МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Забайкальский государственный университет»
(ФГБОУ ВПО «ЗабГУ»)
Факультет: Энергетический
Кафедра: Информатики, вычислительной техники и прикладной математики

КУРСОВОЙ ПРОЕКТ

по Базам данных
(наименование дисциплины)

на тему: «Разработка реляционной базы данных по выбранной предметной области (Интернет-магазин одежды)»

Выполнил студент группы
ИВТ-18-2, Долгов Александр Артемович
(группа, фамилия, имя, отчество)

Руководитель работы: _____
(должность, ученая степень, фамилия, имя, отчество)

Чита
2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Забайкальский государственный университет»
(ФГБОУ ВПО «ЗабГУ»)
Факультет: Энергетический
Кафедра: Информатики, вычислительной техники и прикладной математики

ЗАДАНИЕ
на курсовой проект

По дисциплине: Базы данных
Студенту: Долгову Александру Артемовичу
(фамилия, имя, отчество)

По специальности (направления подготовки): 09.03.01 информатика и вычислительная техника

1. Тема курсовой работы: «Разработка реляционной базы данных по выбранной предметной области (Интернет-магазин одежды)»
2. Срок подачи студентом законченной работы 31.12.20
3. Исходные данные к работе: тема курсовой работы
4. Перечень подлежащих разработке в курсовой работе вопросов:

Дата выдачи задания 1.09.2020 г.
Руководитель курсовой работы:

_____
(подпись, расшифровка подписи)

Задание принял к исполнению
« 1 » сентября 2020 г.

Подпись студента _____ / Долгов А.А. /
(Ф.И.О.)

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Забайкальский государственный университет»

(ФГБОУ ВПО «ЗабГУ»)

Факультет: Энергетический

Кафедра: Информатики, вычислительной техники и прикладной математики

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

по 09.03.01 Информатика и вычислительная техника

(наименование направления подготовки)

на тему «Разработка реляционной базы данных по выбранной предметной области (Интернет-магазин одежды)»

Выполнил студент группы ИВТ-18-2, Долгов Александр Артемович

(группа, фамилия, имя, отчество)

Руководитель работы: _____

(должность, ученая степень, фамилия, имя, отчество)

КАЛЕНДАРНЫЙ ГРАФИК

Наименование раздела курсовой работы	**Месяц, неделя**															
	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
1. Получение задания на курсовую работу		+														
2. Работа с источниками информации и подбор материала		+	+	+												
3. Написание первого раздела					+	+	+									
4. Написание второго раздела								+	+	+						
5.Оформление работы в соответствии с методическими указаниями и стандартом предприятия											+	+				
6. Представление руководителю чернового варианта работы													+			
7. Корректировка работы в соответствии с замечаниями руководителя; подготовка доклада и презентации														+	+	
8.Защита работы																+

РЕФЕРАТ

Пояснительная записка – 40с., рисунки – 9, источники – 4, таблиц – 2.

В данной курсовой работе разработана реляционная база данных по выбранной предметной области («Интернет-магазин одежды») при помощи системы управления базами данных PostgreSQL.

Перечень ключевых слов: базы данных, предметная область, «сущность-связь», СУБД, PostgreSQL, нормальные формы, нормализация, ограничения целостности, запросы, триггеры, первичный ключ, потенциальный ключ.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Компьютеры были созданы для решения вычислительных задач, однако со временем они все чаще стали использоваться для построения систем обработки документов, а точнее, содержащейся в них информации. Такие системы обычно и называют информационными. В качестве примера можно привести систему учета отработанного времени работниками предприятия и расчета заработной платы, систему учета продукции на складе, систему учета книг в библиотеке и т.д. Все вышеперечисленные системы имеют следующие особенности:

1. для обеспечения их работы нужны сравнительно низкие вычислительные мощности
2. данные, которые они используют, имеют сложную структуру
3. необходимы средства сохранения данных между последовательными запусками системы

Другими словами, информационная система требует создания в памяти ЭВМ динамически обновляемой модели внешнего мира с использованием единого хранилища - базы данных, основная функция которой будет являться хранение информации, относящейся к определенной теме – предметной области.

Предметная область - часть реального мира, подлежащая изучению с целью организации управления и автоматизации.

Таким образом, задачей курсовой работы будет являться изучение и разработка базы данных по выбранной предметной области.

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Основные термины и определения

Информационная система (ИС) — это система, предназначенная для ведения информационной модели какой-либо области человеческой деятельности (предметной области). Эта система должна обеспечивать средства для протекания информационных процессов:

— хранение,

— передача,

— поиск и преобразование информации.

Особенности информационных систем:

— для обеспечения их работы нужны сравнительно низкие вычислительные мощности;

— данные, которые они используют, имеют сложную структуру;

— необходимы средства сохранения данных между последовательными запусками системы.

Предметная область – часть реального мира, подлежащая изучению с целью организации управления и автоматизации. Представляется множеством фрагментов, каждый из которых характеризуется множеством объектов и процессов, использующих объекты, а также множеством пользователей, характеризуемых различными взглядами на предметную область.

Данные (калька от лат. *data*) — это представление фактов и идей в формализованном виде, пригодном для передачи и обработки в некотором информационном процессе. Данные – это выделенная (из системы, благодаря обособленности существования носителя) информация.

База данных (по Дж. Мартину) – совокупность взаимосвязанных данных, совместно используемых несколькими приложениями и хранящимися с (минимальной) регулируемой избыточностью. Данные запоминаются таким образом, чтобы они, по мере возможности, не зависели от программ. Для обработки данных применяется общий управляющий метод доступа. Если базы данных не пересекаются по структуре, то говорят о системе баз данных.

База данных (по материалам CODASYL**,** COnference on DAta SYstems Language) – состоит из всех экземпляров записей, экземпляров наборов записей и областей, которые контролируются конкретной схемой. (Под схемой можно понимать карту всей логической структуры базы данных)

Свойства БД:

— структурная целостность;

— непротиворечивость (семантическая целостность);

— отсутствие избыточности.

1.2 Основы проектирования

Процесс проектирования БД представляет собой последовательность переходов от неформального словесного описания информационной структуры предметной области к формализованному описанию объектов предметной области в терминах некоторой модели. Выделяют следующие этапы проектирования:

1. Системный анализ и словесное описание информационных объектов предметной области.

2. Проектирование инфологической модели предметной области — частично формализованное описание объектов предметной области в терминах некоторой семантической модели, например, в терминах ER-модели.

3. Даталогическое или логическое проектирование БД, то есть описание БД в терминах принятой даталогической модели данных.

4. Физическое проектирование БД, то есть выбор эффективного размещения БД на внешних носителях для обеспечения наиболее эффективной работы приложения.

1.3 Модель «сущность-связь»

Перед созданием системы автоматизированной обработки информации, разработчик должен сформировать понятия о предметах, фактах и событиях, и привести их к необходимой модели данных. Одним из наиболее удобных инструментов унифицированного представления данных является модель "сущность-связь" (entity - relationship model, ER - model).

В основе ER-модели лежат следующие базовые понятия:

– Сущность, с помощью которой моделируется класс однотипных объектов. Сущность имеет имя, уникальное в пределах моделируемой системы. Предполагается, что в системе существует множество экземпляров данной сущности. Объект, которому соответствует понятие сущности, имеет свой набор атрибутов — характеристик, определяющих свойства данного представителя класса. Набор атрибутов, однозначно идентифицирующий конкретный экземпляр сущности, называют ключевым.

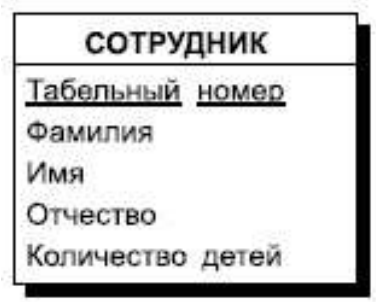

Рисунок 1.1 - Пример определения сущности в модели ER

Между сущностями могут быть установлены связи — бинарные ассоциации, показывающие, каким образом сущности соотносятся или взаимодействуют между собой.

Связи делятся на три типа по множественности:

1) один-к-одному (1:1),
2) один-ко-многим (1:M),
3) многие-ко-многим (M:M).

Связь один-к-одному означает, что экземпляр одной сущности связан только с одним экземпляром другой сущности. Связь 1: M означает, что один экземпляр сущности, расположенный слева по связи, может быть связан с несколькими экземплярами сущности, расположенными справа по связи. Связь "один-к-одному" (1:1) означает, что один экземпляр одной сущности связан только с одним экземпляром другой сущности, а связь "многие-ко-многим" (M:M) означает, что один экземпляр первой сущности может быть связан с несколькими экземплярами второй сущности, и наоборот, один экземпляр второй сущности может быть связан с несколькими экземплярами первой сущности.

Примеры различных связей приведены на рисунках 1.1 и 1.2.

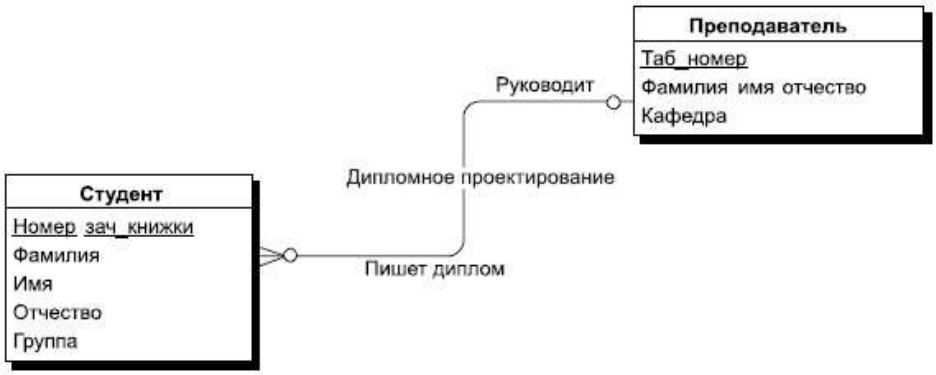

Рисунок 1.1. Пример отношения "один-ко-многим" при связывании сущностей "Студент" и "Преподаватель"

Рисунок 1.2. Пример моделирования связи "многие-ко-многим"

1.4 Нормализация БД и нормальные формы

В теории реляционных баз данных обычно выделяется следующая последовательность нормальных форм:

— первая нормальная форма (1NF);

— вторая нормальная форма (2NF);

— третья нормальная форма (3NF);

— нормальная форма Бойса-Кодда (BCNF);

— четвертая нормальная форма (4NF);

— пятая нормальная форма, или нормальная форма проекции-соединения (5NF или PJ/NF).

Основные свойства нормальных форм состоят в следующем:

— каждая следующая нормальная форма в некотором смысле лучше предыдущей нормальной формы;

— при переходе к следующей нормальной форме свойства предыдущих нормальных форм сохраняются.

В основе процесса проектирования лежит метод нормализации, т. е. декомпозиции отношения, находящегося в предыдущей нормальной форме, на два или более отношений, которые удовлетворяют требованиям следующей нормальной формы.

1.4.1 Первая нормальная форма

Отношение находится в 1НФ, если все его атрибуты являются простыми, все используемые домены должны содержать только скалярные значения. Не должно быть повторений строк в таблице.

1.4.2 Вторая нормальная форма

Отношение находится во 2НФ, если оно находится в 1НФ и каждый не ключевой атрибут неприводимо зависит от Первичного Ключа (ПК).

Неприводимость означает, что в составе потенциального ключа отсутствует меньшее подмножество атрибутов, от которого можно также вывести данную функциональную зависимость.

1.4.3. Третья нормальная форма

Отношение находится в 3НФ, когда находится во 2НФ и каждый не ключевой атрибут нетранзитивно зависит от первичного ключа. Проще говоря, второе правило требует выносить все не ключевые поля, содержимое которых может относиться к нескольким записям таблицы в отдельные таблицы.

1.4.4 Нормальная форма Бойса-Кодда (НФБК) (частная форма третьей нормальной формы)

Определение 3НФ не совсем подходит для следующих отношений:

1) отношение имеет два или более потенциальных ключа;

2) два и более потенциальных ключа являются составными;

3) они пересекаются, т.е. имеют хотя бы один общий атрибут.

Для отношений, имеющих один потенциальный ключ (первичный), НФБК является 3НФ.

Отношение находится в НФБК, когда каждая нетривиальная и неприводимая слева функциональная зависимость обладает потенциальным ключом в качестве детерминанта.

1.4.5 Четвертая нормальная форма

Отношение находится в 4НФ, если оно находится в НФБК и все нетривиальные многозначные зависимости фактически являются функциональными зависимостями от ее потенциальных ключей.

В отношении R (A, B, C) существует многозначная зависимость R.A -> -> R.B в том и только в том случае, если множество значений B, соответствующее паре значений A и C, зависит только от A и не зависит от C.

Предположим, что рестораны производят разные виды пиццы, а службы доставки ресторанов работают только в определенных районах города. Составной первичный ключ соответствующей переменной отношения включает три атрибута: {Ресторан, Вид пиццы, Район доставки}.

Такая переменная отношения не соответствует 4НФ, так как существует следующая многозначная зависимость:

{Ресторан} → {Вид пиццы}

{Ресторан} → {Район доставки}

То есть, например, при добавлении нового вида пиццы придется внести по одному новому кортежу для каждого района доставки. Возможна логическая аномалия, при которой определенному виду пиццы будут соответствовать лишь некоторые районы доставки из обслуживаемых рестораном районов.

Для предотвращения аномалии нужно декомпозировать отношение, разместив независимые факты в разных отношениях. В данном примере следует выполнить декомпозицию на {Ресторан, Вид пиццы} и {Ресторан, Район доставки}.

Однако, если к исходной переменной отношения добавить атрибут, функционально зависящий от потенциального ключа, например цену с учётом стоимости доставки ({Ресторан, Вид пиццы, Район доставки} → Цена), то полученное отношение будет находиться в 4НФ и его уже нельзя подвергнуть декомпозиции без потерь.

1.5. Ограничение целостности баз данных

Ограничение целостности - это некоторое утверждение, которое может быть истинным или ложным в зависимости от состояния базы данных.

База данных находится в согласованном (целостном) состоянии, если выполнены (удовлетворены) все ограничения целостности, определенные для базы данных.

Любое ограничение целостности является семантическим понятием, т.е. появляется как следствие определенных свойств объектов предметной области и/или их взаимосвязей.

Вместе с понятием целостности базы данных возникает понятие реакции системы на попытку нарушения целостности.

2. ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1 Описание предметной области

Основная часть

Вы являетесь сотрудником коммерческого отдела компании, продающей различные товары (одежду) через Интернет. Вашей задачей является отслеживание финансовой составляющей ее работы. Работа компании организована следующим образом: на Интернет сайте представлены (выставлены на продажу) некоторые товары. Каждый из них имеет некоторое название, цену и количество, доступное к продаже. Для проведения исследований и оптимизации работы магазина вы пытаетесь собирать данные с клиентов. При этом для вас определяющее значение имеют стандартные анкетные данные, а также телефон и адрес электронной почты для связи. По каждому факту продажи вы автоматически фиксируете клиента, товары, количество, дату продажи.

Развитие постановка задачи

В результате эксплуатации базы данных выяснилось, что иногда возникают проблемы, связанные с нехваткой информации о наличии нужных товаров на складе в нужном количестве. Кроме того, обычно клиенты в рамках одного заказа покупают не один вид товара, а несколько видов.

2.2 Создание диаграммы «сущность-связь»

Для создания ER-модели использовался pgModeler — инструмент для проектирования баз данных.

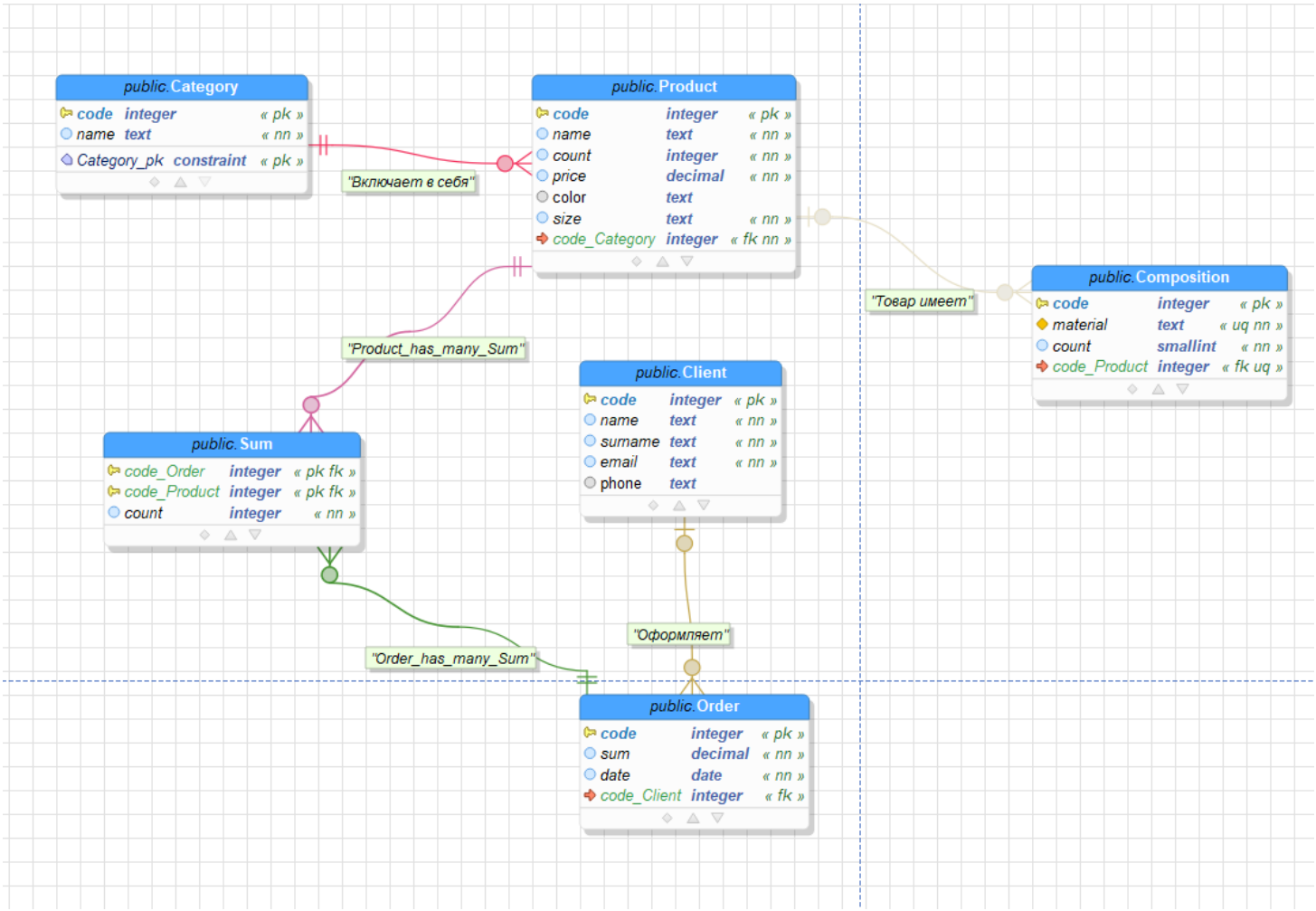

Рисунок 2.1 – ER-модель

Были выявлены следующие сущности:

1. Category – категория товаров. Включает в себя следующие атрибуты:

   code – код категории, тип: integer

   name – название категории, тип: text

2. Product – товар. Включает в себя следующие атрибуты:

   code – код товара, тип: integer

name – название товара, тип: text

count – количество товара на складе, тип: integer

price – цена товара, тип: decimal

color – цвет товара, тип: text

size – размер товара, тип: text

3. Composition – состав одежды. Включает в себя следующие атрибуты:

   code – код состава, тип: integer

   material – материал одежды (хлопок, полиэстер и т.д.), тип: text

   count – количество материала, в %, тип: smallint

4. Client – покупатель/клиент. Включает в себя следующие атрибуты:

   code – id клиента, тип: integer

   name – имя клиента, тип: text

   surname – фамилия клиента, тип: text

   email – электронная почта клиента, тип: text

   phone – номер телефона клиента, тип: text

5. Order – заказ клиента. Включает в себя следующие атрибуты:

   code – код заказа, тип: integer

   sum – общая сумма заказа, тип: decimal

   date – дата заказа, тип: date

6. Sum - корзина с товарами в определенном количестве и содержащихся в различных заказах

   code_Order – код заказа, является внешним ключом, тип: integer

   code_Client – код клиента, является внешним ключом, тип: integer

Связи между сущностями:

1. Category - Product: отношение ***один-ко-многим***, одна категория включает в себя множество типов товаров

2. Product – Composition: отношение ***один-ко-многим***, один товар может иметь несколько составов (например, толстовка имеет 50% хлопок, 50% полиэстер)
3. Client – Order: отношение ***один-ко-многим,*** один клиент может оформлять множество заказов
4. Product – Sum: отношение ***один-ко-многим***, один товаро может участвовать во множестве заказах
5. Order – Sum: отношение ***один-ко-многим***, один заказ может включать в себя множество товаров

2.3 Нормализация отношений. Приведение таблиц к нормальной форме

На основе анализа предметной области определим таблицу «Интернет-магазин» и заполним ее данными исходя из выделенных сущностей:

A	B	C	D	E	F	G	H
Код категории	Название категории	Код товара	Название товара	Кол-во товара	Цена	Цвет	Размер
1	Толстовки	111	Худи "Базовое"	100	1000	Белый	XS
1	Толстовки	222	Худи "Базовое"	100	1000	Зеленый	XM
2	Рубашки	333	Рубашка классическая	200	2000	Фиолетовая	XM
3	Штаны	444	Джинсы скинни	300	4000	Асфальт	XM
1	Толстовки	555	Худи "Мне неприятно"	100	1000	Черный	XS
1	Толстовки	222	Худи "Базовое"	100	1000	Зеленый	XM
1	Толстовки	111	Худи "Базовое"	100	1000	Белый	XS

(продолжение таблицы)

I	J	K	L	M	N	O	P
Код состава	Материал	Кол-во	Код клиента	Имя	Фамилия	E-mail	Телефон
1	Хлопок	60	123	Иван	Иванов	1@gmail.ru	111222
2	Полиэстер	40	123	Иван	Иванов	1@gmail.ru	111222
3	Хлопок	100	321	Антон	Иванов	222@gmail.ru	111222
2	Полиэстер	40	123	Иван	Иванов	1@gmail.ru	111222
1	Хлопок	60	NULL	NULL	NULL	NULL	NULL
2	Полиэстер	40	123	Иван	Иванов	1@gmail.ru	111222
1	Хлопок	60	123	Иван	Иванов	1@gmail.ru	111222

(продолжение таблицы)

Q	R	S	T
ID_Заказ	Сумма	Дата	Кол-во выбранного товара
234	2000	25.10.2020	2
234	2000	25.10.2020	2
555	6000	25.10.2020	3
666	4000	25.10.2020	1
NULL	NULL	NULL	NULL
334	2000	25.10.2020	2
334	2000	25.10.2020	2

Выявим первичный ключ таблицы, по которому можно будет однозначно определить все данные. В его состав войдут следующие столбцы: C, Q и I. Соответственно первичным ключ будет составным.

Исходя из представленных в таблице данных, возникают следующие аномалии:

— **Insert**: Невозможно добавить новый товар, который не будет куплен, соответственно пустое значение в Q, которое входит в состав первичного ключа

— **Delete**: При удалении категории удаляются товары и клиенты

— **Udpate**: При изменении названия товара требуется большое кол-во операции Update

Изобразим таблицу в виде схемы:

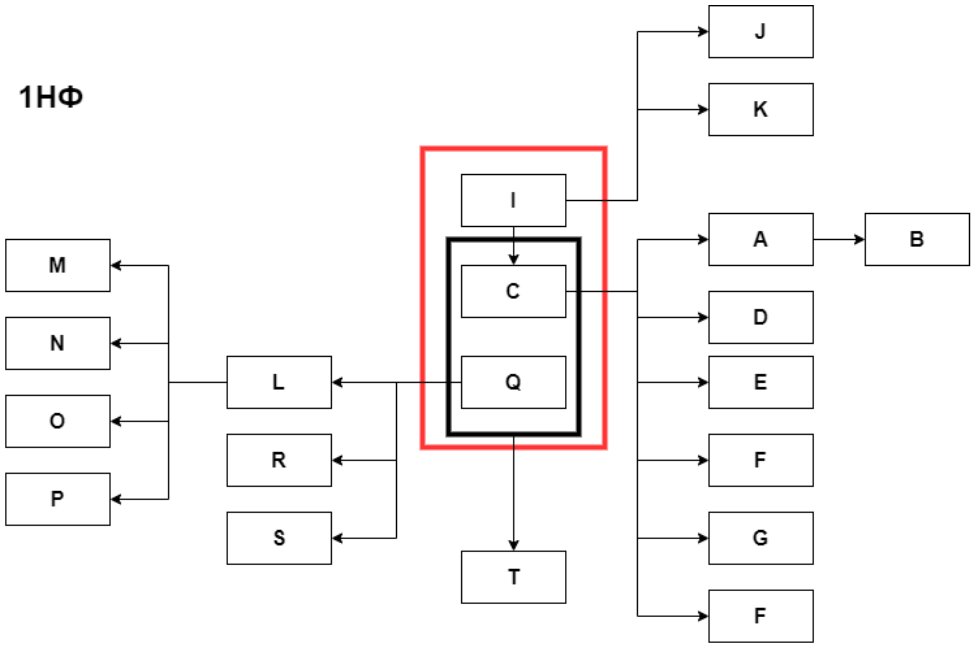

Рисунок 2.2 – Отношение в 1НФ

Отношение находится в 1НФ, так как каждый кортеж имеет только одно значение для каждого из атрибутов.

Отношение не находится во 2НФ, потому что присутствуют атрибуты, которые зависят от части первичного ключа. Для перехода ко 2НФ необходимо произвести декомпозицию.

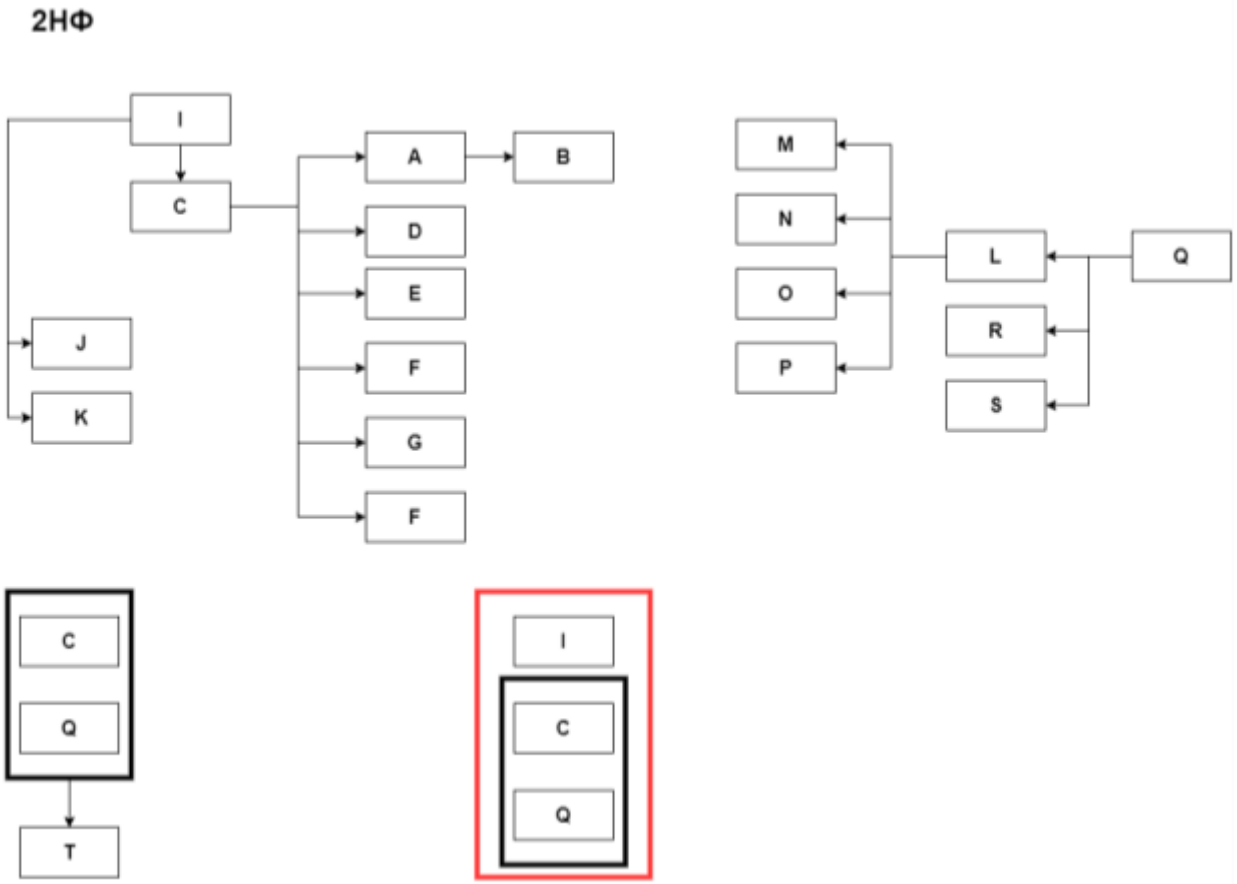

Рисунок 2.3 – Отношение во 2НФ

Далее необходимо привести таблицу к третьей нормальной форме. Для этого таблица должна находиться во второй нормальной форме, и не должно быть зависимостей одних не ключевых атрибутов от других, и все атрибуты должны зависеть только от первичного ключа. Таблица находится во второй нормальной форме, но в ней присутствуют транзитивные зависимости между не ключевыми атрибутами. Поэтому нужно все транзитивные зависимости вынести в отдельные отношения, в которых детерминанты становятся первичными ключами.

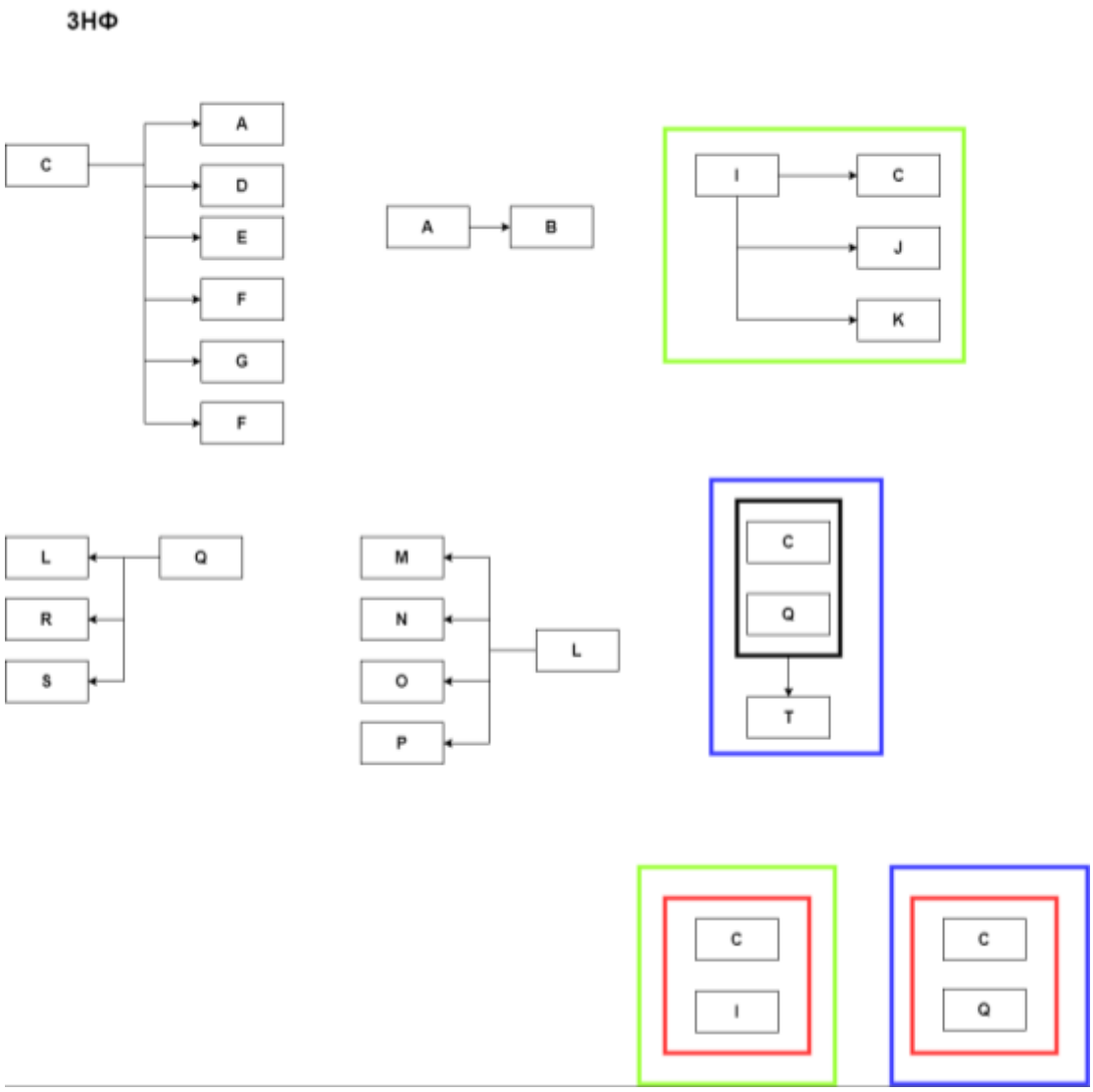

Рисунок 2.4 – Отношение в 3НФ

После исключения всех транзитивных зависимостей таблица «Интернет-магазин» находится в 3НФ, но не находится в 4НФ, потому что между C, Q, I существует многозначная зависимость. Товар (C) участвует независимо в двух процессах:

1. Учет состава (I)
2. Учет договоров (Q)

Чтобы избавиться от многозначной зависимости необходимо разбить отношение C, Q, I на C, I и C, Q (см. рисунок 2.4). Также после разбиения эти две

таблицы можно исключить, поскольку атрибуты, входящие в них, уже встречаются в других таблицах, полученных выше при нормализации. Синим и зеленым цветами на рисунке 2.4 выделены эквивалентные таблицы.

Таким образом, было получено 6 таблиц, которые соответствуют той ER-модели, полученной в разделе 2.2.

2.4 Описание декларативных ограничений целостности

Таблица	Поле(я)	Тип ограничения	Комментарий	SQL-код
Категория	Код, название	NOT NULL	Код, название должны быть не пустыми	code integer NOT NULL, name text NOT NULL
Категория	Код	PRIMARY KEY	Код должен быть уникальным	CONSTRAINT "Category_pk" PRIMARY KEY (code)
Товар	Код, название, количество, цена, размер, код категории	NOT NULL	Код, название, количество, цена, размер должны быть не пустыми	code integer NOT NULL, name text NOT NULL, count integer NOT NULL, price decimal NOT NULL, size text NOT NULL, "code_Category" integer NOT NULL,
Товар	Код	PRIMARY KEY	Код должен быть уникальным	CONSTRAINT "Product_pk" PRIMARY KEY (code)
Товар	Цвет	-	Цвет может иметь пустое значение	color text

Состав	Код, материал, количество	NOT NULL	Код, материал, количество, должны быть не пустыми	code integer NOT NULL, material text NOT NULL, count smallint NOT NULL
Состав	Код	PRIMARY KEY	Код должен быть уникальным	CONSTRAINT "Composition_pk" PRIMARY KEY (code)
Состав	Материал, код товара	UNIQUE KEY	Материал, код товара должны быть уникальными	CONSTRAINT material_uk UNIQUE ("code_Product", material)
Клиент	Код, название, имя, фамилия, почта	NOT NULL	Код, название, имя, фамилия, почта не должны быть пустыми	code integer NOT NULL, name text NOT NULL, surname text NOT NULL, email text NOT NULL
Клиент	Код	PRIMARY KEY	Код должен быть уникальным	CONSTRAINT "Client_pk" PRIMARY KEY (code)
Клиент	Телефон	-	Телефон может иметь пустое значение	phone text
Заказ	Код, сумма, дата,	NOT NULL	Код, сумма, дата, не должны быть пустыми	code integer NOT NULL, sum decimal NOT NULL, date date NOT NULL
Заказ	Код	PRIMARY KEY	Код должен быть уникальным	CONSTRAINT "Order_pk" PRIMARY KEY (code)

Таблица Sum	Код заказа, код товара, количество	NOT NULL	Код заказа, код товара, количество не должны быть пустыми	code_Order" integer NOT NULL, "code_Product" integer NOT NULL, count integer NOT NULL,
Таблица Sum	Код заказа, код товара	PRIMARY KEY	Код заказа, код товара должны быть уникальными	CONSTRAINT "Sum_pk" PRIMARY KEY ("code_Order","code_Product")

2.5 Реализация запросов к базе данных

1. Получить список клиентов с суммарной стоимостью приобретенных товаров свыше 10000 руб.

Необходимо выполнить соединение таблиц “Sum”, “Товар” и “Заказ” и сгруппировать их по коду клиента и сумме товаров с помощью агрегатной функции SUM.

```
SELECT t4."code_Client", SUM(t1."count" * t2."price") FROM "Sum" t1
            INNER JOIN "Product" t2 ON t1."code_Product"=t2."code"
                INNER JOIN "Order" t4 ON t1."code_Order"=t4."code"
                Group by t4."code_Client" having SUM(t1."count" * t2."price")>10000;
```

2. Получить список из 5 самых распространенных товаров

Необходимо в таблице “Sum” сгруппировать по коду товара, отсортировать по количеству товара и вывести 5 товаров.

```
SELECT t1."code_Product", SUM(t1."count") FROM "Sum"
t1
        GROUP BY t1."code_Product"
            Order by "sum" desc limit 5;
```

3. Прибыль интернет магазина на заданный диапазон дат

Для получения прибыли необходимо сделать выборку по дате из таблицы “Order”, затем обернуть все это в еще один SELECT, чтобы можно было просуммировать все значения. Также была написана функция, чтобы можно было в качестве параметров передавать в нее любые даты.

```
DROP FUNCTION profit(date,date);

CREATE OR REPLACE FUNCTION profit(date1 date,
date2 date)
  RETURNS TABLE(date_begin text, date_end text,
sum decimal)
  LANGUAGE 'plpgsql' AS $$
  BEGIN

  RETURN QUERY SELECT  date1::text, date2::text,
SUM(t2."sum") FROM
  (SELECT t1."date", SUM (t1."sum") FROM "Order"
t1
       WHERE date BETWEEN date1 AND date2
           Group by t1."date") AS t2;
  END;
  $$;
```

```
SELECT  * FROM profit('2020-10-26', '2020-11-25');
```

4. Получить текущее количество каждого товара на складе

Для получения текущего количества товара на складе нам нужно лишь выбрать код, наименование и количество из таблицы «Товар».

```
SELECT code,name,number FROM "Product"
```

5. Получить распределение числа заказов по заданным диапазонам цен

Для запроса нужно создать таблицу диапазонов цен и заполнить её.

```
CREATE TABLE "R"
(
    "code" integer PRIMARY KEY,
    "from" decimal,
    "before" decimal
);

INSERT INTO "R" VALUES
(1, 0, 100),
(2, 100, 500),
(3, 500, 1000),
(4, 1000, 1500),
(5, 1500, 2000),
(6, 2000, 2500),
(7, 2500, 3000),
(8, 3000, 4000);
```

Далее написать запрос, в котором будут браться соответствующие значения из таблицы R и группировать по коду товара и диапазону цен, в который он входит (from и before).

```
    SELECT t4."code", t4."from", t4."before",
COUNT(t4."count") FROM
        (SELECT t3."code", t3."from", t3."before",
t1."count", t1."code_Order" FROM "Sum" t1
            INNER JOIN "Product" t2 ON
t1."code_Product"=t2."code"
                INNER JOIN "R" t3 ON (t2."price" >=
t3."from" AND t2."price" <= t3."before")
                    Group by t3."code", t3."from",
t3."before", t1."count", t1."code_Order") AS t4
                        Group by t4."code", t4."from",
t4."before";
```

2.6 Описание процедурных ограничений целостности

Проанализировав данные и бизнес-правила предметной области, кроме декларативных ограничений целостности, описанных ранее, были выявлены следующие процедурные ограничения:

- Значения поля «count» из таблицы «Товар» должно быть больше значения поля «count» из таблицы «Sum», а также поле «count» из таблицы «Товар» должно автоматически изменяться по мере продажи товара;
- По мере добавления к заказу различных товаров в таблице “Sum”, необходимо рассчитывать сумму в таблице “Order” каждый раз, когда добавляется новый товар.

Поскольку нарушения могут возникнуть при вводе неправильных данных в таблицы, то для реализации ограничений следует использовать триггерные функции, которые вызываются триггерами. Для того, чтобы не нарушить указанные выше ограничения, достаточно будет написать одну триггерную функцию и создать для нее соответствующий триггер.

В триггерной функции Sum_Fn_CheckProductsInStock проверяется наличие нужного количества товара на складе, затем, если условие истинно, количество товара на складе обновляется и рассчитывается сумма для таблицы “Заказ” (цена * кол-во выбранного товара), прибавляя при этом предыдущую сумму:

```
CREATE FUNCTION "Sum_Fn_CheckProductsInStock"()
    RETURNS TRIGGER
    LANGUAGE 'plpgsql'
AS
$$
declare n_price integer; -- переменная, куда будет записана цена товара
BEGIN
    if exists(
        SELECT * FROM "Product" WHERE new."code_Product" = "Product"."code"
            AND ("Product"."count" - new."count") >= 0
    ) then
        UPDATE "Product" SET "count" = "Product"."count" - new."count"
            WHERE new."code_Product" = "Product"."code";
```

```
            n_price = (SELECT t1."price" FROM "Product"
t1
                       WHERE t1."code" =
new."code_Product");

            UPDATE "Order" SET "sum" = "sum" + (n_price
* new."count")
                    WHERE "code" = new."code_Order";


            return new;
        else
            return null;
        end if;
    END;
    $$;
```

Триггер Sum_Tr_CheckProductsInStock:

```
    CREATE TRIGGER "Sum_Tr_CheckProductsInStock"
        BEFORE INSERT OR UPDATE
        ON "Sum"
        FOR EACH ROW
        EXECUTE PROCEDURE
"Sum_Fn_CheckProductsInStock"();
```

2.7 Описание доступа к данным

Для того, чтобы начать пользоваться созданной базой данных, необходимо к ней подключиться. Я использовал методы языка программирования PHP, чтобы создать класс базы данных DB.

Метод pg_connect, представленный в коде ниже, открывает соединение с базой данных PostgreSQL, где необходимо в поле dbname указать название существующей базы данных, а также заполнить все остальные обязательные поля (user, password сервера и т.д.).

```
<?php
    class DB {
        var $dblogin = "postgres";
        var $dbpass = "gud623";
        var $db = "online_shop";
        var $dbhost="localhost";
        var $dbport=5432;

        var $link;

        function get_link() {
            return $this->link;
        }

        function connect() {
            $this->link = pg_connect("host=localhost port=5432
dbname=online_shop user=postgres password=gud623");
        }

        function close() {
            pg_close($this->link);
        }
```

```
    }

?>
```

Для отправки запросов в базу данных используется AJAX-технология, подход которой заключается в «фоновом» обмене данными браузера с веб-сервером. Например, загрузка всех товаров из базы данных представлена ниже.

```
    function load_products() {
        $.ajax({
            method: "GET",
            url: "functions_php/products/get_products.php",
        })
        .done(function (data) {
            data = $.parseJSON(data);
            var tableSize = $('#my-table-products tr').length;
            if (tableSize == 1) {
                var tbody = document.getElementById('my-table-
products').getElementsByTagName("TBODY")[0];
                for (i = 0; i < data.length; i++) {
                    var row = document.createElement("TR");
                    var td1 = document.createElement("TD");

td1.appendChild(document.createTextNode(data[i]['code']));
                    var td2 = document.createElement("TD");

td2.appendChild(document.createTextNode(data[i]['name']));
                    var td3 = document.createElement("TD");

td3.appendChild(document.createTextNode(data[i]['count']));
                    var td4 = document.createElement("TD");

td4.appendChild(document.createTextNode(data[i]['price']));
                    var td5 = document.createElement("TD");
```

```
td5.appendChild(document.createTextNode(data[i]['color']));
                    var td6 = document.createElement("TD");

td6.appendChild(document.createTextNode(data[i]['size']));
                    row.appendChild(td1);
                    var td7 = document.createElement("TD");

td7.appendChild(document.createTextNode(data[i]['code_Category']))
;
                    row.appendChild(td1);
                    row.appendChild(td2);
                    row.appendChild(td3);
                    row.appendChild(td4);
                    row.appendChild(td5);
                    row.appendChild(td6);
                    row.appendChild(td7);
                    tbody.appendChild(row);
                }
            } else {
                clear_table('my-table-products');
                load_products();
            }
        });
    }
```

Метод «$.ajax()» принимает в качестве параметра пачку настроек и URL цели (в моем случае файл, хранящий соответствующую функцию get_products). Ниже предоставляется ее код:

```
function get_products() {
        $db = new DB();
        $db->connect();
```

```
            $query = "SELECT * FROM \"Product\"";
            $result = pg_query($query) or die('Ошибка запроса: ' .
pg_last_error());

            $line = pg_fetch_all($result, PGSQL_ASSOC);
            $db->close();
            return $line;
        }
```

В строчке 2 и 3 создается подключение к базе данных, в $query помещается строка с запросом, pg_query выполняет запрос и возвращает ресурс результата запроса, который присваивается переменной $result. pg_fetch_all — выбирает все данные из результата запроса и помещает их в массив, соответственно переменная $line содержит массив со всеми записями из результата запроса.

Ниже предоставляется скриншот результата работы метода load_products() после нажатия на кнопку «Загрузить продукты».

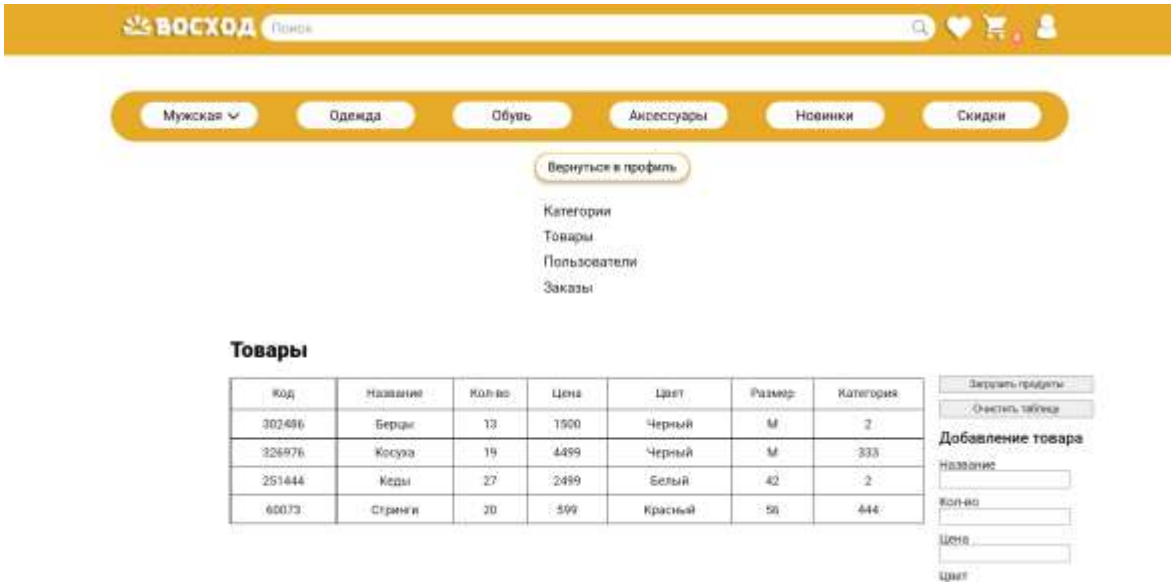

Рисунок 2.5 Вывод товаров из БД на страницу в браузере

Код метода вставки товара представлен ниже:

```
    function insert_product($tittle, $count, $price, $color, $size,
$code_category) {
```

```
            $db = new DB();
            $db->connect();

            do  {
                $code = generate_id();
                $check  =  "SELECT  *  FROM  \"Product\"  WHERE
\"code\"='$code'";
                $result = pg_query($db->link,$check);
                $num_rows = pg_num_rows($result);
            }  while ($num_rows >= 1);

            $insert_query = "INSERT INTO \"Product\" VALUES
            ('$code',  '$tittle',  '$count',  '$price',  '$color',
'$size', '$code_category')";
            $result = pg_query($db->link,$insert_query);

            $db->close();
        }
```

Результат выполнения данного метода:

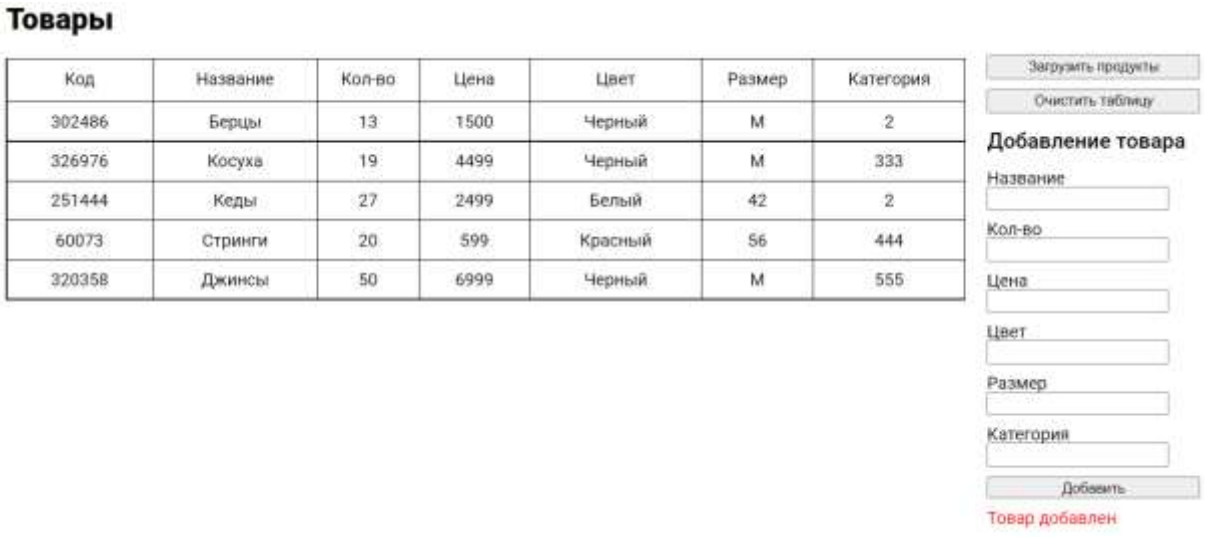

Рисунок 2.6 – Вставка нового товара в БД и отображение его на странице браузера

ЗАКЛЮЧЕНИЕ

Разработанная база данных «Интернет-магазин одежды» позволяет быстро и эффективно работать с данными этой предметной области. Данная БД является учебной и не охватывает всю бизнес-логику. Однако является прототипом, демонстрирующим работу в данной отрасли. Данная БД может быть расширена для автоматизации нерассмотренных в рамках данного курсового проекта концепций в предметной области «Интернет-магазин одежды».

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Хернандес М., Вьескас Д. SQL-запросы. Практическое руководство.: Пер. с англ. – М.: Лори, 2003. – 473 с.: ил.

2. Гончаров А. Ю. язык SQL. Самоучитель с примерами., Москва, 2011г.

3. Мейер М. Теория реляционных баз данных. – М.: Мир, 2010.

4. http://www.mstu.edu.ru/study/materials/zelenkov/toc.html