Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Модели данных и системы управления базами данных

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту
на тему

# БАЗА ДАННЫХ СИСТЕМЫ ОБРАБОТКИ ЗАКАЗОВ ДЛЯ МАЛОГО БИЗНЕСА

Студент: группы 953505
Кореневский С.А.
Руководитель: ассистент каф. информатики
Плиска В. С.

Минск 2022

Заведующий кафедрой ИИТП
_____Волорова Н. А.
«___» _____ 2022 г.

# ЗАДАНИЕ
## по курсовому проекту

Группа 953505

Студенту *Кореневскому Станиславу Александровичу*

**1.Тема проекта**: База данных системы обработки заказов для малого бизнеса.

**2.Сроки сдачи студентом законченного проекта:** 21.12.2022 г.

**3.Исходные данные к проекту:** Для написания курсового проекта была выбрана база данных PostgreSQL. В качестве среды разработки была выбрана интегрированная среда разработки DataGrip.

**4.Содержание расчетно-пояснительной записки** (перечень подлежащих разработке вопросов):

Введение

Раздел 1. Обзор предметной области

Раздел 2. Разработка концептуальной модели

Раздел 3. Разработка логической модели

Раздел 4. Разработка физической модели

Раздел 5. Формирование представлений

Раздел 6. Разработанные триггеры

Заключение. Список использованных источников. Приложение

**5.Перечень графического материала** (с указанием обязательных чертежей и графиков):

_____

**6.Консультанты по проекту**: Плиска В. С.

**7.Дата выдачи задания**: 10.09.2022 г

**8.Календарный график работы над проектом на весь период проектирования** (с указанием сроков выполнения и трудоемкости отдельных этапов):

| № п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
|---|---|---|---|
| 1. | Разработка общего положения | 10.10.2022 | 30% |
| 2. | Разработка программного продукта | 14.11.2022 | 65% |
| 3. | Разработка инструкции по эксплуатации | 30.11.2022 | 100% |
| 4. | Защита курсового проекта | 19.12.2022 | Согласно графику |

Руководитель _____ (Плиска В. С.)

Задание принял к исполнению 10.09.2022 _____ (_____)

# СОДЕРЖАНИЕ

# ВВЕДЕНИЕ

В настоящее время в городах находится большое число различного вида кафе, и других мест, что может вызывать некоторые проблемы: посетители запутываются в выборе мест, не могут найти подходящее места или не замечают их, владельцы бизнеса же часто не имеют возможностей сбора большого количества данных, а также их последующей обработки и анализа в целях получения некоторой полезной для бизнеса информации.

В этой связи, полезно приложение, которое свяжет места малого бизнеса с конечными потребителями. Это позволяет уменьшить “порог входа” на рынок, владельцам бизнеса качественнее и быстрее развиваться, а людям — получать бонусы, быстрее и легче находить интересующие их места для проведения времени.

В данной курсовой работе будет описана и разработана БД для хранения и репрезентации данных системы обработки заказов для малого бизнеса. Будут изучены основные принципы построения БД для поставленной задачи.

# 1 ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

## 1.1 Описание бизнес сущностей и их правил

Список сущностей используемых в базе данных:
- Компания.
- Предложение.
- Пост.
- Точка.
- Админ.
- Пользователь.
- Комментарий.
- Рейтинг.
- Способ оплаты.
- Заказ.
- Аудит заказа.
- Оплата заказа.
- Имя товара.
- Товар.
- Товар из заказа.
- Группа.
- Разрешение.
- Краткий комментарий.
- Краткие комментарии точки.
- Похожие пользователи.
- Похожие компании.
- Похожие точки.
- Похожие товары.
- Рекомендации точек пользователем.
- Рекомендации товаров пользователем.
- Статистика пользователя.
- Статистика точки.
- Рекомендация изменения продукта.
- Оценка заработка точки.
- Лог.

Сущности из списка выше будут определять правила, по которым данные и сама БД будет формироваться. Их поведение должно быть отражено в структуре разрабатываемой БД максимально близко реальности, не противореча выделенным правилам. Список правил выглядит следующим образом:

1 У одной компании может быть несколько точек.

2 Одна точка принадлежит одной компании.

3 У каждой точки может быть несколько админов.

4 Каждый пользователь может оставлять не ограниченное количество

комментариев.

5 Каждый пользователь может выставлять рейтинг точке только один раз, а далее обновлять существующий рейтинг.

6 Компания должна иметь возможность создавать несколько постов и несколько предложений.

7 Клиент может иметь несколько карт.

8 Вести историю состояний заказа.

# 2 РАЗРАБОТКА КОНЦЕПТУАЛЬНОЙ МОДЕЛИ

## 2.1 Общее описание концептуально модели и ее целей

Также следует отметить, что концептуальная модель, которая является начальной стадией в разработке всей модели, должна разрабатываться в рамках функциональных требований к приложению и, помимо законов предметной области, удовлетворять и им. Иначе разрабатываемая в рамках приложения модель быстро потеряет суть, а без нее и все программное обеспечение станет бесполезным [2].

Ниже указаны требования выдвигаемые к базе данных, необходимые для полноценной работы приложения. Они выведены из технического задания, а именно из функциональных требований, так как без модели данных невозможно качественно реализовать пользовательский опыт.

На основании предметной области и функциональных требований к приложению, выведем требования к разрабатываемой модели данных:

1 Необходимо хранить информацию о пользователях, компаниях и их точках.

2 Необходимо хранить статистику поведения пользователей и посещения точек.

3 Необходимо хранить схожесть компаний, точек, пользователей и продуктов.

4 Необходимо хранить основные важные фразы из комментариев и их оценку.

5 При добавлении нового товара в заказ, автоматически обновлять общую стоимость заказа, и статистику пользователей, и точки.

6 При создании или изменении заказа необходимо сохранять изменение в другую таблицу.

7 Необходимо хранить лог.

## 2.2 Функциональные требования к базе данных

В результате анализа предметной области было получено множество сущностей, которое необходимо отразить в разрабатываемой модели. При этом крайне важно брать в расчет их роль и связь друг с другом в рамках предметной области и бизнес-правила, которые они собой порождают, что обеспечит высокое качество их поведение в сравнении с реальными объектами.

Кратко перечислим основные сущности и их роли, которые необходимо отразить в концептуальной модели:

- Компания: сущность, хранящая информацию о компании малого бизнеса владеющей некоторыми точками.
- Точка: сущность, хранящая информацию о месте обслуживания компании.

- Пост: сущность, хранящее описание некоторого поста от компании и доступного всем пользователям.
- Предложение: сущность, хранящая некоторое предложение компании всем пользователям.
- Админ: сущность, хранящая информацию о работниках точки.
- Пользователь: сущность которая хранит информацию всех пользователей системы (работников точек, админов системы обслуживания, людей использующих приложение).
- Комментарий: сущность, в которой хранятся комментарии пользователей.
- Рейтинг: сущность, которая хранит оценку точки, выставленную пользователям.
- Способ оплаты: сущность, хранящая номер карты пользователя.
- Заказ: сущность, которая хранит информацию о заказе.
- Аудит заказа: сущность, хранящая историю обновления заказа.
- Оплата заказа: сущность, хранящая информацию о времени проведения заказа.
- Имя товара: сущность которая хранит все имена товаров всех точек.
- Товар: сущность которая хранит информацию о товаре некоторой точки.
- Товар из заказа: сущность которая хранит количество заказанного товара.
- Группа: сущность которая определяет группы с дополнительными правами доступа пользователей.
- Разрешение: сущность, определяющая дополнительные права доступов.
- Краткий комментарий: сущность, хранящая выжимку из комментариев по всем точкам.
- Краткий комментарий точки: сущность, объединяющая точку и краткие комментарии, относящиеся к ней.
- Похожие пользователи: сущность, хранящая похожих пользователей и их степень схожести.
- Похожие точки: сущность, хранящая похожие точки и их степень схожести.
- Похожие компании: сущность, хранящая похожие компании и их степень схожести.
- Похожие продукты: сущность, хранящая похожие продукты и их степень схожести.
- Рекомендация точек пользователям: сущность, хранящая точки рекомендованные пользователям вместе со степенью рекомендации.
- Рекомендация товаров пользователям: сущность, хранящая товары рекомендованные пользователям вместе со степенью рекомендации.
- Статистика пользователя: сущность, хранящая статистику пользователей.

- Статистика точки: сущность, хранящая статистику точек.
- Рекомендация изменения продукта: сущность, хранящая возможные рекомендации продуктов точек.
- Оценка разработка точки: сущность хранящая оценку заработков точек.
- Лог: сущность хранящая информацию о всех действиях приложения.

Составленная концептуальная схема, которая удовлетворяет правилам предметной области и требованиям, выдвигаемым к модели данных функциональными требованиями, приведена на рисунке 2.1.

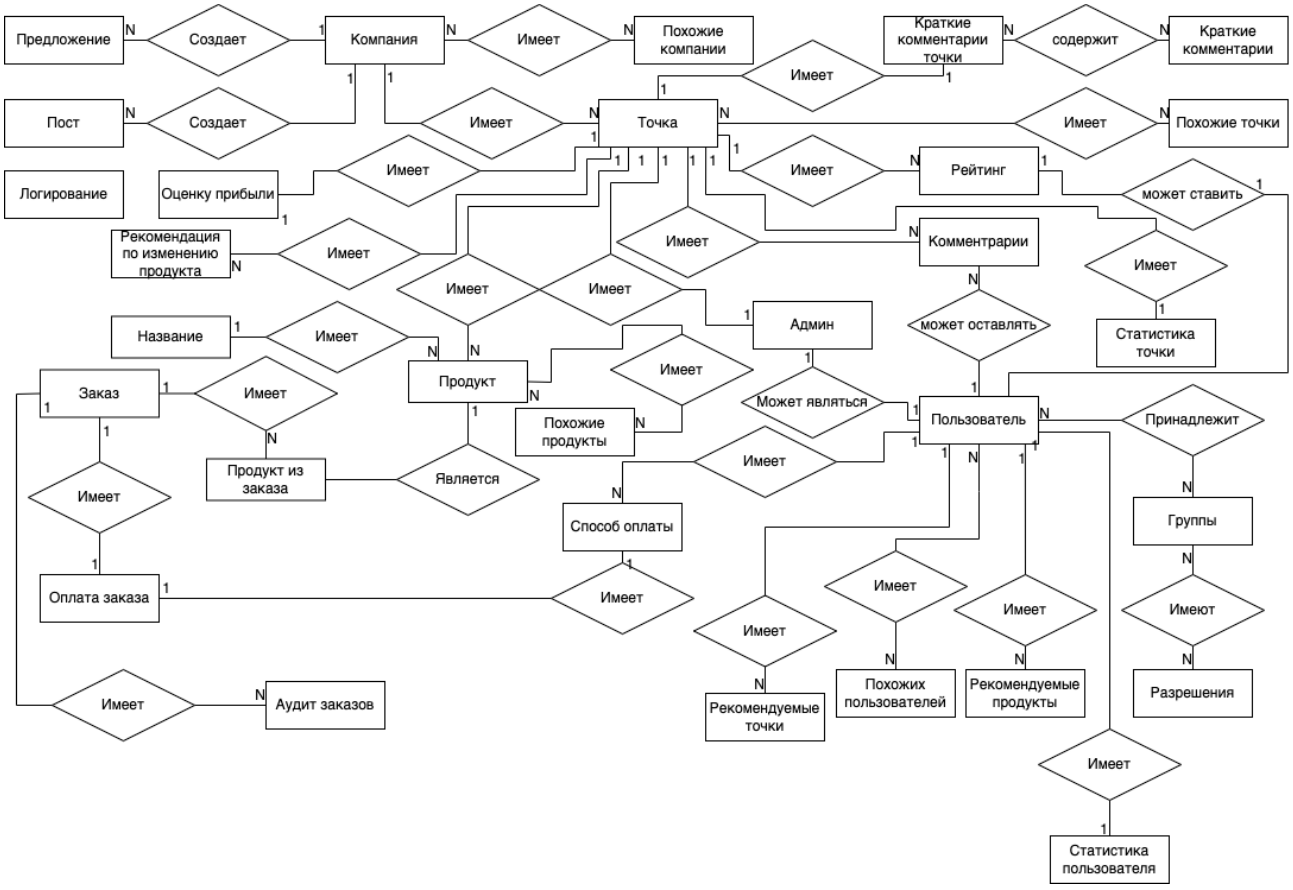


Рисунок 2.1 — Концептуальная схема модели данных

# 3 РАЗРАБОТКА ЛОГИЧЕСКОЙ МОДЕЛИ

## 3.1 Общее описание целей логической модели

Логическая модель является следующим этапом проектирования модели данных с учетом принятия типа используемой базы данных, а также с более детальным описанием связей и атрибутов, но без уточнения технических деталей используемой базы данных.

В разработке логической модели используется методология IDEF1X, которая ориентирована на проектирование реляционной модели базы данных. Она призвана отобразить сущности предметной области в виде непосредственных таблиц проектируемой базы данных, а также отобразить поля и их роли соответствующих таблиц, будь то первичные или внешние ключи, хранение бизнес или логических данных, требуемых самой моделью [3].

## 3.2 Логическое проектирование

В рамках логической модели каждая сущность получает развитие в виде детального описание содержащихся в ней полей, а также происходит переход семантических связей в более формализованный вид, соответствующий реляционной модели. Следует отметить, что множественность связей между таблицами логической модели должна соответствовать связям, описанным в концептуальной модели.

На основании концептуальной модели, предметной области и функциональных требований к реляционной базе данных, была получена логическая модель, представленная на рисунке 3.1.

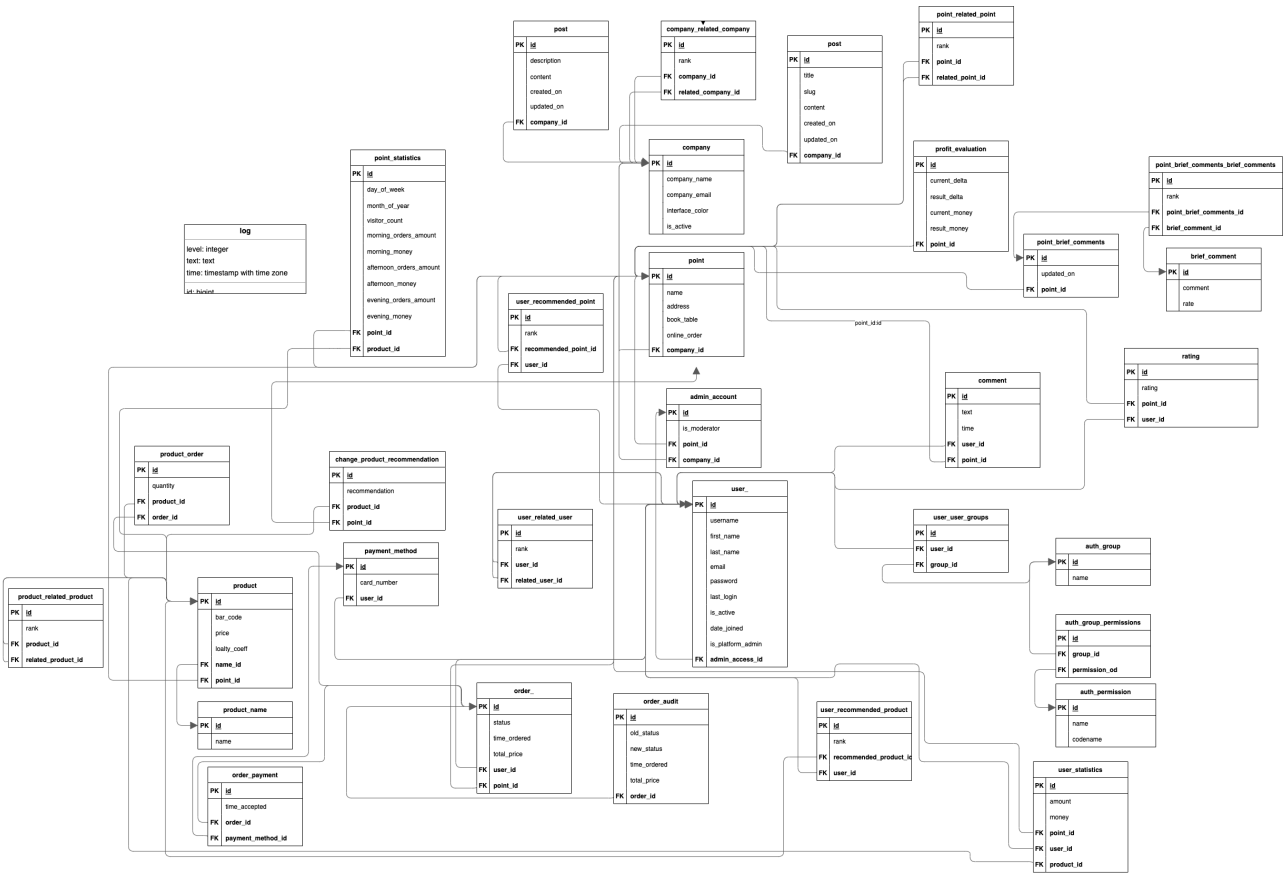


Рисунок 3.1 — Логическая схема модели базы данных

## 3.3 Спецификация логической модели

На этапе проектирования логической модели были отображены атрибуты сущностей описываемой модели. Определим их суть и назначения в таблицах 3.1 — 3.30:

Таблица 3.1 — Компания (Company)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор пользователя. |
| Company_name | | | Название компании. |
| Bussines_email | | | Почта компании. |
| Interface_color | | | Цвет интерфейса. |
| Is_active | | | Статус активности компании. |

Таблица 3.2 — Точка (Point)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор точки. |
| Company_id | | X | Идентификатор компании. |
| Name | | | Имя точки. |
| Address | | | Адрес точки. |
| Book_table | | | Статус возможности бронирования места. |
| Online_order | | | Статус возможности заказа онлайн. |

Таблица 3.3 — Предложения (Offer)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Company_id | | X | Идентификатор компании. |
| Description | | | Краткое описание предложения. |
| Content | | | Основной текст предложения. |
| Created_on | | | Дата создания предложения. |
| Updated_on | | | Дата обновления предложения. |

Таблица 3.4 — Пост (Post)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Company_id | | X | Идентификатор компании. |
| Title | | | Заголовок. |
| Slug | | | Ссылка на некоторый ресурс. |
| Content | | | Основной текст. |
| Created_on | | | Дата создания предложения. |
| Updated_on | | | Дата обновления предложения. |

Таблица 3.5 — Админ (Admin_account)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Company_id | | X | Идентификатор компании. |
| Point_id | | X | Идентификатор точки. |
| Is_moderator | | | Статус возможности редактирования. |

Таблица 3.6 — Пользователь (User_)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Username | | | Уникальное имя пользователя. |
| First_name | | | Имя пользователя. |
| Last_name | | | Фамилия пользователя. |
| Email | | | Почта пользователя. |
| Password | | | Пароль пользователя. |
| last_login | | | Дата последней активности пользователя в сети. |
| Is_active | | | Статус активности пользователя. |
| Date_joined | | | Дата регистрации пользователя на пллатформе. |
| Is_platform_admin | | | Статус, показывающий является ли пользователь админом платформы. |
| Admin_access_id | | X | Идентификатор |

Таблица 3.7 — Комментарий (Comment)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Point_id | | X | Идентификатор точки. |
| User_id | | X | Идентификатор пользователя. |
| text | | | Текст комментария. |

| | | | |
|---|---|---|---|
| time | | | Дата написания комментария. |

Таблица 3.8 — Рейтинг (Rating)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Point_id | | X | Идентификатор точки. |
| User_id | | X | Идентификатор пользователя. |
| Rating | | | Числовое значение рейтинга, изменяющегося от 0 до 5 включительно. |

Таблица 3.9 — Способ оплаты (Payment_method)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| User_id | | X | Идентификатор пользователя. |
| Card_number | | | Номер карточки. |

Таблица 3.10 — Заказ (Odder_)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| User_id | | X | Идентификатор пользователя. |
| Point_id | | X | Идентификатор точки. |
| Status | | | Статус заказа. |
| Time_ordered | | | Время оформления заказа. |
| Total_price | | | Общая сумма заказа. |

Таблица 3.11 — Аудит заказа (Odder_audit)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| User_id | | X | Идентификатор пользователя. |

| | | | |
|---|---|---|---|
| Point_id | | X | Идентификатор точки. |
| Old_status | | | Предшествующий статус заказа. |
| New_status | | | Измененный татус заказа. |
| Time_ordered | | | Время оформления заказа. |
| Total_price | | | Общая сумма заказа. |

Таблица 3.12 — Оплата заказа (Order_payment)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Order_id | | X | Идентификатор заказа. |
| Order_method_id | | X | Идентификатор метода оплаты заказа. |
| Time_accepted | | | Время оформления платежа по заказу. |

Таблица 3.13 — Имя товара (Product_name)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Name | | | Имя товара. |

Таблица 3.14 — Товар (Product)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Name_id | | X | Уникальный идентификатор имени товара. |
| Point_id | | X | Идентификатор точки. |
| Bar_code | | | Штрих-код продукта. |
| Price | | | Цена продукта. |
| Loyalty_coeff | | | Коэффициент бонусов, изменятся от 0 до 1 включительно. |

Таблица 3.15 — Товар из заказа (Product_order)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Order_id | | X | Идентификатор заказа. |
| Product_id | | X | Идентификатор продукта. |
| Quantity | | | Количество продукта из заказа. |

Таблица 3.16 — Группа (group)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Name | | | Имя группы. |

Таблица 3.17 — Разрешение (Permission)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Name | | | Имя разрешения. |
| Codename | | | Код разрешения. |

Таблица 3.18 — Краткий комментарий (Brief_comment)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Comment | | | Некоторая фраза из комментария. |
| Rate | | | Коэффициент оценки эмоциональности комментария. Изменяется от 0 до 1 включительно. |

Таблица 3.19 — Краткие комментарии точки (Point_brief_comments)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |

| | | | |
|---|---|---|---|
| Point_id | X | | Уникальный идентификатор точки. |
| Updated_on | | | Дата обновления выжимки из комментариев. |

Таблица 3.20 — Похожие пользователи(User_related_user)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| User_id | | X | Идентификатор пользователя. |
| Related_user_id | | X | Идентификатор пользователя. |
| Rank | | | Номер в очереди схожести пользователей. |

Таблица 3.21 — Похожие компании (Company_related_company)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Company_id | | X | Идентификатор компании. |
| Related_company_id | | X | Идентификатор компании. |
| Rank | | | Номер в очереди схожести компаний. |

Таблица 3.22 — Похожие точки (Point_related_point)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Point_id | | X | Идентификатор точки. |
| Related_point_id | | X | Идентификатор точки. |
| Rank | | | Номер в очереди схожести пользователей. |

Таблица 3.23 — Похожие продукты (Product_related_product)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Product_id | | X | Идентификатор продукта. |

| | | | |
|---|---|---|---|
| Related_product_id | | X | Идентификатор продукта. |
| Rank | | | Номер в очереди схожести продуктов. |

Таблица 3.24 — Рекомендации точек пользователям (User_recommended_point)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Point_id | | X | Идентификатор точки. |
| Recomended_point_id | | X | Идентификатор точки. |
| Rank | | | Номер в очереди схожести точек. |

Таблица 3.25 — Рекомендации продуктов пользователям (User_recommended_product)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Product_id | | X | Идентификатор продукта. |
| Recomended_product_id | | X | Идентификатор продукта. |
| Rank | | | Номер в очереди схожести продуктов. |

Таблица 3.26 — Статистика пользователя (User_statistics)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Product_id | | X | Идентификатор продукта. |
| Point_id | | X | Идентификатор точки. |
| User_id | | X | Идентификатор продукта. |
| Amount | | | Количество раз которые данный пользователей купил данный продукт в данном заведении. |
| Money | | | Количество денег потраченных на данный продукт в данном заведении данным пользователем. |

Таблица 3.27 — Статистика точки (Point_statistics)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Product_id | | X | Идентификатор продукта. |
| Point_id | | X | Идентификатор точки. |
| User_id | | X | Идентификатор продукта. |
| Day_of_week | | | Номер дня в неделе, в который принимается заказ |
| Month_of_year | | | Номер месяца в году, в который принимается заказ |
| Visitor_count | | | Количество посетивших пользователей заведение в данный день и в данную часть дня. |
| Morning_orders_amount | | | Количество заказов утром в данный день недели данного месяца. |
| Afternoon_orders_amount | | | Количество заказов днем в данный день недели данного месяца. |
| Evening_orders_amount | | | Количество заказов вечером в данный день недели данного месяца. |
| Morning_money | | | Количество денег заработанных утром в данный день недели данного месяца. |
| Afternoon_money | | | Количество денег заработанных днем в данный день недели данного месяца. |
| Evening_money | | | Количество денег заработанных вечером в данный день недели данного месяца. |

Таблица 3.28 — Рекомендация изменения продукта (Change_product_recommendation)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Product_id | | X | Идентификатор продукта. |
| Point_id | | X | Идентификатор точки. |
| Recomendation | | | Текст рекомендации. |

Таблица 3.29 — Оценка заработка точки (Profit_evaluation)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Point_id | | X | Идентификатор точки. |
| Current_delta | | | Процент изменения прибыли по сравнению с предыдущим месяцем. |
| Result_delta | | | Предсказание процента изменения прибыли в конце месяца. |
| Current_money | | | Изменения прибыли по сравнению с предыдущим месяцем. |
| Result_money | | | Предсказание прибыли в конце месяца. |

Таблица 3.30 — Лог (Log)

| Название поля | Первичный ключ | Внешний ключ | Описание |
|---|---|---|---|
| Id | X | | Уникальный идентификатор предложения. |
| Level | | | Уровень логирования. |
| Text | | | Текст лога. |
| Time | | | Время логирования. |

# 4 РАЗРАБОТКА ФИЗИЧЕСКОЙ МОДЕЛИ

## 4.1 Общие цели и содержание физической модели

В результате двух предыдущих этапов моделирования разрабатываемая модель приобрела новый вид и достаточно точно определяет содержание сущностей и их взаимодействие. Однако она еще не готова для создания и запуска, так как не учитывает технические детали определенной системы управления базами данных. Цель физического проектирование определить, каким образом представляется информация в указанной базе данных.

Результатом данного шага является создание конечной схемы со всеми необходимыми деталями для создания модели в реальной базе данных.

## 4.2 Обзор реляционной базы данных PostgreSQL

PostgreSQL — свободная объектно-реляционная система управления базами данных (СУБД).

Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows.

Сильными сторонами PostgreSQL считаются:

- высокопроизводительные и надёжные механизмы транзакций и репликации;
- расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются PL/pgSQL, PL/Perl, PL/Python и PL/Tcl; дополнительно можно использовать PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme, PL/sh и PL/V8, а также имеется поддержка загрузки модулей расширения на языке C;
- наследование;
- возможность индексирования геометрических (в частности, географических) объектов и наличие базирующегося на ней расширения PostGIS;
- встроенная поддержка слабоструктурированных данных в формате JSON с возможностью их индексации;
- расширяемость (возможность создавать новые типы данных, типы индексов, языки программирования, модули расширения, подключать любые внешние источники данных).

### 4.3 Физическое проектирование

В рамках физической модели таблицы, отношения и колонки получают детали специфичные для конкретной базы данных, в данном случае для PostgreSQL. Физическая схема является готовым описанием всех таблиц, связей необходимых индексов, типов, ограничений и кластеров. Получаемое в результате данного этапа описание, готово к использованию при создании артефактов базы данных.

На основании логической модели, правил предметной области и функциональных требований к реляционной базе данных, была получена физическая модель.

# 5 ФОРМИРОВАНИЕ ПРЕДСТАВЛЕНИЙ

## 5.1 Информация о пользователе, его группах и номере карточки

```
create or replace view user_group_and_credentials as
select username, first_name, last_name, email, name as group_name, card_number from user_
join user_groups ug
    on user_.id = ug.user_id
join auth_group ag
    on ug.group_id = ag.id
join payment_method pm
    on user_.id = pm.user_id;
```

## 5.2 Информация о компании, ее точках и предсказании доходов

```
create or replace view company_point_statistics as
select company_name, name as point_name, address, book_table, online_order, current_delta, result_delta,
       current_money, result_money
from company
join point p on company.id = p.company_id
join profit_evaluation pe on p.id = pe.point_id
join point_statistics ps on p.id = ps.point_id
```

## 5.3 Информация об основных фразах комментариев и их рейтингов для точки

```
create or replace view points_brief_comments as
select point_id, updated_on, comment, rate from point_brief_comments
join point_brief_comments_brief_comments m2m
```

```
    on point_brief_comments.point_id = m2m.point_brief_comments_id
join brief_comment bc on m2m.brief_comment_id = bc.id;
```

# 6 РАЗРАБОТАННЫЕ ТРИГГЕРЫ

## 6.1 Триггер для ведения таблицы Orders_audit

```
-- Order_audit function
create or replace function order_trigger_function()
    returns trigger
    language plpgsql
as $$
begin
    insert into public.order_audit (order_id, old_status, new_status, time_ordered, total_price)
    values (new.id, old.status, new.status, new.time_ordered, new.total_price);

    return new;
end;
$$;


-- Order_audit trigger
create or replace trigger order_audit_trigger
    before insert or update on public.order_
    for each row
    execute procedure order_trigger_function();
```

**6.2 Триггер для обновления статистики и суммы заказа, при обновлении продуктов заказа**

```
-- Point_statistics & User_statistics function
create or replace function product_order_trigger_function()
    returns trigger
    language plpgsql
as $$
    declare
        _point_id bigint;
        _user_id bigint;
        _price numeric(6, 2);
        _time_ordered timestamp with time zone;
        _hour integer;
        _morning_orders_amount integer = 0;
        _afternoon_orders_amount integer = 0;
        _evening_orders_amount integer = 0;
        _morning_money numeric(6, 2) = 0;
        _afternoon_money numeric(6, 2) = 0;
        _evening_money numeric(6, 2) = 0;
        _day_of_week integer;
        _month_of_year integer;
begin
    select point_id, user_id, time_ordered
    into _point_id, _user_id, _time_ordered
    from order_
    where id = new.order_id;
```

```
select price
into _price
from product
where id = new.product_id;

_month_of_year := extract(month from _time_ordered);
_day_of_week := extract(dow from _time_ordered);
_hour := extract(hour from _time_ordered);
if _hour <= 12 then
   _morning_money := _price * new.quantity;
   _morning_orders_amount := new.quantity;
elsif _hour <= 18 then
   _afternoon_money := _price * new.quantity;
   _afternoon_orders_amount := new.quantity;
else
   _evening_money := _price * new.quantity;
   _evening_orders_amount := new.quantity;
end if;

--  update Order total_price
update order_ set total_price = (total_price + _price * new.quantity) where
order_.id = new.order_id;



--  insert or update User_statistics
if (select count(*)
   from public.user_statistics
```

```
            where point_id = _point_id and product_id = new.product_id and user_id =
_user_id) = 0 then
                insert into public.user_statistics (point_id, product_id, user_id, amount,
money)
                values (_point_id, new.product_id, _user_id, new.quantity, _price *
new.quantity);
            else
                update public.user_statistics
                set amount = amount + new.quantity, money = money + _price *
new.quantity
                where point_id = _point_id and product_id = new.product_id and user_id =
_user_id;
            end if;



            --  insert or update Point_statistics
            if (select count(*)
                from public.point_statistics
                where point_id = _point_id and product_id = new.product_id and
day_of_week = _day_of_week and
                    month_of_year = _month_of_year) = 0 then
                insert into public.point_statistics (point_id, product_id, day_of_week,
month_of_year, visitor_count,
                                                     morning_orders_amount, morning_money,
afternoon_orders_amount,
                                                     afternoon_money, evening_orders_amount,
evening_money)
                values (_point_id, new.product_id, _day_of_week, _month_of_year, 1,
_morning_orders_amount, _morning_money,
```

```
                    _afternoon_orders_amount, _afternoon_money,
_evening_orders_amount, _evening_money);
        else
            update public.point_statistics
            set visitor_count = visitor_count + 1, morning_orders_amount =
morning_orders_amount + _morning_orders_amount,
                morning_money = morning_money + _morning_money,
afternoon_money = afternoon_money + _afternoon_money,
                afternoon_orders_amount = afternoon_orders_amount +
_afternoon_orders_amount,
                evening_orders_amount = evening_orders_amount +
_evening_orders_amount,
                evening_money = evening_money + _evening_money
            where point_id = _point_id and product_id = new.product_id and
day_of_week = _day_of_week and
                 month_of_year = _month_of_year;
        end if;

        return new;
    end;
    $$;


    -- Point_statistics & User_statistics trigger
    create or replace trigger product_order_statistics_trigger
        before insert on public.product_order
        for each row
        execute procedure product_order_trigger_function();
```

# ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы была спроектирована и создана модель данных в рамках предметной области для системы обработки заказов для малого бизнеса. Были разработаны 3 схемы в соответствии со спецификациями: концептуальная, логическая и физическая. Созданная модель соответствует выдвинутым функциональным требованиям приложения и выделенным правилам предметной области. Она предлагает эффективную структуру для вывода данных.

Были описаны методы по формированию представлений для описанной модели и разработанны триггеры. В результате работы были изучены основные принципы проектирования базы данных и способы ее описания.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Дейт, К. Дж., Введение в системы баз данных / Дж. К. Дейт – Москва.: Издательский дом «Вильямс», 2001. – 1072 с.

[2] Сидорова, Н. П. Базы данных: практикум по проектированию реляционных баз данных: учебное пособие / Н. П. Сидорова – Москва: Директ-медиа, 2020. – 92 с.

[3] Ильюшечкин, В. М. Основы использования и проектирования баз данных / В. М. Ильюшечкин – М.: Юрайт, 2019. – 213 с.

## ПРИЛОЖЕНИЕ А — Листинг кода

```
-- Log schema
create table public.log
(
    id    bigint generated by default as identity
        constraint log_pk
        primary key,
    level integer not null
        constraint log_level
        check(level = 0 or level = 10 or level = 20 or level = 30 or level = 40 or level = 50),
    text  text not null,
    time  timestamp with time zone not null
);

alter table public.log
    owner to postgres;


-- Company schema
create table public.company
(
    id  bigint generated by default as identity
        constraint company_pk
        primary key,
    company_name    varchar(30) not null,
    business_email  varchar(64) not null,
```

```
    interface_color varchar(10) not null,
    is_active       boolean     not null
);

alter table public.company
    owner to postgres;


-- Point schema
create table public.point
(
    id  bigint generated by default as identity
        constraint point_pk
        primary key,
    company_id  bigint not null
                constraint point_company_id_fk_company_id
                references public.company
                deferrable initially deferred,
    name varchar(50) not null,
    address varchar(50) not null,
    book_table boolean not null,
    online_order boolean not null
);

alter table public.point
    owner to postgres;
```

```
create index point_company_id_index
    on public.point (company_id);


-- Admin_account schema
create table public.admin_account
(
    id  bigint generated by default as identity
        constraint admin_account_pk
        primary key,
    company_id bigint not null
            constraint admin_account_company_id_fk_company_id
            references public.company
            deferrable initially deferred,
    point_id bigint
            constraint admin_account_point_id_fk_point_id
            references public.point
            deferrable initially deferred,
    is_moderator boolean not null
);

alter table public.admin_account
    owner to postgres;

create index admin_account_company_id_index
    on public.admin_account (company_id);
```

```
create index admin_account_point_id_index
    on public.admin_account (point_id);



-- User schema
create table public.user_
(
    id  bigint generated by default as identity
        constraint user_pk
        primary key,
    username varchar(150) not null
            constraint user_username_key unique,
    first_name varchar(20) not null,
    last_name varchar(20) not null,
    email varchar(254) not null
         constraint user_email_key unique,
    password  varchar(128) not null,
    last_login timestamp with time zone,
    is_active boolean not null,
    date_joined timestamp with time zone not null,
    is_platform_admin boolean not null,
    admin_access_id bigint
                constraint user_admin_access_id_fk_admin_account_id
                references public.admin_account
                deferrable initially deferred
);
alter table public.user_
```

```
    owner to postgres;

create index user_username_index
    on public.user_ (username varchar_pattern_ops);

create index user_email_index
    on public.user_ (email varchar_pattern_ops);

create index user_admin_access_id_index
    on public.user_ (admin_access_id);

-- Comments table
create table public.comment
(
    id  bigint generated by default as identity
        constraint comment_pk
        primary key,
    point_id bigint not null
            constraint comment_point_id_fk_point_id
            references public.point
            deferrable initially deferred,
    user_id  bigint not null
            constraint comment_user_id_fk_user_id
            references public.user_
            deferrable initially deferred,
    text     text not null,
    time     timestamp with time zone not null
```

```
);

alter table public.comment
    owner to postgres;

create index comment_point_id_index
    on public.comment (point_id);

create index comment_user_id_index
    on public.comment (user_id);


-- Rating table
create table public.rating
(
    id  bigint generated by default as identity
        constraint rating_pk
        primary key,
    point_id bigint not null
            constraint rating_point_id_fk_point_id
            references public.point
            deferrable initially deferred,
    user_id  bigint        not null
            constraint rating_user_id_fk_user_id
            references public.user_
            deferrable initially deferred,
    rating numeric(3, 2) not null
```

```
        constraint rating_rating_check
        check(rating >= 0 and rating <= 5),
    constraint point_id_user_id_unique
        unique (point_id, user_id)
);

alter table public.rating
    owner to postgres;

create index rating_point_id_index
    on public.rating (point_id);

create index rating_user_id_index
    on public.rating (user_id);

-- Payment_method schema
create table public.payment_method
(
    id  bigint generated by default as identity
        constraint payment_method_pk
        primary key,
    user_id bigint not null
            constraint payment_method_user_id_fk_user_id
            references public.user_
            deferrable initially deferred,
    card_number varchar(16) not null
);
```

| Обозначение | Наименование | Дополнительные сведения |
|---|---|---|
| БГУИР КП 1-40 04 01 023 ПЗ | «База данных системы обработки заказов для малого бизнеса» | 38 |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

| | | | | | БГУИР КП 1-40 04 01 082 Д1 | | | |
|---|---|---|---|---|---|---|---|---|
| Изм. | Л | № докум. | Подп. | Дата | «База данных системы обработки заказов для малого бизнеса» | | Лист | Листов |
| Разраб. | | Студент | Кореневский С. А. | 19.12.2022 | | Т | 38 | 38 |
| Пров. | | Руково-дитель | Плиска В.С. | 19.12.2022 | | Кафедра информатики гр. 953505 | | |
| | | | | | | | | |