участке

## 1.2 Анализ существующих СУБД

**1) MySQL**

**1.2.1. Назначение СУБД**

MySQL— система управления реляционными базами данных с открытым исходным кодом. Где «реляционные» значит, что данные хранятся в виде таблиц. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

**1.2.2. Основные возможности**

Основные возможности СУБД MySQL является:

1. Поддерживается эффективная работа неограниченного количества пользователей, одновременно работающих с базой данных.
2. Количество строк в таблицах может достигать 50 млн.
3. Максимально быстрое выполнение команд. Существует мнение, что MySQL – и вовсе самый быстрый сервер из существующих.
4. Простая и эффективная система безопасности.

**1.2.3. Типы данных**

Основные типы данных СУБД MySQL отображены в таблице 2.

|  |  |
| --- | --- |
| Символьные типы | Char, varchar… |
| Числовые типы | Int, decimal, float, double… |
| Типы для работы с датой и временем | Date, time, datetime, year, timestamp |
| Составные типы | Enum, set |
| Бинарные типы | tinyblob, blob, mediumblom, largeblom |

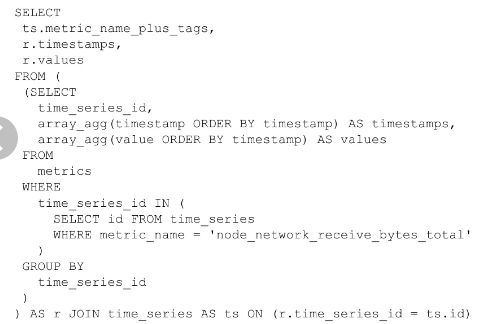
Таблица 2 — типы данных.

**1.4 Язык запросов**

Основные операторы, которые будут использоваться в дальнейших запросапредставлены в таблице 3.

|  |  |
| --- | --- |
| **Команда** | **Описание** |
| CREATE | Создает новую таблицу, представление таблицы или другой объект в БД |
| ALTER | Модифицирует существующий в БД объект, такой как таблица |
| DELETE | Удаляет существующую таблицу, представление таблицы или другой объект в БД |
| SELECT | Извлекает записи из одной или нескольких таблиц |
| INSERT | Создает записи |
| UPDATE | Модифицирует записи |
| DELETE | Удаляет записи |
| GRANT | Наделяет пользователя правами |
| REVOKE | Отменяет права пользователя |

Ниже представлен пример использования запроса (Рисунок 1).

Рисунок 1 – пример запроса.

**2. Neo4j**

**2.1 Назначение СУБД**

Neo4j является ведущей в мире графической базой данных с открытым исходным кодом, которая разработана с использованием технологии Java.

Состоит из двух элементов — узлов (вершин) и отношений (ребер). База данных графиков — это база данных, используемая для моделирования данных в форме графиков.

**2.2 Основные возможности**

Основные возможности СУБД Neo4j:

1. предоставляет результаты на основе данных в реальном времени.
2. отлично доступен для крупных корпоративных приложений реального времени с транзакционными гарантиями.
3. поддерживает полные правила ACID (атомарность, согласованность, изоляция и долговечность).

**2.3 Типы данных**

Типы данных: boolean, byte, short, int, long, float, double, char, string,

**2.4 Язык запросов**

Cypher является декларативным графовым языком запросов, который позволяет писать выразительные и эффективные запросы на получение данных из хранилища графов и их изменение.

Язык запросовCQL расшифровывается как Cypher Query Language. Как база данных Oracle имеет язык запросов SQL, Neo4j имеет CQL в качестве языка запросов. Пример запроса CREATE (<node-name>:<label-name>).

**3. Cossandra**

**3.1 Назначение СУБД**

Распределённая система управления базами данных, относящаяся к классу NoSQL-систем и рассчитанная на создание высоко масштабируемых и надёжных хранилищ огромных массивов данных, представленных в виде хэша. Отказоустойчивая СУБД: помещённые в базу данные автоматически реплицируются на несколько узлов распределённой сети или даже равномерно распределяются в нескольких дата-центрах.

**3.2 Основные возможности**

Основные возможности СУБД Cossandra является:

1. гибкая схема данных, основанная на комбинации столбцовых семейств.
2. высокая пропускная способность, особенно для операций записи (около 80-360 МБ/с на узел)
3. собственный SQL-подобный язык запросов, который позволяет выполнять простейшие запросы SELECT с выборкой по определённому условию.

**3.3 Типы данных**

Ниже представлена таблица типов данных, используемых на языке Cossandra.

|  |  |
| --- | --- |
| **Тип данных** | **Описание** |
| BytesType | любые байтовые строки (без валидации) |
| AsciiType | ASCII строка |
| UTF8Type | UTF-8 строка |
| IntegerType | число с произвольным размером |
| Int32Type | 4-байтовое число |
| LongType | 8-байтовое число |
| BooleanType | два значения: true = 1 или false = 0 |

**3.4 Язык запросов**

Cassandra Query Language (CQL). CQL - это простой интерфейс для доступа к Cassandra в качестве альтернативы традиционному языку структурированных запросов (SQL). CQL добавляет уровень абстракции, который скрывает детали реализации этой структуры и предоставляет собственный синтаксис для коллекций и других распространенных кодировок.

Ниже представлен пример запроса (рисунок 3).



Рисунок 3 – пример запроса

**1.3 Обоснование выбора СУБД**

Эффективность работы нашей системы зависит от нашего выбора СУБД. Лучшим вариантом СУБД для нашей системы будет MySQL. Из-за ее возможностей:

1. Поддерживается эффективная работа неограниченного количества пользователей, одновременно работающих с базой данных.
2. Количество строк в таблицах может достигать 50 млн.
3. Максимально быстрое выполнение команд. Существует мнение, что MySQL – и вовсе самый быстрый сервер из существующих.
4. Простая и эффективная система безопасности.

# 2 ПРАКТИЧЕСКАЯ ЧАСТЬ

## 2.1 Формирование требований

1. **Обоснование выбора СУБД**

Эффективность работы нашей системы зависит от нашего выбора СУБД. Лучшим вариантом СУБД для нашей системы будет MySQL. Из-за ее возможностей:

1. Поддерживается эффективная работа неограниченного количества пользователей, одновременно работающих с базой данных.
2. Количество строк в таблицах может достигать 50 млн.
3. Максимально быстрое выполнение команд. Существует мнение, что MySQL – и вовсе самый быстрый сервер из существующих.
4. Простая и эффективная система безопасности.
5. **Описать основные сущности и их характеристики, которые будут храниться в БД (в каком виде и как).**

Основные сущности и их описания отображены на рисунке 1.

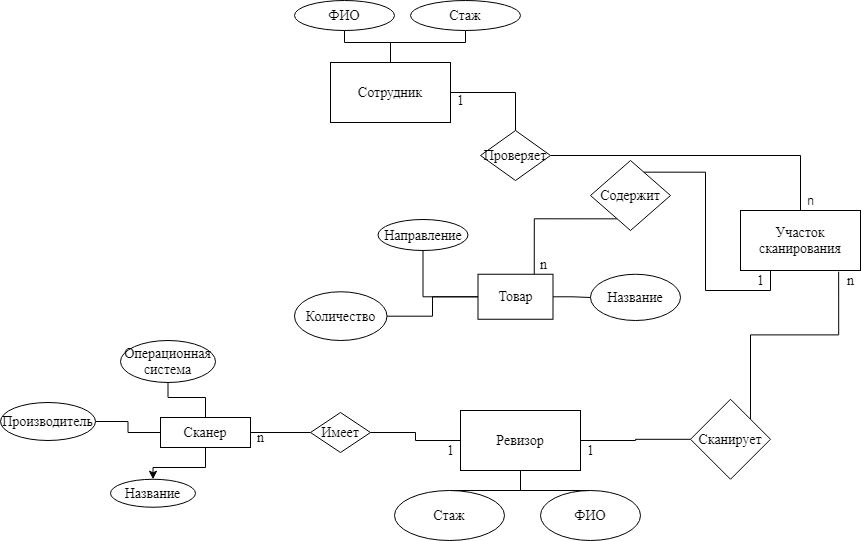
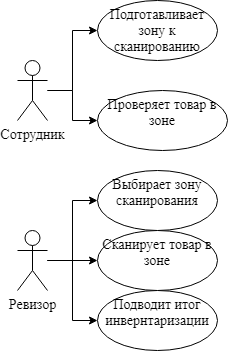
****

Рисунок 1 – диаграмма сущностей.

1. **Основные сценарии использования СУБД с помощью use-case**

Основные сценарии отображены на рисунке 2.

* 
* Рисунок 2- основные сценарии.

1. **Список API методов, предоставляемой БД для работы «из вне»**

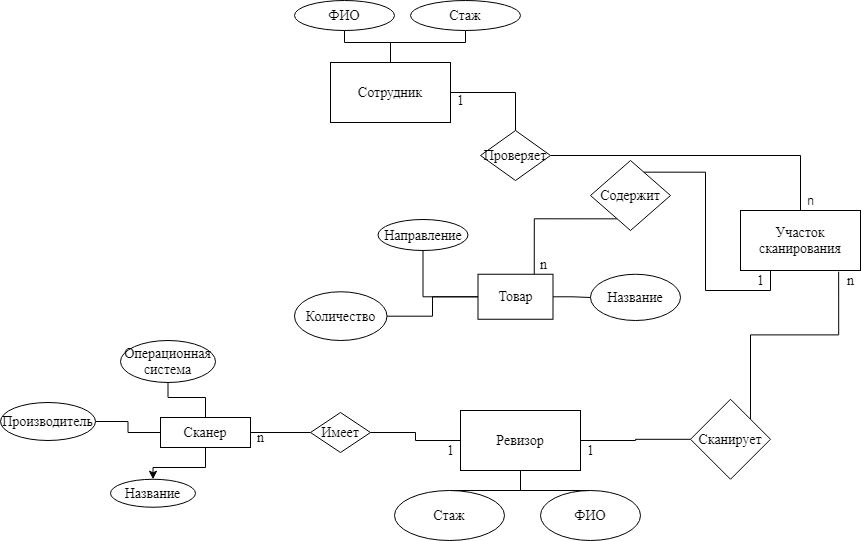
Таблица 1 – описание методов.

|  |  |
| --- | --- |
| * **Название метода** | **Описание метода** |
| Добавить сотрудника | Аргументы: ФИО(строка),стаж(временная метка) |
| Изменить данные сотрудника | Аргументы: код(число), ФИО(строка), новое ФИО(строка) стаж(временная метка), новый стаж(временная метка) |
| Изменить данные сотрудника(ФИО) | Аргументы: код(число), ФИО(строка), новое ФИО(строка) |
| Изменить данные сотрудника(стаж) | Аргументы: код(число), ФИО(строка), стаж(временная метка), новый стаж(временная метка) |
| Удалить сотрудника | Аргументы: код(число) |
| Просмотр информации о сотруднике | Аргументы: код(число), ФИО(строка), стаж(временная метка) |
| Добавить ревизора | Аргументы: ФИО(строка),стаж(временная метка) |
| Изменить данные ревизора | Аргументы: код(число), ФИО(строка), новое ФИО(строка) стаж(временная метка), новый стаж(временная метка) |
| Изменить данные ревизора(ФИО) | Аргументы: код(число), ФИО(строка), новое ФИО(строка) |
| Изменить данные ревизора(стаж) | Аргументы: код(число), ФИО(строка), стаж(временная метка), новый стаж(временная метка) |
| Удалить ревизора | Аргументы: код(число) |
| Просмотр информации о ревизоре | Аргументы: код(число), ФИО(строка), стаж(временная метка) |
| Добавить сканер | Аргументы: производитель(строка), название(строка) |
| Изменить сканер | Аргументы: код сканера(число) название(строка) производитель(строка) |
| Удалить сканер | Аргументы: код сканера(число) |
| Добавить товар | Аргументы: код товара(число), название(строка), направление(строка), количество(число) |
| Изменить товар | Аргументы: код товара(число), название(строка), новое название(строка), направление(строка), новое направление(строка), количество(число), новое количество(число) |
| Посмотреть информацию о товаре | Аргументы: код товара(число), название(строка), направление(строка), количество(число) |
| Удалить товар | Аргументы: код товара(число) |
| Изменить товар(название) | Аргументы: код товара(число), название(строка), новое название(строка) |
| Изменить товар(количество) | Аргументы: код товара(число), количество(число), новое количество(число) |
| Изменить товар(направление) | Аргументы: код товара(число), направление(строка), новое направление(строка) |
| Добавить участок сканирования | Аргументы: код участка(число) |
| Изменить участок сканирования | Аргументы: код участка(число), новый код участка(число) |
| Удалить участок сканирования | Аргументы: код участка(число) |

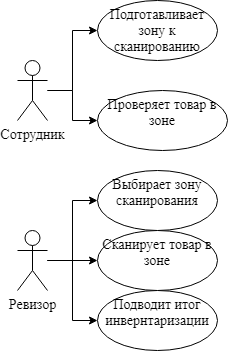
1. **Список пользователей их роли.**

Ревизор и сотрудник имеют полный доступ к просмотру базы данных Инвентаризация.

### 2.1.1 Основные сущности базы данных

****

### 2.1.2 Варианты использования



### 2.1.3 Определение API для взаимодействия с базой данных

|  |  |
| --- | --- |
| * **Название метода** | **Описание метода** |
| Добавить сотрудника | Аргументы: ФИО(строка),стаж(временная метка) |
| Изменить данные сотрудника | Аргументы: код(число), ФИО(строка), новое ФИО(строка) стаж(временная метка), новый стаж(временная метка) |
| Изменить данные сотрудника(ФИО) | Аргументы: код(число), ФИО(строка), новое ФИО(строка) |
| Изменить данные сотрудника(стаж) | Аргументы: код(число), ФИО(строка), стаж(временная метка), новый стаж(временная метка) |
| Удалить сотрудника | Аргументы: код(число) |
| Просмотр информации о сотруднике | Аргументы: код(число), ФИО(строка), стаж(временная метка) |
| Добавить ревизора | Аргументы: ФИО(строка),стаж(временная метка) |
| Изменить данные ревизора | Аргументы: код(число), ФИО(строка), новое ФИО(строка) стаж(временная метка), новый стаж(временная метка) |
| Изменить данные ревизора(ФИО) | Аргументы: код(число), ФИО(строка), новое ФИО(строка) |
| Изменить данные ревизора(стаж) | Аргументы: код(число), ФИО(строка), стаж(временная метка), новый стаж(временная метка) |
| Удалить ревизора | Аргументы: код(число) |
| Просмотр информации о ревизоре | Аргументы: код(число), ФИО(строка), стаж(временная метка) |
| Добавить сканер | Аргументы: производитель(строка), название(строка) |
| Изменить сканер | Аргументы: код сканера(число) название(строка) производитель(строка) |
| Удалить сканер | Аргументы: код сканера(число) |
| Добавить товар | Аргументы: код товара(число), название(строка), направление(строка), количество(число) |
| Изменить товар | Аргументы: код товара(число), название(строка), новое название(строка), направление(строка), новое направление(строка), количество(число), новое количество(число) |
| Посмотреть информацию о товаре | Аргументы: код товара(число), название(строка), направление(строка), количество(число) |
| Удалить товар | Аргументы: код товара(число) |
| Изменить товар(название) | Аргументы: код товара(число), название(строка), новое название(строка) |
| Изменить товар(количество) | Аргументы: код товара(число), количество(число), новое количество(число) |
| Изменить товар(направление) | Аргументы: код товара(число), направление(строка), новое направление(строка) |
| Добавить участок сканирования | Аргументы: код участка(число) |
| Изменить участок сканирования | Аргументы: код участка(число), новый код участка(число) |
| Удалить участок сканирования | Аргументы: код участка(число) |

## 2.2 Проектирование базы данных

1. **Определение таблиц**

Определения основных полей даны в таблицах 1-5.

Таблица 1- auditor

|  |  |  |
| --- | --- | --- |
| **Названия полей** | **Типы данных** | **Ограничения** |
| id\_auditor | INT | AI,NN,PK |
| first\_name | VARCHAR(30) | NN |
| last\_name | VARCHAR(30) | NN |
| middle\_name | VARCHAR(30) | NN |
| expirience | FLOAT | NN |
| id\_scanner | INT | NN,FK |

Таблица 2 – employee

|  |  |  |
| --- | --- | --- |
| **Названия полей** | **Типы данных** | **Ограничения** |
| id\_employee | INT | AI,NN,PK |
| first\_name | VARCHAR(30) | NN |
| last\_name | VARCHAR(30) | NN |
| middle\_name | VARCHAR(30) | NN |
| expirience | FLOAT | NN |

Таблица 3 – scanner

|  |  |  |
| --- | --- | --- |
| **Названия полей** | **Типы данных** | **Ограничения** |
| id\_scanner | INT | AI,NN,PK |
| name | VARCHAR(30) | NN |
| operation\_system | VARCHAR(30) | NN |
| producer | VARCHAR(30) | NN |

Таблица 4 – section

|  |  |  |
| --- | --- | --- |
| **Названия полей** | **Типы данных** | **Ограничения** |
| id\_section | INT | AI,NN,PK |
| id\_auditor | INT | NN, FK |
| id\_employee | INT | NN, FK |
| id\_item | INT | NN, FK |

Таблица 5 – Item

|  |  |  |
| --- | --- | --- |
| **Названия полей** | **Типы данных** | **Ограничения** |
| id\_item | INT | AI,NN,PK |
| name | VARCHAR(30) | NN |
| amount | INT UNSIGNED | NN |
| trend | VARCHAR(30) | NN |

### 2.2.1 Логическая схема базы данных

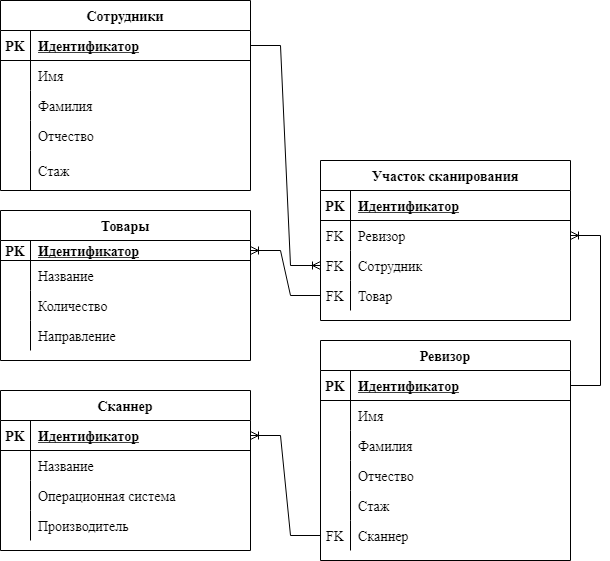


Рисунок 2. – Логическая схема базы данных

### 2.2.2 Физическая схема базы данных

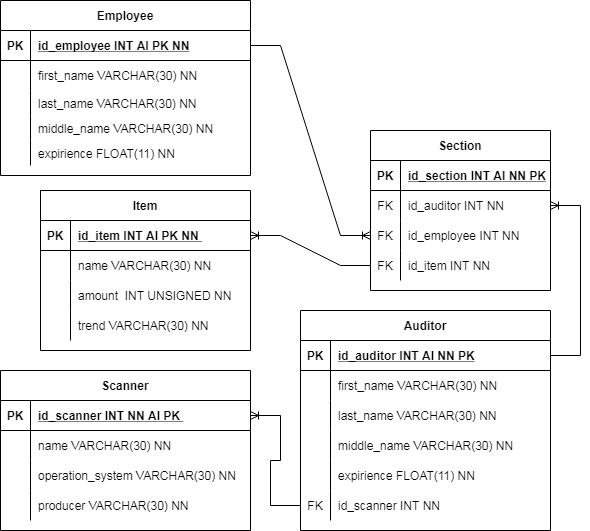


Рисунок .2 – Физическая схема базы данных

### 2.2.3 Определение типовых запросов к объектам базы данных

1. **Основные запросы.**

Основные запросы находятся в таблицах 6-10.

Таблица 6 – запросы к таблице auditor.

|  |  |
| --- | --- |
| **Запрос** | **Описание** |
| SELECT \*FROM auditor WHERE id=2 | Выводит информацию о ревизоре |
| INSERT INTO auditor(last\_name, first\_name, middle\_name, expirience, id\_scanner)  VALUES (`Овалов`,`Михаил`,`Владимирович`,22, 4) | Добавляет нового ревизора |
| UPDATE auditor set expirience=12  WHERE id=4 | Обновляет стаж выбранного ревизора |
| DELETE FROM auditor WHERE id=5 | Удаляет выбранного ревизора |

Таблица 7 – запросы к таблице employee.

|  |  |
| --- | --- |
| **Запрос** | **Описание** |
| SELECT \*FROM employee WHERE id=3 | Выводит информацию о сотруднике |
| INSERT INTO employee(last\_name, first\_name, middle\_name, expirience)  VALUES (`Типов`,`Владимир`,`Валерьевич`,11) | Добавляет нового сотрудника |
| UPDATE employee set expirience=2  WHERE id=5 | Обновляет стаж выбранного сотрудника |
| DELETE FROM employee WHERE id=5 | Удаляет выбранного сотрудника |

Таблица 8 – запросы к таблице items.

|  |  |
| --- | --- |
| **Запрос** | **Описание** |
| SELECT \* FROM items | Выводит информацию о товаре |
| INSERT INTO item(name,amount,trend)  VALUES(`Trappa`,`245`,` Походы’) | Добавляет товар в таблицу items |
| UPDATE item set amount=`110` WHERE id=3 | Обновляет запись |
| DELETE FROM item where id=2 | Удаляет данные о товаре |

Таблица 9 – запросы к таблице scanner.

|  |  |
| --- | --- |
| **Запрос** | **Описание** |
| SELECT \* FROM scanner | Выводит информацию о сканере |
| INSERT INTO scanner(name,operation\_system,producer)  VALUES(`DCP-345`,`KIS`,` Vault-tec’) | Добавляет товар в таблицу scanner |
| UPDATE scanner set operation\_system=`MAL` WHERE id=3 | Обновляет запись |
| DELETE FROM scanner where id=2 | Удаляет данные о сканнере |

Таблица 10 –запросы к таблице section.

|  |  |
| --- | --- |
| **Запрос** | **Описание** |
| SELECT \* FROM section | Выводит информацию о участке сканирования |
| INSERT INTO section(id\_auditor,id\_employee,id\_item)  VALUES(`1`,`3`,` 4’) | Добавляет участок сканирования в таблицу section |
| UPDATE section set id\_auditor=`10` WHERE id=3 | Обновляет запись |
| DELETE FROM section where id=2 | Удаляет данные о участке |

### 2.2.4 Определение процедур и функций API

1. **API функции с фактической сигнатурой.**

Основные функции отображены в таблице 11.

Таблица 11— основные функции API.

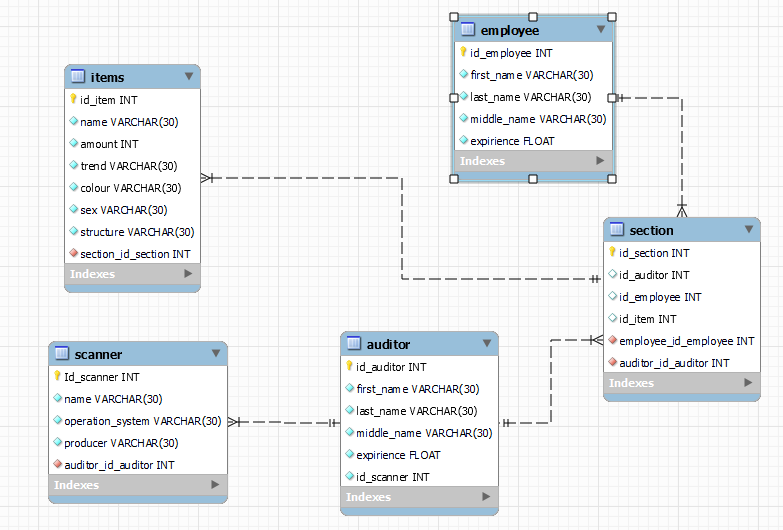
|  |  |
| --- | --- |
| **Название функции** | **Описание функции** |
| add\_auditor(first\_name varchar(30), last\_name varchar(30), middle\_name varchar(30), expirience float(11),id\_scanner) | Добавляет ревизора в базу данных |
| Change\_att(first\_name varchar(30), last\_name varchar(30), middle\_name varchar(30), experience float(11),id\_scanner) | Изменяет один из переданных параметров(Имя, фамилия, Отчество, стаж, сканнер) |
| change\_att\_auditor\_first\_name  (id int,first\_name varchar(30)) | Изменяет имя ревизора |
| change\_att\_auditor\_last\_name  (id int,last\_name varchar(30)) | Изменяет фамилию ревизора |
| сhange\_att\_auditor\_middle\_name  (id int,middle\_name varchar(30)) | Изменяет отчество ревизора |
| change\_att\_auditor\_expirience  (id int,expirience float(11)) | Изменяет стаж ревизора |
| Change\_att\_auditor\_id\_scanner (id int, id\_scanner int) | Изменяет сканнер ревизора |
| del\_ auditor(id\_auditor int(11)) | Удаляет ревизора по идентификатору |
| get\_information\_auditor(first\_name varchar(30), last\_name varchar(30), middle\_name varchar(30), expirience float, id\_scanner id) | Возвращает данные о ревизоре. |
| add\_employee(first\_name varchar(30), last\_name varchar(30), middle\_name varchar(30), expirience float(11)) | Добавляет сотрудника в базу данных |
| Change\_att(first\_name varchar(30), last\_name varchar(30), middle\_name varchar(30), experience float(11)) | Изменяет один из переданных параметров(Имя, фамилия, Отчество, стаж) |
| change\_att\_employee\_first\_name  (id int,first\_name varchar(30)) | Изменяет имя сотрудника |
| change\_att\_employee\_last\_name  (id int,last\_name varchar(30)) | Изменяет фамилию сотрудника |
| сhange\_att\_employee\_middle\_name  (id int,middle\_name varchar(30)) | Изменяет Отчество сотрудника |
| change\_att\_employee\_expirience  (id int,expirience float(11)) | Изменяет стаж сотрудника |
| del\_ employee(id\_employee int(11)) | Удаляет сотрудника по идентификатору |
| get\_information\_employee(first\_name varchar(30), last\_name varchar(30), middle\_name varchar(30), expirience float) | Возвращает данные о сотруднике. |
| add\_scanner(name varchar(30), operation\_system(30), producer(30)) | Добавляет сканнер в базу данных |
| change\_att\_scanner\_producer  (id int, producer varchar(30)) | Изменяет производителя сканнера |
| change\_att\_scanner\_name  (id int, model varchar(30)) | Изменяет название сканнера |
| change\_att\_scanner\_operation\_system  (id int, operation\_system varchar(30)) | Изменяет операционную систему сканнера |
| Change\_att(name varchar(30), operation\_system(30), producer(30)) | Изменяет один из переданных параметров(название, производитель, операционная система) |
| del\_scanner(id\_scanner int(2)) | Удалить сканнер по идентификатору |
| add\_item(name varchar(30), amount int unsigned, trend varchar(30)) | Добавляет товар в базу данных |
| change\_att(name varchar(30), amount int unsigned, trend varchar(30)) | Изменяет один из переданных параметров(название, производитель, операционная система) |
| del\_item(id\_item int(10)) | Удаляет товар по идентификатору |
| Change\_att\_item\_name (id int, name varchar(30)) | Изменяет название товара |
| change\_att\_item\_trend(id int, trend varchar(30) | Изменяет направление товара |
| Change\_att\_item\_amount(id int, amount int unsigned) | Изменяет количество товара |

## 2.3 Разработка базы данных

(Приводите пример выполнения запросов к таблицам, запросов к представлениям, вызовы процедур и функций)

### 2.3.1 Разработанные таблицы

(Приводите диаграмму таблиц из MySQL Workbench, а также скриншоты выполнения скриптов, то есть то, что они реально создаются)



### 2.3.2 Разработанные представления

(Приводите диаграмму представлений из MySQL Workbench, а также скриншоты выполнения скриптов, то есть то, что они реально создаются)

1 - CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `informationaboutauditors` AS

SELECT

`auditor`.`first\_name` AS `first\_name`,

`auditor`.`last\_name` AS `last\_name`,

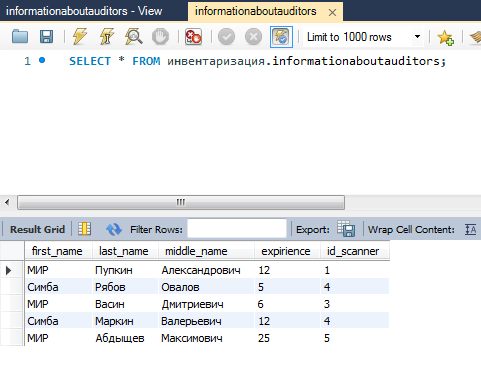
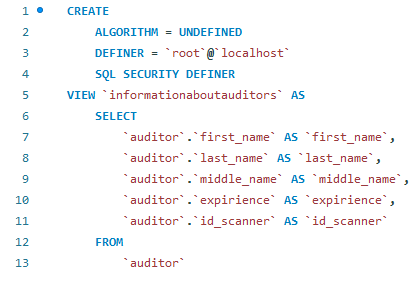
`auditor`.`middle\_name` AS `middle\_name`,

`auditor`.`expirience` AS `expirience`,

`auditor`.`id\_scanner` AS `id\_scanner`

FROM

`auditor`

2 - CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `informationaboutemployee` AS

SELECT

`e`.`first\_name` AS `first\_name`,

`e`.`last\_name` AS `last\_name`,

`e`.`middle\_name` AS `middle\_name`,

`e`.`expirience` AS `expirience`,

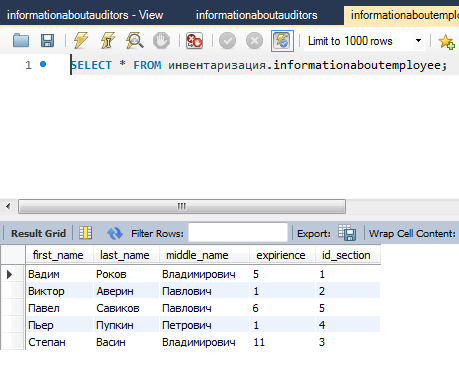
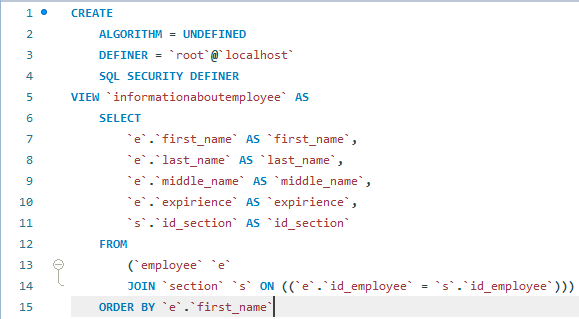
`s`.`id\_section` AS `id\_section`

FROM

(`employee` `e`

JOIN `section` `s` ON ((`e`.`id\_employee` = `s`.`id\_employee`)))

ORDER BY `e`.`first\_name`

### 2.3.3 Разработанные процедуры и функции

(Приводите список созданных процедур и функций из MySQL Workbench, а также скриншоты выполнения скриптов, то есть то, что они реально создаются)

DELIMITER $$

CREATE DEFINER=`Ревизор`@`%` PROCEDURE `scanner\_producer`()

BEGIN

update scanner set producer='Maxwell'

where id\_scanner=4;

END$$

DELIMITER ;

# 3 ОХРАНА ТРУДА И ТЕХНИКА БЕЗОПАСНОСТИ

(Не трогайте, вам потом самим скажут что сюда вставлять)

# ЗАКЛЮЧЕНИЕ

В результате выполнения ВКР была разработана …

В процессе выполнения ВКР были выполнены следующие задачи:

1)

2)

3)

4) …

Достоинства разработанной базы данных:

Недостатки разработанной базы данных:

# СПИСОК ЛИТЕРАТУРЫ

(Сюда вставляете все источники литературы, которые указывали в отчётах практических работ 1-4)

# ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД ОБЪЕКТОВ БАЗЫ ДАННЫХ

Таблица А.1 – Исходный код создания таблиц

|  |
| --- |
| **example\_base.sql** |
| CREATE DATABASE BooksBase CHARACTER SET utf8 COLLATE utf8\_general\_ci;  use `BooksBase`;  CREATE TABLE Books (  idBook int(11) PRIMARY KEY AUTO\_INCREMENT,  caption varchar(255) NOT NULL,  year int(4) NOT NULL,  countPages int(5) NOT NULL,  idGenre int(5) NOT NULL,  INDEX(idGenre)  ) ENGINE = InnoDB;  CREATE TABLE Genres (  idGenre int(11) PRIMARY KEY AUTO\_INCREMENT,  caption varchar(255) NOT NULL UNIQUE,  INDEX(idGenre)  ) ENGINE = InnoDB;  ALTER TABLE `Books`  ADD CONSTRAINT `FK1\_Books`  FOREIGN KEY (`idGenre`) REFERENCES `Genres` (`idGenre`)  ON DELETE CASCADE ON UPDATE CASCADE; |

Таблица А.2 – Исходный код создания хранимых процедур

|  |
| --- |
| **example\_procedures.sql** |
| use `BooksBase`;  DELIMITER //  CREATE PROCEDURE addBook(\_caption varchar(255), \_year int(4), \_countPages int(5), \_idGenre int(11))  BEGIN  INSERT INTO Books  (`caption`, `year`, `countPages`, `idGenre`)  VALUE  (\_caption, \_year, \_countPages, \_idGenre);  END;  CREATE PROCEDURE changeBookCaption(\_idBook int(11), newCaption varchar(255))  BEGIN  UPDATE `Books` SET `caption`=newCaption WHERE `idBook`=\_idBook;  END;  CREATE PROCEDURE changeBookYear(\_idBook int(11), newYear varchar(255))  BEGIN  UPDATE `Books` SET `year`=newYear WHERE `idBook`=\_idBook;  END;  CREATE PROCEDURE changeBookCountPages(\_idBook int(11), \_countPages int(5))  BEGIN  UPDATE `Books` SET `countPages`=\_countPages WHERE `idBook`=\_idBook;  END;  CREATE PROCEDURE removeBook(\_idBook int(11))  BEGIN  DELETE FROM `Books` WHERE `idBook`=\_idBook;  END;  CREATE PROCEDURE addGenre(\_caption varchar(255))  BEGIN  INSERT INTO Genre  (`caption`)  VALUE  (\_caption);  END;  CREATE PROCEDURE removeGenre(\_idGenre int(11))  BEGIN  DELETE FROM `Genre` WHERE `idGenre`=\_idGenre;  END;  //  /\* Для вызова процедур использовать CALL \*/  /\* CALL addBook("Эдда", 1151, 200, 2); \*/  /\* Для удаления процедур использовать DROP PROCEDURE \*/  /\* DROP PROCEDURE addBook;\*/  DELIMITER ; |

Таблица А.3 – Исходный код создания представлений

|  |
| --- |
| **example\_views.sql** |
| use `BooksBase`;  CREATE VIEW viewBooks  AS  SELECT b.caption, b.year, b.countPages, g.caption as 'Genre' FROM `Books` as b  INNER JOIN `Genres` as g ON b.idGenre=g.idGenre  ORDER BY b.caption |