**Содержание**

[**ВВЕДЕНИЕ** 2](#_Toc161701179)

[**1. Аналитический обзор литературы и формирование требований** 3](#_Toc161701180)

[**1.1. Анализ прототипов** 3](#_Toc161701181)

[**1.1.1. Пицца лисицца** 3](#_Toc161701182)

[**1.1.2. Додо пицца** 4](#_Toc161701183)

[**1.1.3. Пицца темпо** 4](#_Toc161701184)

[**1.2. Требования к проекту** 5](#_Toc161701185)

[**2. Разработка программной системы** 6](#_Toc161701186)

[**2.1. Проектирование базы данных** 6](#_Toc161701187)

[**2.2. Хранимые процедуры** 9](#_Toc161701188)

[**2.3. Функции** 10](#_Toc161701189)

[**2.4. Представления** 10](#_Toc161701190)

[**2.5. Технологии базы данных** 11](#_Toc161701191)

[**Заключение** 12](#_Toc161701192)

[**Список литературы** 13](#_Toc161701193)

## **ВВЕДЕНИЕ**

В современном мире пиццерии становятся все более популярными и востребованными местами, где люди могут насладиться вкусной пиццей и провести время с семьей и друзьями. Если рассматривать эту отрасль с точки зрения предпринимателей и владельцев пиццерий, то возникает необходимость в эффективной системе управления и автоматизации процессов, связанных с заказами, доставкой и учетом продуктов.

Целью данного проекта было разработка реляционной базы данных для пиццерии. Эта база данных предназначалась для обеспечения пиццерии функциями управления заказами, инвентарем, доставкой и другими операциями.

База данных — это организованная структура, предназначенная для хранения информации, систематизированная таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины. Реляционная база данных — база данных, основанная на реляционной модели данных. В качестве СУБД для базы данных была выбрана Oracle, в связи с ее высокой производительностью и надежностью.

В основной части будут затронуты все аспекты разработки проекта и обоснованы некоторые технические приёмы, к которым приходилось прибегнуть, с целью реализации работы веб-сервера с базой данных.

Для обеспечения работы приложения использовались ASP.NET Core в связке с WPF.

Основные требования к приложению:

* Регистрация и авторизация пользователя.
* Составление заказа.
* Выбор услуги.
* Поиск по услугам и товарам.
* Просмотр каталога (меню).
* Просмотр корзины и статуса заказа.
* Просмотр истории заказов.
* Составление отзыва к заказу.

В пояснительной записке вы сможете найти краткую информацию о похожих продуктах, архитектуре, реализации проекта, руководстве пользователя.

## **Аналитический обзор литературы и формирование требований**

## **Анализ прототипов**

Были проанализированы цели и задачи, поставленные в данном курсовом проекте, а также рассмотрены аналогичные примеры их решений. На основании анализа всех достоинств и недостатков данных альтернативных решений были сформулированы требования к базе данных.

# **Пицца лисицца**

Самым популярным и широко известным ресурсом в мире пиццы является интернет-платформа "Пицца лисицца". Этот ресурс предоставляет уникальные возможности для любителей пиццы и профессиональных пиццемейстеров.

Главным преимуществом "Пиццы лисиццы" является обширный каталог пицц, включающий множество разнообразных рецептов и вариаций этого блюда. Пользователи ресурса имеют возможность подробно ознакомиться с каждым блюдом, изучить их состав, рейтинг и отзывы, чтобы сделать осознанный выбор. Также стоит отметить, что данный сервис может предоставлять функциональность персонализации заказа, позволяя клиентам добавлять или убирать ингредиенты, выбирать размер порции и настраивать свои предпочтения. Это создает индивидуальный опыт для каждого клиента.

Из выявленных недостатков данного ресурса необходимо выделить недоступность и медленную загрузку страниц из-за большого количества клиентов.

Интерфейс интернет-ресурса представлен на рисунке 1.1.

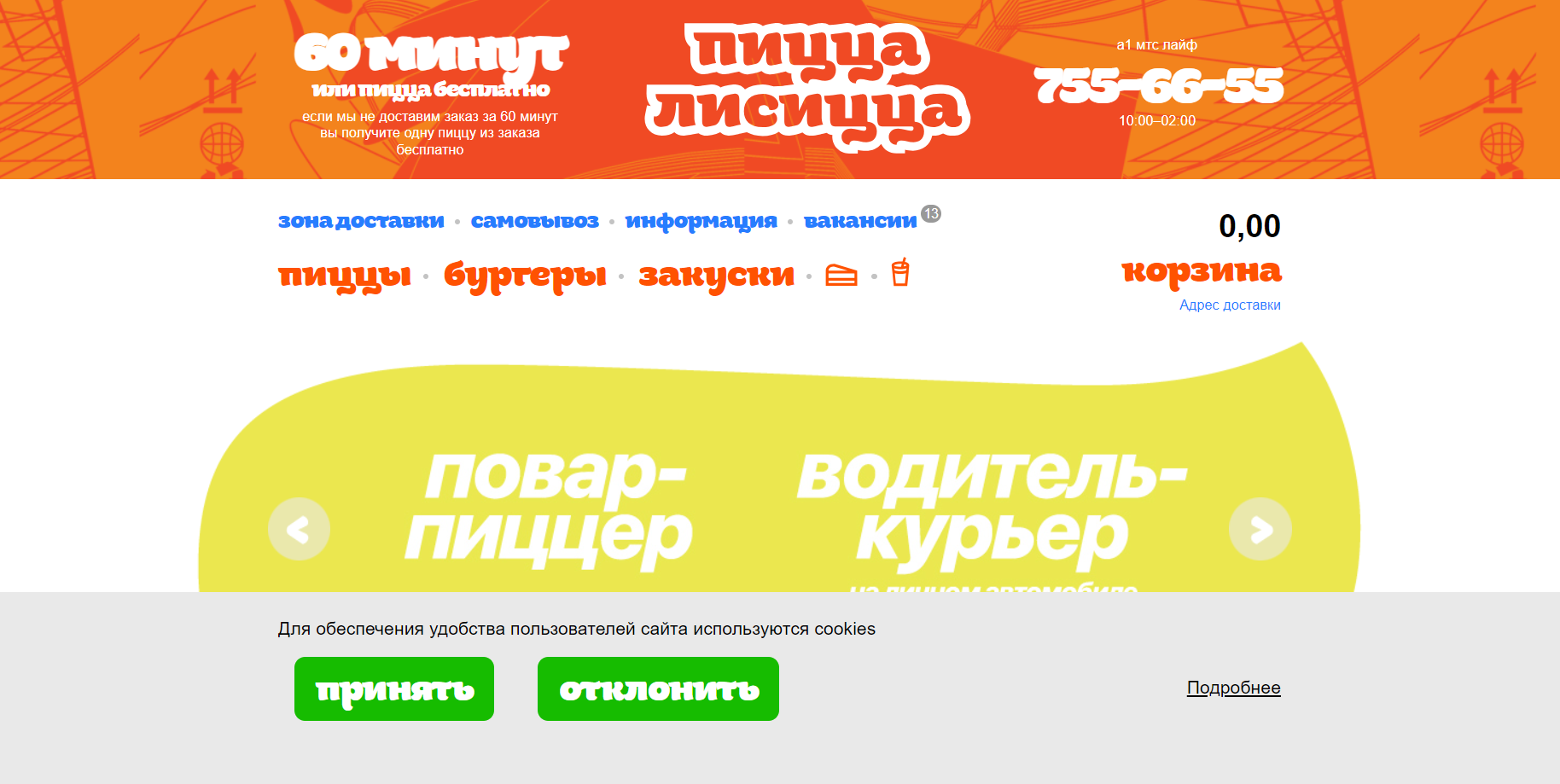


Рисунок 1.1 – Интернет-ресурс «Пицца лисицца»

# **Додо пицца**

Еще одним альтернативным решением задач, поставленных в данном курсовом проекте, является интернет-ресурс «Додо пицца». Данное программное средство предоставляет пользователю функцию отслеживания заказа, которая позволяет клиентам узнать текущий статус и ожидаемое время доставки. Это помогает клиентам быть в курсе процесса и позволяет им планировать свое время. Еще одним достоинством является выбор варианта оплаты, который включает в себя кредитные карты, электронные платежные системы или наличные при доставке. Это обеспечивает удобство и выбор для клиентов.

Одним из главных недостатков выбора такой альтернативы является отсутствие предоставления информации о пищевой ценности каждой пиццы, такую как калории, содержание жиров, углеводов и прочее. Это полезно для клиентов, которые следят за своим питанием и хотят знать о составе заказываемой пиццы.

Интерфейс интернет-ресурса представлен на рисунке 1.2.

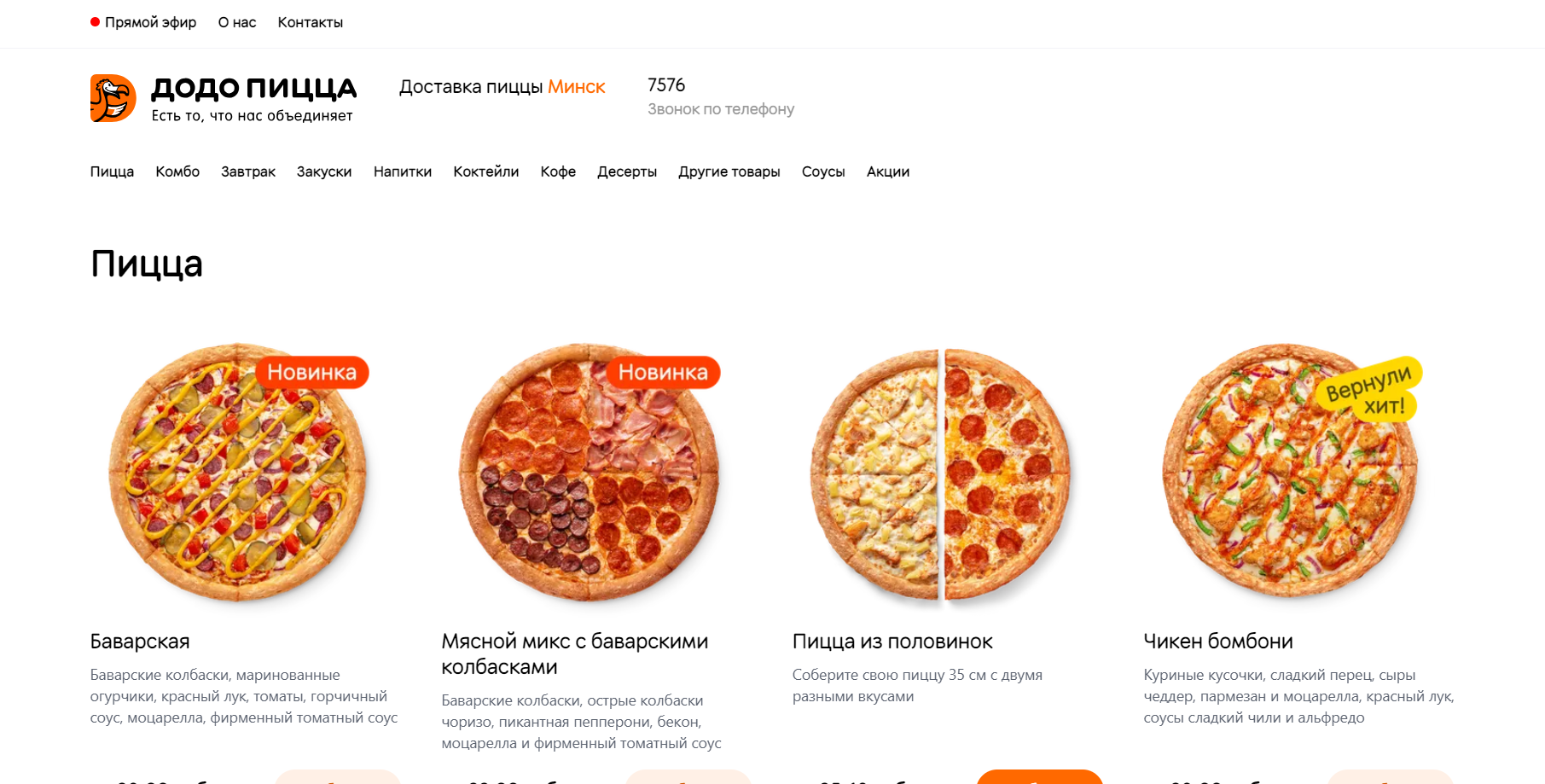


Рисунок 1.2 – Интернет-ресурс «Додо пицца»

# **Пицца темпо**

Ещё одной альтернативой решения поставленных задач является интернет-ресурс «Пицца темпо». С помощью данного программного средства пользователь может ознакомиться со списком доступных для заказа пицц, а также с помощью сайта оптимизированного для мобильных устройств, может заказать пиццу с помощью своего смартфона или планшета в любом месте и в любое время. Немаловажным преимуществом является то, что сервис предоставляет возможность предлагать клиентам специальные акции, скидки или программы лояльности. Это позволяет клиентам получать дополнительные преимущества и стимулирует их к повторным заказам.

Однако недостатками данной альтернативы являются узкая направленность только на продажу пицц и отсутствие другого меню.

Интерфейс интернет-ресурса представлен на рисунке 1.3.

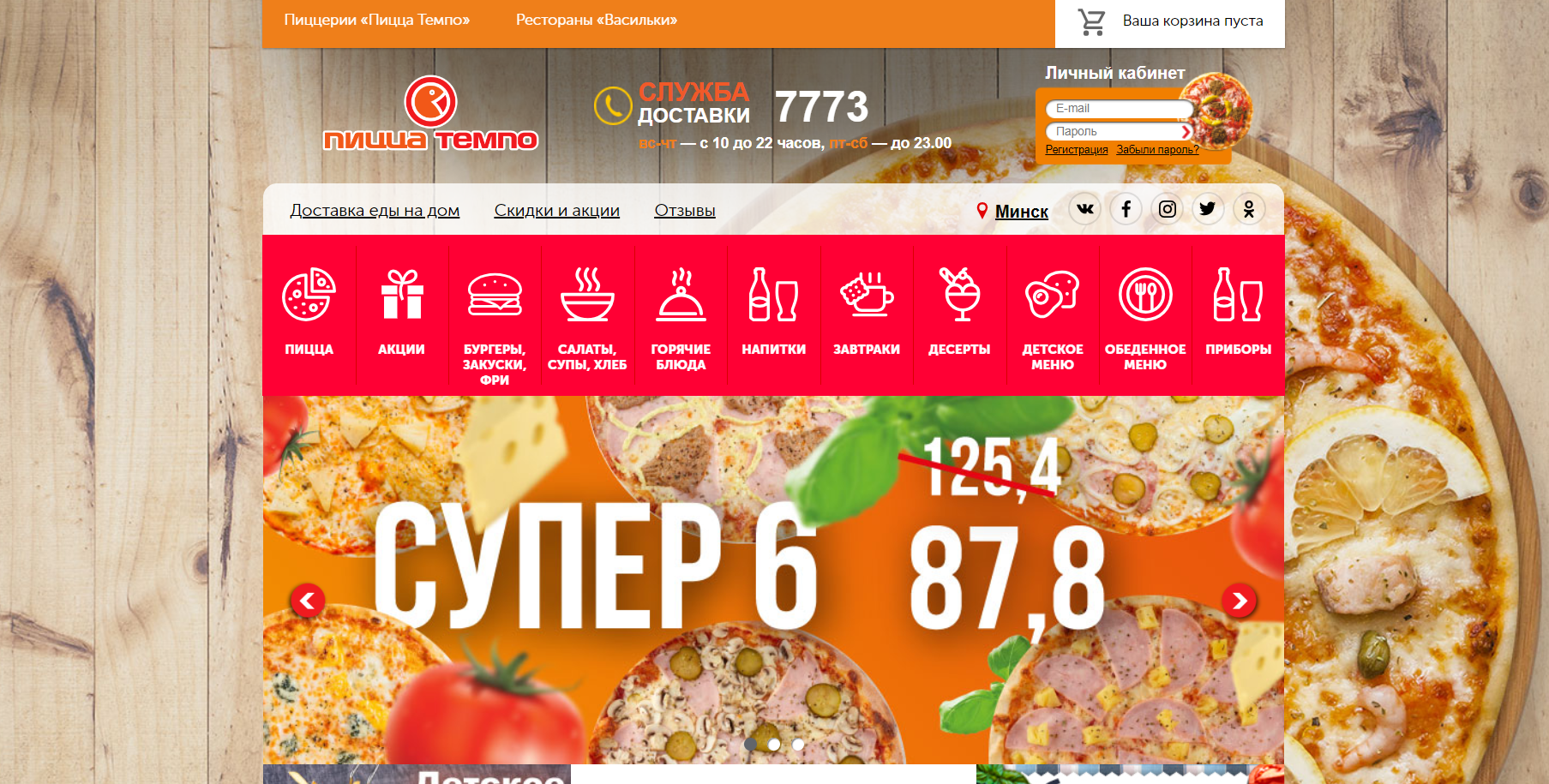


Рисунок 1.3 – Интернет-ресурс «Пицца темпо»

## **Требования к проекту**

Обзор вышеперечисленных известных аналогов позволяет проанализировать все преимущества и недостатки альтернативных возможностей и позволяет сформулировать список требований, предъявляемых к разработке базы данных в данном курсовом проекте. База данных должна обеспечивать возможность выполнения перечисленных ниже функций:

* Регистрация и авторизация пользователя.
* Добавление и обновление информации о товарах.
* Удаление товаров.
* Уведомление о событиях в базе данных.
* Составление заказа.
* Выбор услуги.
* Поиск по услугам и товарам.
* Просмотр каталога (меню).
* Просмотр корзины и статуса заказа.
* Просмотр истории заказов.
* Составление отзыва к заказу.

## **Разработка программной системы**

# **Проектирование базы данных**

Процесс проектирования базы данных состоит из следующих этапов:

* сбор информации;
* определение сущностей;
* определение атрибутов для каждой сущности;
* определение связей между сущностями;
* нормализация.

На этапе сбора информации вам необходимо точно определить, как будет использоваться база данных, и какая информация будет в ней храниться.

Далее следует этап определения сущностей и на нем определяются сущности, из которых будет состоять база данных.

Сущность — это объект в базе данных, в котором хранятся данные. Сущность может представлять собой нечто вещественное (дом, человек, предмет, место) или абстрактное (банковская операция, отдел компании, маршрут автобуса). В физической модели сущность называется таблицей. Сущности состоят из атрибутов (столбцов таблицы) и записей (строк в таблице).

Обычно базы данных состоят из нескольких основных сущностей, связанных с большим количеством подчиненных сущностей. Основные сущности называются независимыми: они не зависят ни от какой-либо другой сущности. Подчиненные сущности называются зависимыми: для того, чтобы существовала одна из них, должна существовать связанная с ней основная таблица.

Следующий этап — это определение атрибутов созданных сущностей. Атрибут представляет свойство, описывающее сущность. Атрибуты часто бывают числом, датой или текстом. Все данные, хранящиеся в атрибуте, должны иметь одинаковый тип и обладать одинаковыми свойствами.

В физической модели атрибуты называют колонками. После определения сущностей необходимо определить все атрибуты этих сущностей. На диаграммах атрибуты обычно перечисляются внутри прямоугольника сущности.

На этапе определения связей работа происходит с ключами сущностей. Ключом называется набор атрибутов, однозначно определяющий запись. Ключи делятся на два класса: простые и составные.

Простой ключ состоит только из одного атрибута. Например, в базе «Паспорта граждан страны» номер паспорта будет простым ключом: ведь не бывает двух паспортов с одинаковым номером.

Составной ключ состоит из нескольких атрибутов. В той же базе «Паспорта граждан страны» может быть составной ключ со следующими атрибутами:  
фамилия, имя, отчество, дата рождения. Это — как пример, т. к. этот составной ключ, теоретически, не обеспечивает гарантированной уникальности записи.

Первичным ключом называется совокупность атрибутов, однозначно идентифицирующих запись в таблице (сущности). Один из возможных ключей становится первичным ключом. На диаграммах первичные ключи часто изображаются выше основного списка атрибутов или выделяются специальными символами.

Любой возможный ключ, не являющийся первичным, называется альтернативным ключом. Сущность может иметь несколько альтернативных ключей. Внешним ключом называется совокупность атрибутов, ссылающихся на первичный или альтернативный ключ другой сущности. Если внешний ключ не связан с первичной сущностью, то он может содержать только неопределенные значения. Если при этом ключ является составным, то все атрибуты внешнего ключа должны быть неопределенными. На диаграммах атрибуты, объединяемые во внешние ключи, обозначаются специальными символами.

Реляционные базы данных позволяют объединять информацию, принадлежащую разным сущностям. Отношение — это ситуация, при которой одна сущность ссылается на первичный ключ второй сущности. Они определяются в процессе проектирования базы. Для этого следует проанализировать сущности и выявить логические связи, существующие между ними.

Тип отношения определяет количество записей сущности, связанных с записью другой сущности. Отношения делятся на три основных типа:

* Один-к-одному. Каждой записи первой сущности соответствует только одна запись из второй сущности. А каждой записи второй сущности соответствует только одна запись из первой сущности. Например, есть две сущности: Люди и Свидетельства о рождении. И у одного человека может быть только одно свидетельство о рождении.
* Один-ко-многим. Каждой записи первой сущности могут соответствовать несколько записей из второй сущности. Однако каждой записи второй сущности соответствует только одна запись из первой сущности. Например, есть две сущности: Заказ и Позиция заказа. И в одном заказе может быть много товаров.
* Многие-ко-многим. Каждой записи первой сущности могут соответствовать несколько записей из второй сущности. Однако и каждой записи второй сущности может соответствовать несколько записей из первой сущности. Например, есть две сущности: Автор и Книга. Один автор может написать много книг. Но у книги может быть несколько авторов.

По критерию обязательности отношения делятся на обязательные и необязательные.

* Обязательное отношение означает, что для каждой записи из первой сущности непременно должны присутствовать связанные записи во второй сущности.
* Необязательное отношение означает, что для записи из первой сущности может и не существовать записи во второй сущности.

Далее следует этап нормализации. Нормализацией называется процесс удаления избыточных данных из базы данных. Каждый элемент данных должен храниться в базе в одном и только в одном экземпляре. Существует пять распространенных форм нормализации. Как правило, база данных приводится к третьей нормальной форме.

В процессе нормализации выполняются определенные действия по удалению избыточных данных. Нормализация повышает быстродействие, ускоряет сортировку и построение индекса, уменьшает количество индексов на сущность, ускоряет операции вставки и обновления.

Нормализованная база данных обычно отличается большей гибкостью. При модификации запросов или сохраняемых данных в нормализованную базу обычно приходится вносить меньше изменений, а внесение изменений имеет меньше последствий.

Чтобы преобразовать сущность в первую нормальную форму, следует исключить повторяющиеся группы значений и добиться того, чтобы каждый атрибут содержал только одно значение, списки значений не допускаются.  
Другими словами, каждый атрибут, в сущности, должен храниться только в одном экземпляре.

Для соответствия второй нормальной форме сущности должны быть в первой нормальной форме. Таблица во второй нормальной форме содержит только те данные, которые к ней относятся. Значения не ключевых атрибутов сущности зависят от первичного ключа. Если более точно, то атрибуты зависят от первичного ключа, от всего первичного ключа и только от первичного ключа.

В третьей нормальной форме исключаются атрибуты, не зависящие от всего ключа. Любая сущность, находящаяся в третьей нормальной форме, находится также и во второй. Это самая распространенная форма базы данных.  
В третьей нормальной форме каждый атрибут зависит от ключа, от всего ключа и ни от чего, кроме ключа.

Ограничения — это правила, за соблюдением которых следит система управления базы данных. Ограничения определяют множество значений, которые можно вводить в столбец или столбцы.

Организовав данные в таблицы и определив связи между ними, можно считать, что была создана модель, правильным образом отражающая бизнес-среду. Теперь нужно обеспечить, чтобы данные, вводимые в базу, давали правильное представление о состоянии дела. Иными словами, нужно обеспечить выполнение деловых правил и поддержку целостности базы данных.

Например, ваша компания занимается доставкой книг. Вы вряд ли примете заказ от неизвестного клиента, ведь тогда вы даже не сможете доставить заказ. Отсюда бизнес-правило: заказы принимаются только от клиентов, информация о которых есть в базе данных.

Корректность данных в реляционных базах обеспечивается набором правил. Правила целостности данных делятся на четыре категории.

Для реализации необходимого функционала была создана база данных, таблицы которой можно разделить на 3 логические группы: таблицы для хранения информации о продуктах, таблицы для хранения информации о заказах пользователей и таблицы для хранения пользовательских данных.

Рассмотрим таблицы базы данных на примере диаграммы, представленной на рисунке 2.1.

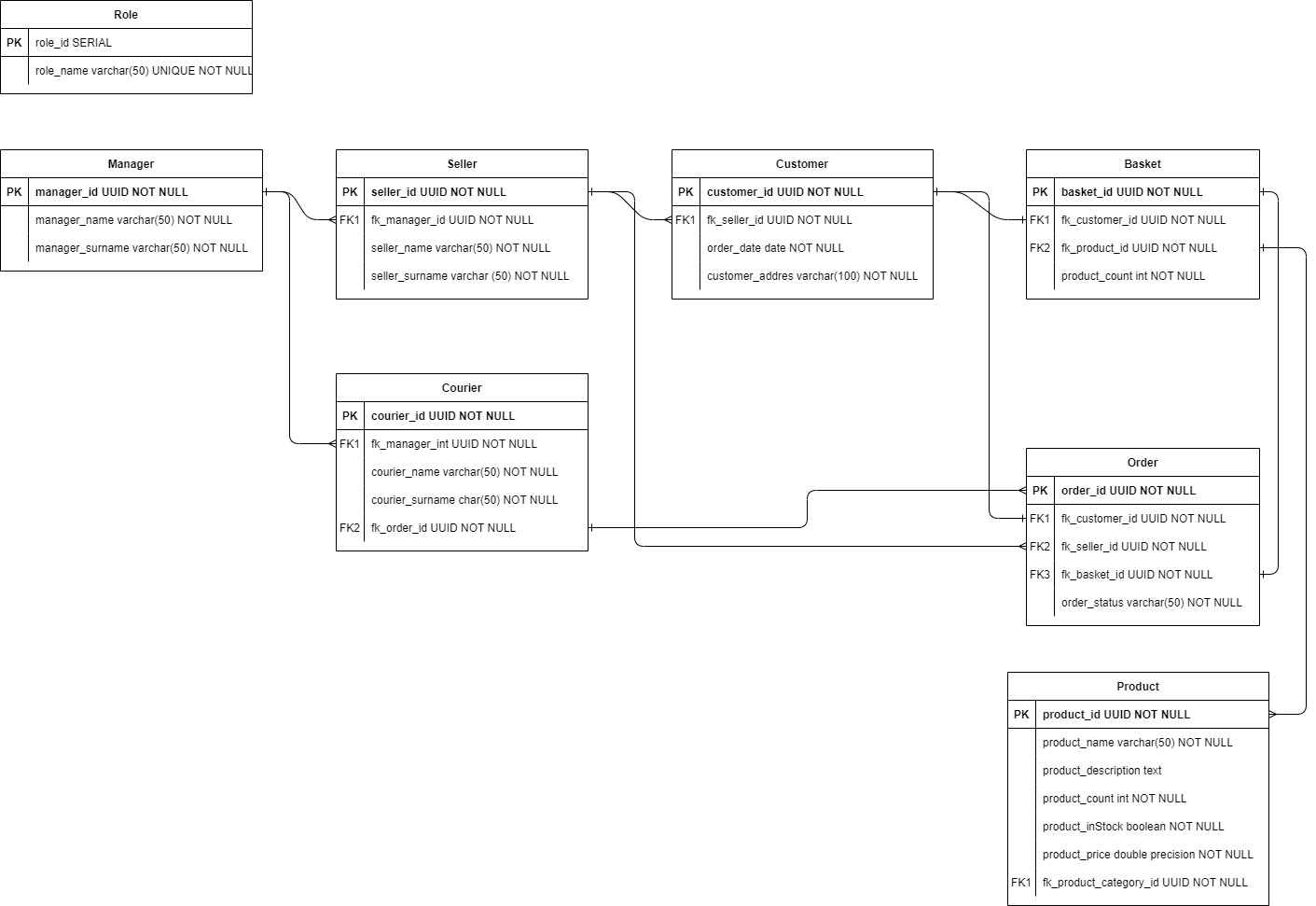


Рисунок 2.1 – Диаграмма таблиц базы данных

База данных состоит из 8 таблиц, связанных между собой внешними ключами.

Таблица «*ROLES*» хранит информацию о ролях, которые могут назначаться поставщиках услуг: Admin, Employee, User.

Таблица «Customer» хранит в себе всю информацию о пользователях, то есть клиентах, включая номер заказа, адрес для доставки (если таковой имеется) и т.д.

Таблица «Manager» хранит информацию о главном менеджере с его личной информацией, также на него будут ссылать его работники.

Таблица «Seller» хранит информацию о самом продавце и его менеджере.

Таблица «Courier» хранит информацию о самом курьере, его менеджере и его заказе.

Таблица «Basket» хранит информацию о составленной клиентом корзине содержащей список товаров и их количество.

Таблица «Product» хранит информацию о каждом товаре и всю его информацию, включая название, описание, наличие и количество (если таковое имеется), а также цену с категорией товара.

Таблица «Order» хранит информацию о заказе включающую саму корзину, статус заказа, а также ссылки на его участников.

# **Хранимые процедуры**

Хранимая процедура – объект базы данных, представляющий собой набор SQL-инструкций, который компилируется один раз и хранится на сервере.

Наиболее используемые процедуры представлены в приложении Б.

При разработке курсового проекта было создано большое количество процедур для следующих целей:

Для пользователя:

* регистрация;
* авторизация;
* выбор услуги;
* просмотр меню;
* создание заказа;
* оплаты заказа;
* получения списка заказов в корзине;
* получения статуса заказа;

Для поставщика услуг:

* авторизация;
* просмотр информации о товарах;
* получения статуса заказа;
* создания, редактирование и удаление товаров;
* подтверждения заказа пользователя.

# **Функции**

Функция SQL — это набор операторов SQL, которые принимают входные данные и выполняют с ними действия SQL, а затем возвращают результаты в виде выходных данных.

Функции выполняют утилитарные дополнительные функции и служат для упрощенной выборки некоторых часто получаемых данных, так как они позволяют удобно получить некоторые данные:

* ….
* ……
* …..
* …..
* …
* …

# **Представления**

Представление - это виртуальная таблица, основанная на результирующем наборе инструкции SQL.

Они содержит строки и столбцы, как и настоящая таблица. Поля в представлении - это поля из одной или нескольких реальных таблиц в базе данных.

Представления используются для абстракции от соединения таблиц и сокрытия их реализации внутри доступных пользователям процедур.

* ...
* …
* …

# **Технологии базы данных**

В данном подразделе будут описаны технологии, использованные в процессе разработки приложения. В программном средстве была реализована технология отправки email уведомления о событиях базы данных.

# **Заключение**

Результатом выполнения курсового проекта стала база данных в .

Были реализованы основные требования, указанные в листе задания вместе со следующими пунктами:

* Регистрация и авторизация пользователя;
* Поиск услуги;
* Оформление заказа;
* Уведомление пользователя о заказе;
* Просмотр корзины;
* Просмотр статуса заказа;
* Импорт и экспорт данных таблицы в JSON формат;
* Заполнение таблицы на 100 000 записей;
* Технология email уведомления о событиях базы данных.

Проект не может окончательно претендовать на полноценный коммерческий продукт, поскольку для демонстрации технологии и работы с базой данных были опущены некоторые атрибуты современных площадок, реализация которых не относится к основной теме курсового проекта (например, платежные системы).

Протестировав все компоненты приложения, можно прийти к заключению, что все основные требования выполнены и приложение работает исправно.

# **Список литературы**

1. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: https://metanit.com – Дата доступа: 15.10.2021.

2. DBMS\_XMLDOM [Электронный ресурс] / Режим доступа: <https://docs.oracle.com/cd/B1930601/appdev.102/b14258/dxmldom.htm#i1076719> – Дата доступа: 20.10.2022.

3. Developing and Using Stored Procedures [Электронный ресурс] / Режим доступа: <https://docs.oracle.com/cd/B2835901/appdev.111/b28843/tdddgprocedures.htm> – Дата доступа: 26.10.2022.

4. Stackoverflow.com [Электронный ресурс] / Режим доступа: https://stackoverflow.com – Дата доступа: 08.12.2022

5. Блинова, Е.А. Курс лекций по Базам данным / Е.А. Блинова. – Минск: БГТУ, 2019. – 175 с.