|  |  |
| --- | --- |
| **Российский университет транспорта (МИИТ)**  **Институт транспортной техники и систем управления**  **Кафедра «Управление и защита информации»** | |
| **Отчет**  **по курсовому проекту**  **по теме «Разработка БД для компании по продаже недвижимости»**  **по дисциплине «Основы построения защищенных баз данных»** | |
|  | Выполнил:  Студент группы ТКИ-442  Зинченко Б.А.  Проверил:  Доцент кафедры УиЗи, к.т.н.  Васильева М.А. |
| Москва 2024 | |

Оглавление

[ЦЕЛЬ КУРСОВОГО ПРОЕКТА 3](#_Toc167302293)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 3](#_Toc167302294)

[2. АНАЛИЗ ИНФОРМАЦИОННЫХ ЗАДАЧ И КРУГА ПОЛЬЗОВАТЕЛЕЙ СИСТЕМЫ 4](#_Toc167302295)

[2.1 Функциональные возможности 4](#_Toc167302296)

[3. ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БД 5](#_Toc167302297)

[3.1 Преобразование ER-диаграммы в схему базы данных 5](#_Toc167302298)

[3.2 Составление реляционных отношений 6](#_Toc167302299)

[3.3 Определение дополнительных ограничений 8](#_Toc167302300)

[3.4 Нормализация полученных отношений 9](#_Toc167302301)

[4. ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БД 9](#_Toc167302302)

[4.1 Разработка скриптов для создания базы данных и таблиц 9](#_Toc167302303)

[4.1.1 Ограничения для таблицы «Объекты» 11](#_Toc167302304)

[4.1.2 Ограничения для таблицы «Товары» 12](#_Toc167302305)

[4.2 Диаграмма базы данных 12](#_Toc167302306)

[4.3 Разработка скриптов для добавления данных в таблицы 13](#_Toc167302307)

[4.4 Разработка необходимых представлений (view) 15](#_Toc167302308)

[4.5 Разработка необходимых функций и процедур 15](#_Toc167302309)

[4.6 Разработка необходимых триггеров 20](#_Toc167302310)

[ЗАКЛЮЧЕНИЕ 20](#_Toc167302311)

# ЦЕЛЬ КУРСОВОГО ПРОЕКТА

Целью курсового проекта является изучение методов и закрепление знаний в проектировании реляционных баз данных (РБД) в системе управления базами данных (СУБД) PostgreSQL.

# 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В данном курсовом проекте для проектирования реляционной базы данных был выбран магазин, занимающийся продажей оргтехниики.

БД создается для информационного обслуживания сотрудников магазина. Магазин реализует товар. Постоянным клиентам и покупателям, набравшим товара на сумму х руб., предоставляется скидка. Предметная область данной системы включает в себя следующие особенности:

1. Показывать ассортимент выбранного товара и его цену;
2. Находить товар по названию (по фирме изготовителя)..
3. Рассчитывать стоимость покупки, учитывая скидку.
4. Показывать количество (стоимость) проданного товара (по выбранному товару, по магазину в целом) за отчетный период.

Выделим базовые сущности предметной области:

1. Объект содержит в себе идентификатор объекта, тип объекта – в данном случае под типом понимается ассортимент товара, его цена и изготовитель.
2. Сущность товар состоит из идентификатора товара и его названия;
3. Клиент – то есть это может быть покупатель или арендатор. Содержит в себе идентификатор клиента и имя клиента;
4. Общая сущность, связывающая все вышеизложенные, называется   
   «О объектах» и содержит в себе идентификатор объекта, идентификатор производителя, идентификатор клиента и количество проданных единиц товара с их стоимостью – то есть покупка единиц оргтехники.

ER-диаграмма БД, выполненная в графическом редакторе *MS Visio*, приведена далее (Рисунок 1). Данная БД содержит связи «один-к-одному».

|  |
| --- |
|  |
| 1. – ER-диаграмма |

# 2 АНАЛИЗ ИНФОРМАЦИОННЫХ ЗАДАЧ И КРУГА ПОЛЬЗОВАТЕЛЕЙ СИСТЕМЫ

Система создается для обслуживания групп пользователей, к которым относятся продавцы (реализовать функцию формирования суммы выкупа за квартал) и администрация магазина.

## 2.1 Функциональные возможности

Система в компании по продаже оргтехники имеет функциональную возможность ведения БД (запись, чтение, модификация и удаление данных), обеспечения логической непротиворечивости БД, а также обеспечения защиты данных от несанкционированного или случайного доступа – это значит, что в базе данных определены права на доступ к информации.

# 3 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БД

## 3.1 Преобразование ER-диаграммы в схему базы данных

База данных создаётся на основании схемы базы данных. Преобразование ER-диаграммы в схему БД выполняется путем сопоставления каждой сущности и каждой связи, имеющей атрибуты, отношения   
(таблицы БД).

На схеме, представленной ниже (Рисунок 2), введено используемое обозначение:



|  |
| --- |
|  |
| 1. – Обозначение на схеме базы данных |

Полученная схема реляционной базы данных (далее, РБД) компании по продаже недвижимости приведена ниже (Рисунок 3).

|  |
| --- |
|  |
| 1. – Схема РБД, полученная из ER-диаграммы компании |

Давайте рассмотрим типы связей между таблицами:

1. **Товары и Покупки**:
   * Один товар может быть куплен множеством клиентов, поэтому это тип связи "Один ко многим" (One-to-Many): один товар может быть связан с множеством записей о покупках в таблице "Покупки".
2. **Клиенты и Покупки**:
   * Каждая покупка относится к одному клиенту, но один и тот же клиент может совершать несколько покупок, так что это также тип связи "Один ко многим" (One-to-Many): один клиент может быть связан с множеством записей о покупках в таблице "Покупки".

## 3.2 Составление реляционных отношений

Одно реляционное отношение соответствует одной сущности (объекту предметной области) и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи. В том случае, если базовое отношение не имеет потенциальных ключей, вводится суррогатный первичный ключ, который не несёт смысловой нагрузки и служит только для идентификации записей.

В данной системе сущности «Объекты», «Покупатели» , «Покупки» и «Скидки» имеют атрибут «id\_object», «id\_client» и «id\_realtor» соответственно с целью компактного сбора в общей сущности - то есть можно назвать такие атрибуты суррогатными первичными ключами.

Отношения приведены ниже (см. Таблица 1, 2, 3). Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной.

Таблица 1-Таблица отношения Товары(Objects).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Товары | Идентификатор объекта | ID | Целое число | Первичный ключ |
| Наименование товара | name\_object | Строка (30) | Обязательное поле |
| Изготовитель | manufacturer | Строка (30) | Обязательное поле |
| Цена | price | Целое число | Обязательное поле, целое число не может быть отрицательным |

Таблица 2-Таблица отношения Покупатели(Buyers).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Покупатели | Идентификатор покупателя | ID | Целое число | Первичный ключ |
| Имя покупателя | buyer\_name | Строка (30) | Обязательное, уникальное |

Таблица 3-Таблица отношения Покупки(Purchases).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Покупки | Идентификатор покупки | ID | Целое число | Первичный ключ |
| Идентификатор объекта | id\_object | Целое число | Внешний ключ к отношению (сущности) товары |
| Идентификатор покупателя | id\_buyer | Целое число | Внешний ключ к отношению (сущности) покупатели |
| Количество купленного товара | total | Целое число | Обязательное поле, целое число |
| Дата покупки | purchase\_date | Дата | Обязательное поле |
| Общ сумма покупки товара | total | int | Целое число |
| Количество купленного товара | quantity | Int | Целое число |

## Определение дополнительных ограничений

Необходимо отметить ограничения кроме тех, которые указаны в Таблицах 1-3.

1. **Ограничения целостности ссылочной целостности**: эти ограничения гарантируют, что значения внешних ключей всегда ссылается на существующие значения в связанных таблицах. Например, в таблице "Покупки" поле ID\_товара (FK) не должно принимать значения, которых нет в таблице "Товары".
2. **Ограничения целостности сущности**: они гарантируют, что каждая запись в таблице имеет уникальное значение для определенного поля или комбинации полей. Например, в таблице "Товары" может быть установлено ограничение целостности, которое не позволит добавить два товара с одинаковым названием.
3. **Ограничения целостности проверки**: они позволяют определить допустимые диапазоны значений для определенных полей. Например, в таблице "Покупки" можно установить ограничение, чтобы количество товаров не могло быть отрицательным.
4. **Ограничения на обновление и удаление**: можно установить правила, которые определяют, что происходит при обновлении или удалении записи, связанной с другими записями. Например, при удалении клиента можно определить правило, что все его покупки также удаляются из базы данных.
5. **Ограничения на нулевые значения**: гарантируют, что определенные поля не могут содержать нулевые значения. Например, в поле "Цена" таблицы "Товары" можно запретить нулевые значения.

## 3.4 Нормализация полученных отношений

Реляционная база данных (РБД) часто нуждается в нормализации для повышения её эффективности, уменьшения избыточности данных и предотвращения аномалий при обновлении, вставке или удалении данных. Нормализация до третьей нормальной формы (3NF) является важным этапом, который обеспечивает соблюдение определённых правил для достижения этих целей. Рассмотрим пример базы данных, состоящей из таблиц "Покупатели", "Товары" и "Покупки".

Нормализация полученных отношений после разделения таблиц "Покупатели", "Товары" и "Покупки" помогает еще лучше понять, как соблюдение нормальных форм (NF) улучшает структуру базы данных.

**Применение нормализации на примере**

**1. Первая нормальная форма (1NF)**

Таблицы, представленные выше, уже находятся в первой нормальной форме, так как:

* Все атрибуты атомарны.
* Каждая строка уникальна.
* Таблицы не содержат повторяющихся групп.

**2. Вторая нормальная форма (2NF)**

Все три таблицы также находятся во второй нормальной форме, так как:

* Находятся в первой нормальной форме.
* Все неключевые атрибуты зависят от всего первичного ключа.

Для таблиц "Покупатели" и "Товары" это несложно, так как у них простой первичный ключ. Для таблицы "Покупки" это также выполнено, так как каждая строка зависит от уникальной комбинации **ID\_покупателя** и **ID\_товара**.

**3. Третья нормальная форма (3NF)**

Все три таблицы находятся в третьей нормальной форме, так как:

* Находятся во второй нормальной форме.
* Все неключевые атрибуты зависят только от первичного ключа и не имеют транзитивных зависимостей.

# 4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БД

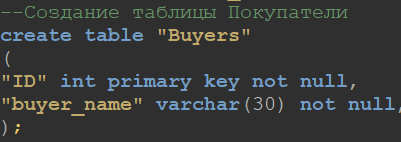
## 4.1 Разработка скриптов для создания базы данных и таблиц

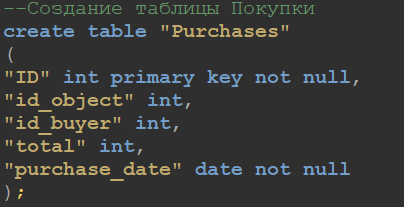
Первоначально создаётся база данных «Компания по недвижимости»:

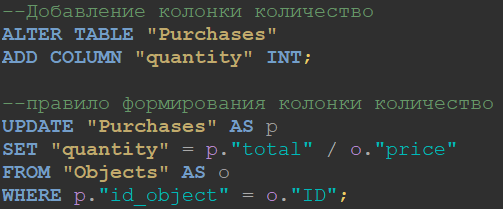


1. – Скрипт создания БД «pc\_shop»

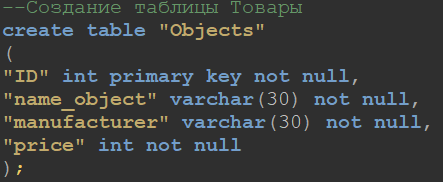
Далее, были созданы таблицы «Товары», «Покупки» и «Покупатели»:



1. – Скрипт создания таблицы «Покупатели»
2. – – Скрипт создания таблицы «Покупки»

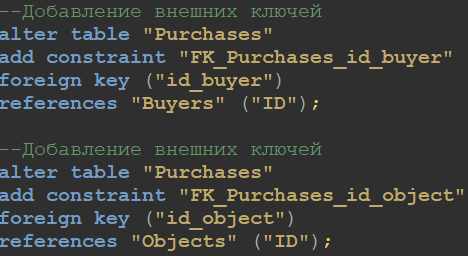


1. – Скрипт добавления колонки количество для таблицы «Покупки»



1. Скрипт создания таблицы «Товары»

Добавление в созданные таблицы внешних ключей:

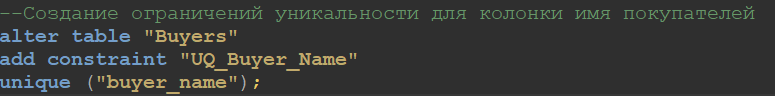


1. Скрипт добавления внешних ключей для «Покупки»

На следующем этапе вносятся ограничения для таблиц.

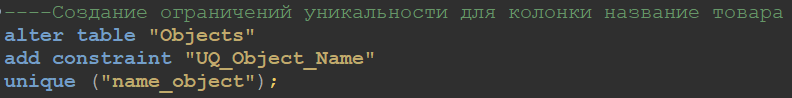
### 4.1.1 Ограничения для таблицы «Объекты»

В таблице «Покупатели» добавлено ограничение на уникальность поля name\_buyer :



1. Скрипт создания ограничений для «Покупатели

### 4.1.2 Ограничения для таблицы «Товары»

В таблице добавлено ограничение на обязательное заполнение поля «name\_object»: 

1. Скрипт создания ограничений для «Товары»

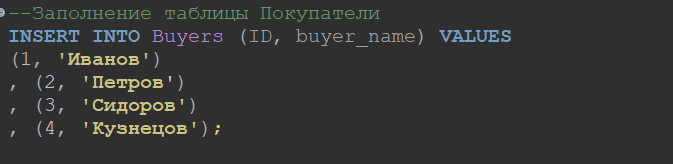
## 4.2 Диаграмма базы данных

Результатом разработки скриптов является диаграмма базы данных, представленная ниже (Рисунок 12).

|  |
| --- |
|  |
| 1. – Диаграмма базы данных |

## 4.3 Разработка скриптов для добавления данных в таблицы

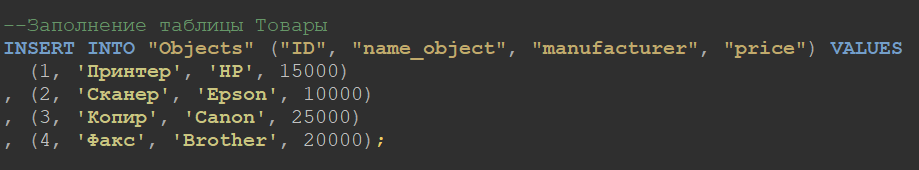
Заполнение таблицы «Покупатели»:



1. Скрипт заполнения таблицы «Покупатели»

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Buyers» |

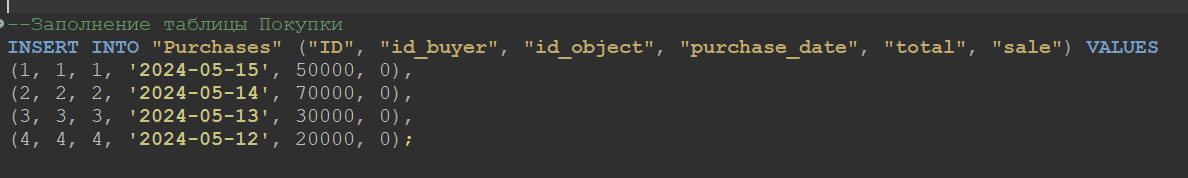
Заполнение таблицы «Товары»:



1. Скрипт заполнения таблицы «Товары»

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Objects» |

Заполнение таблицы «Покупки»:



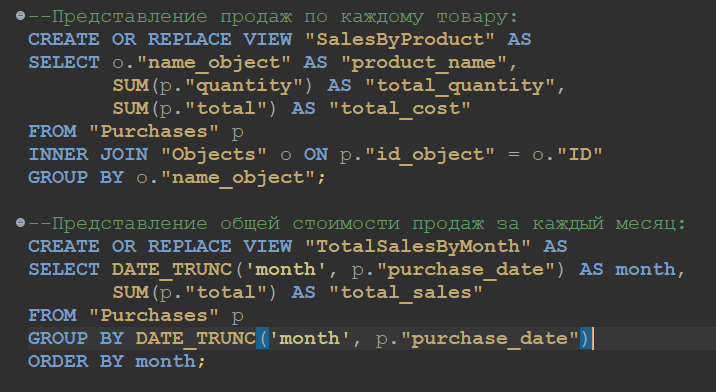
1. Скрипт заполнения таблицы «Покупки"

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Purchases» |
|  |

Заполнение таблицы «О объектах» реализуется с помощью процедуры, которая будет представлена далее (Раздел 4.6).

## 4.4 Разработка необходимых представлений (view)

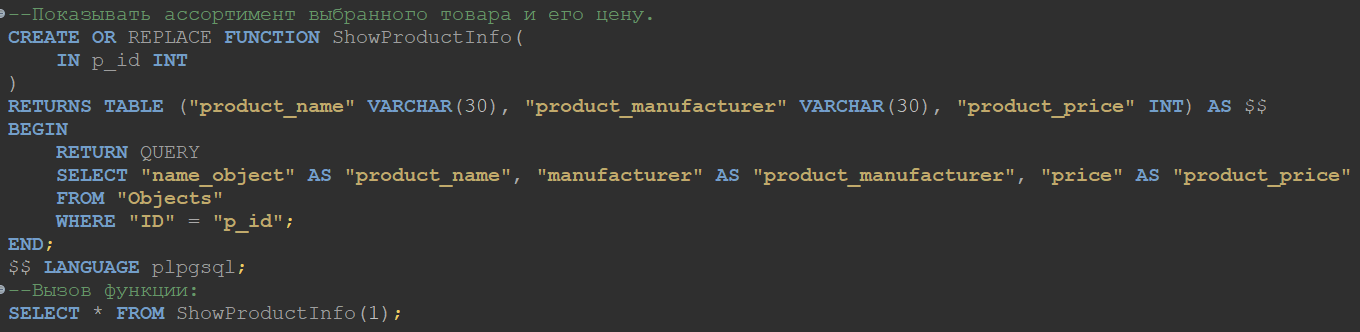
Представления могут быть полезны для удобного доступа к данным. На примере данной базы данных можно создать несколько представлений:



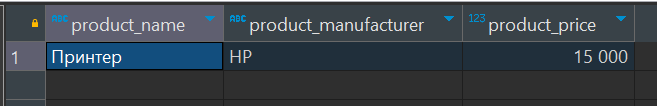
1. Представления в данной БД

## 4.5 Разработка необходимых функций и процедур

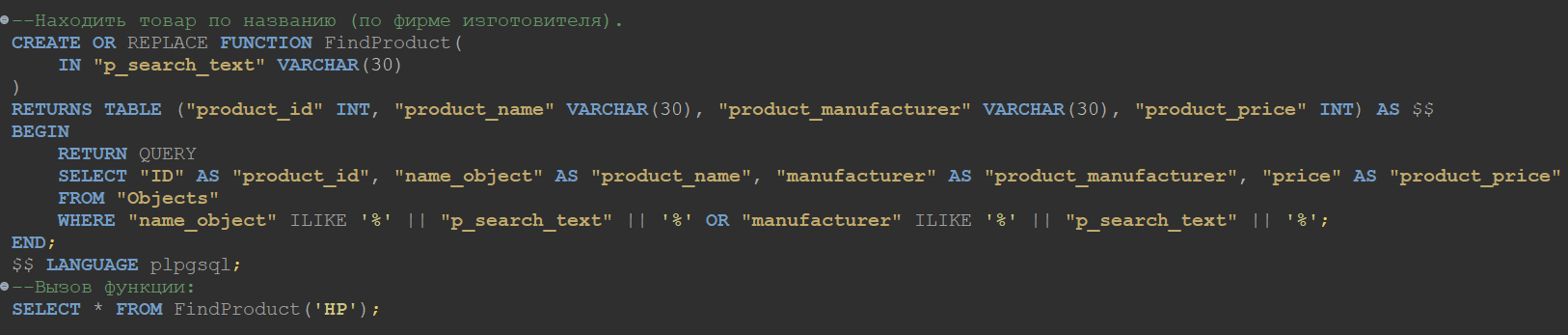
В данной базе данных был реализован следующий набор функций и процедур:

--Показывать ассортимент выбранного товара и его цену.

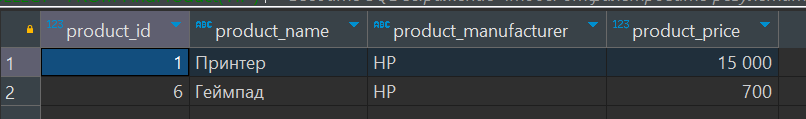
1. Скрипт ShowProductInfo



1. Результат выполнения скрипта ShowProductInfo

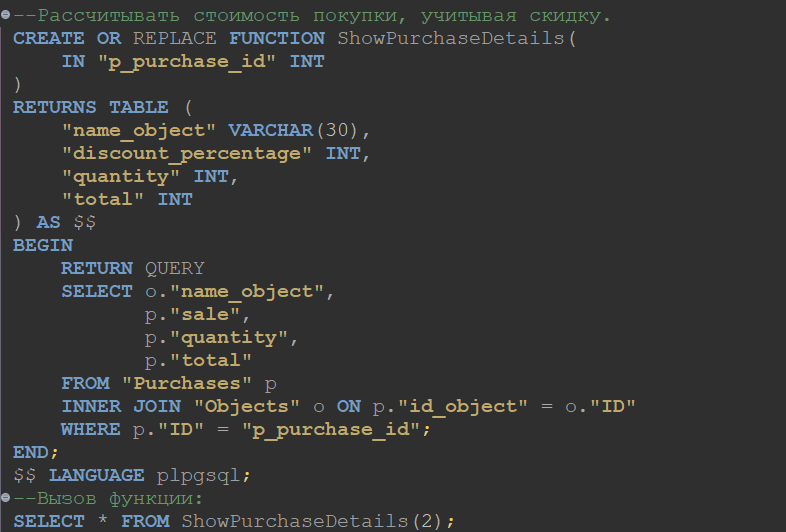
--Находить товар по названию (по фирме изготовителя). 

1. Скрипт FindProduct

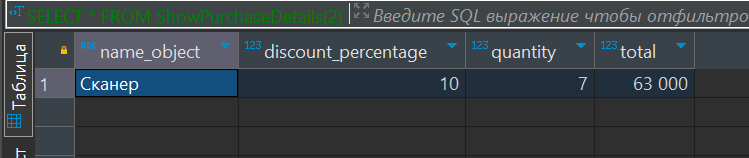


1. Результат выполнения скрипта FindProduct

--Рассчитывать стоимость покупки, учитывая скидку.

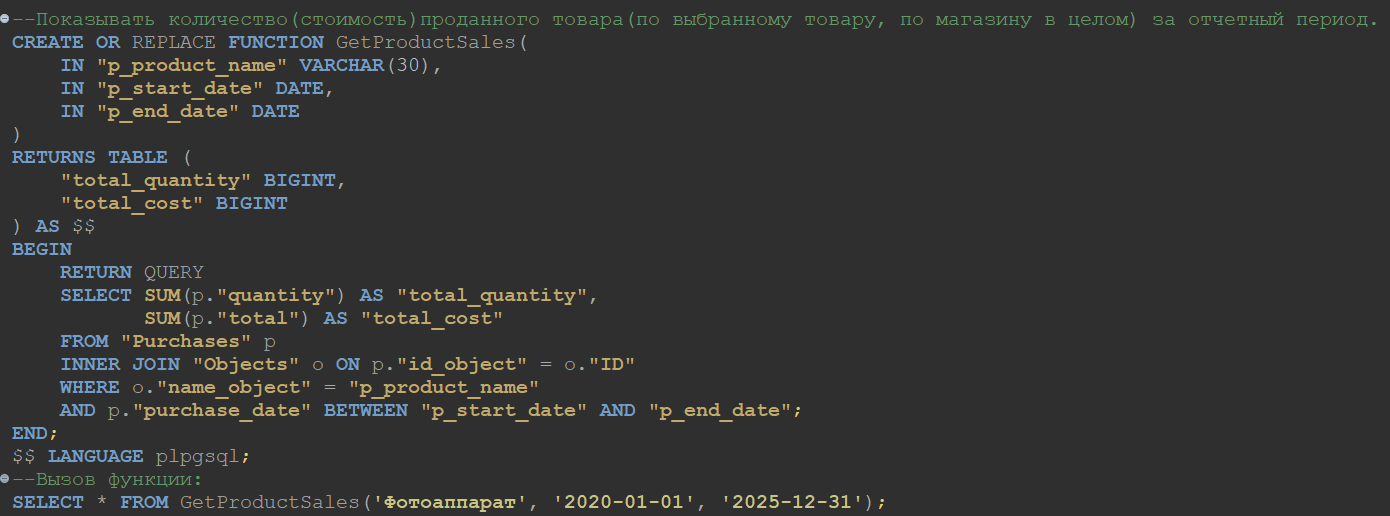


1. Скрипт ShowPurchaseDetail

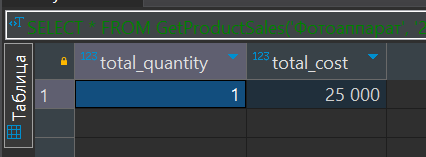


1. Результат выполнения скрипта ShowPurchaseDetail

--Показывать количество(стоимость)проданного товара(по выбранному товару, по магазину в целом) за отчетный период.



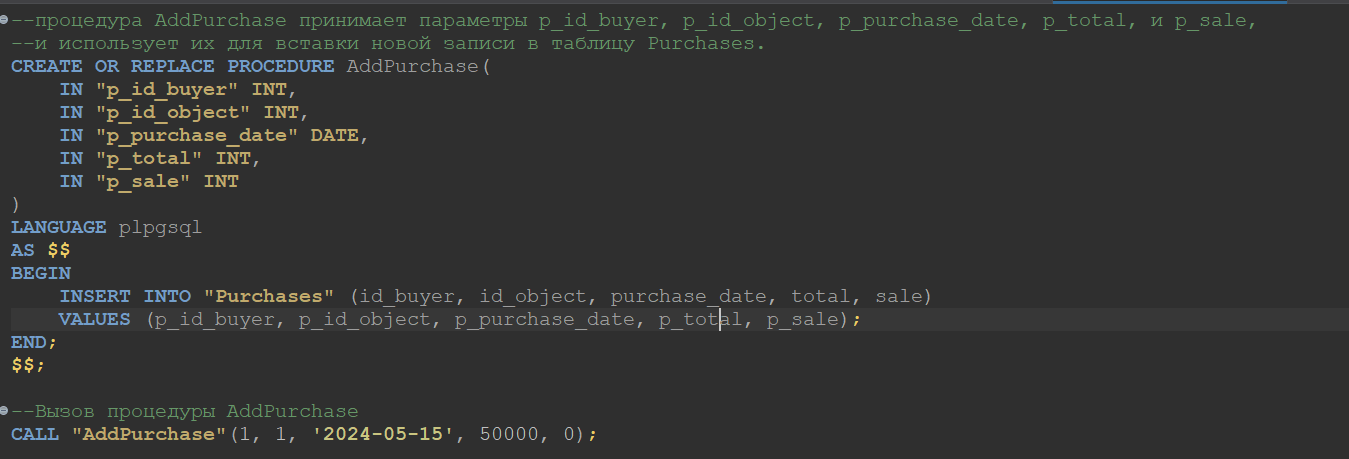
1. Скрипт GetProductSales



1. Результат выполнения скрипта GetProductSales

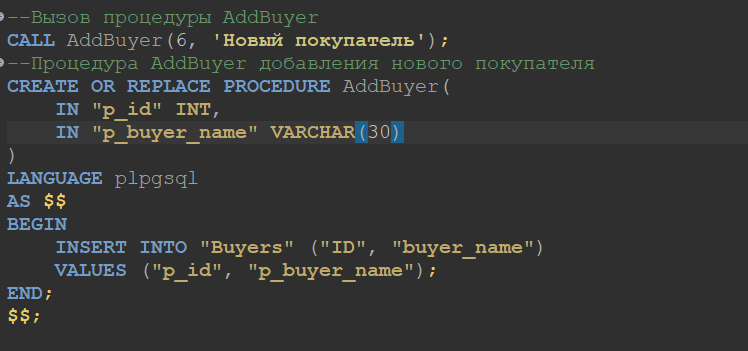
Также в данной работе реализованы следующие процедуры:

Процедура добавления элемента в таблицу покупки:



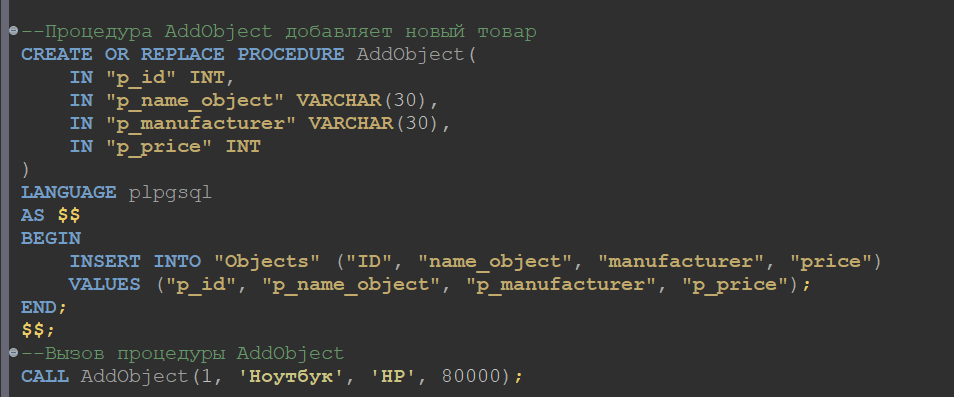
1. Скрипт процедуры AddPurchase

Процедура процедуры добавления нового покупателя:



1. Скрипт процедуры AddBuyer

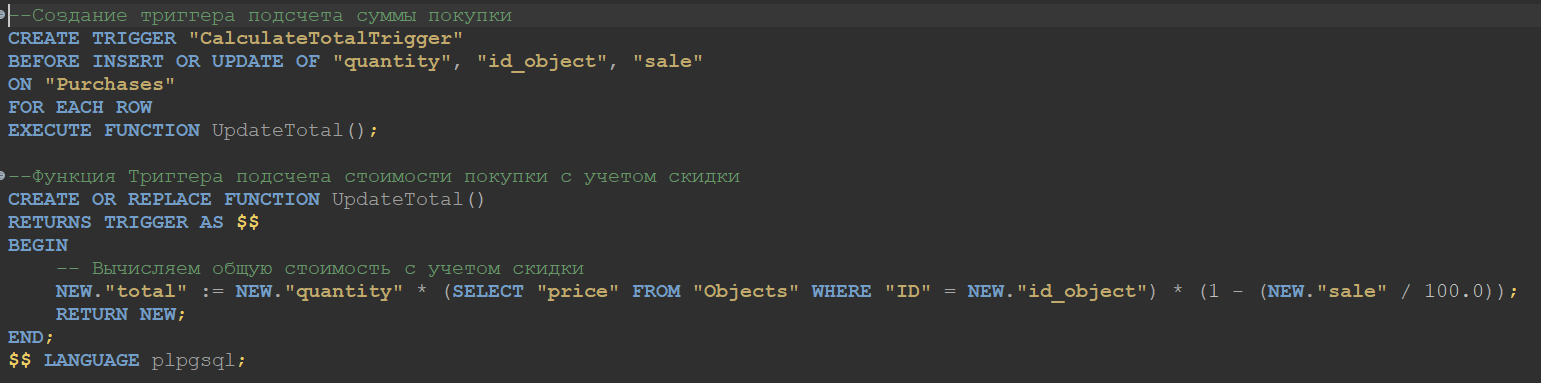
Процедура добавления нового товара:



1. Скрипт процедуры AddObject

## 4.6 Разработка необходимых триггеров

В данной работе необходимый триггер заключается в том, чтобы автоматически формировать стоимость покупки(total) в зависимости от цены товара(price), размера скидки на данный товар(sale) и количества единиц купленного товара(quantity). Данный триггер работает как при добавлении нового элемента в таблицу Purchases, так и при изменении значения одной из колонок



1. Скрипт триггера CalculateTotalTrigger

# ЗАКЛЮЧЕНИЕ

В данной курсовой работе были изучены методы и закреплены знания в проектировании реляционных баз данных (РБД) в системе управления базами данных (СУБД) PostgreSQL на примере базы данных «Магазин по продаже оргтехники». В ходе работы были рассмотрены основные этапы проектирования базы данных, такие как:

1. Анализ предметной области и определение требований к базе данных.
2. Разработка концептуальной модели данных с использованием ER-диаграмм.
3. Преобразование концептуальной модели в логическую модель с нормализацией данных для обеспечения целостности и устранения избыточности.
4. Реализация физической модели базы данных в СУБД PostgreSQL с использованием SQL-запросов для создания таблиц, ключей и индексов.
5. Наполнение базы данных тестовыми данными и выполнение основных операций по добавлению, обновлению и выборке данных.

Таким образом, в результате проделанной работы были получены практические навыки проектирования и реализации реляционных баз данных, что является важным аспектом для дальнейшего профессионального развития в области информационных технологий и систем управления базами данных.