ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Высшая школа бизнеса

Отчет по контрольному домашнему заданию

**ПО БАЗАМ ДАННЫХ**

2 курс, направление подготовки: 38.03.05 «Бизнес-информатика»

образовательная программа «Бизнес-информатика»

Выполнили:

Калякина Мария, ББИ1908

Бокова Полина, ББИ1910

Петяева Елизавета, ББИ1908

Ради Мария, ББИ1908

Москва 2021

Оглавление

[Постановка задачи 3](#_Toc74476799)

[Описание предметной области: 3](#_Toc74476800)

[Проектирование базы данных 6](#_Toc74476801)

[Инфологическая модель кинотеатра 6](#_Toc74476802)

[Описание сущностей 7](#_Toc74476803)

[Даталогическая модель 8](#_Toc74476804)

[Описание таблиц 8](#_Toc74476805)

[Описание связей между таблицами 12](#_Toc74476806)

[Реализация базы данных 15](#_Toc74476807)

[Заполнение начальными данными 16](#_Toc74476808)

[Функции 18](#_Toc74476809)

[Представления 19](#_Toc74476810)

[Триггеры 21](#_Toc74476811)

[Хранимые процедуры 23](#_Toc74476812)

[Запросы 29](#_Toc74476813)

[Индекс 38](#_Toc74476814)

[Отчетные формы Power BI и Excel 41](#_Toc74476815)

[Отчеты в Power BI 41](#_Toc74476816)

[Отчет в MS Excel 43](#_Toc74476817)

[Описание роли участников проекта 44](#_Toc74476818)

# Постановка задачи

Объектом исследования в данной работе служит среднестатистический интернет-магазин, предметом исследования – база данных интернет-магазина, разработанная для систематизации и автоматизации его работы.

Цель работы – создание практичной и удобной для использования базы данных, оптимизирующей работу с товарами и заказами, для сотрудников интернет-магазина.

В ходе работы перед нами были поставлены следующие задачи:

1. Анализ основных положений, связей и механизмов, присущих интернет-магазину
2. Проектирование базы данных, состоящее из построения инфологической модели предметной области и даталогической моделей данных.
3. Реализация базы данных в СУБД.
4. Предоставление отчетных форм в Power BI и Excel

## Описание предметной области:

Интернет-магазин (англ. online shop или e-shop) — сайт, торгующий товарами в интернете. Позволяет пользователям сформировать заказ на покупку, выбрать способ оплаты и доставки заказа в сети Интернет. Мы выбрали в качестве объекта исследования интернет-магазин одежды.

С помощью сайта покупателю предоставляется возможность выбрать необходимые ему товары из предложенных на сайте. Продукция разделена по категориям, представляющим собой совокупность товаров одного вида и используемых для упрощения поиска необходимой продукции. Затем клиент имеет возможность добавить понравившиеся товары в корзину. После чего выбрать удобный формат оплаты и доставки и оформить полноценный заказ, заполнив некоторое количество данных о себе.

Спроектированная база данных интернет-магазина обеспечивает оптимизацию и систематизацию процесса учета товаров, а также предоставляет информационные ресурсы для аналитического отдела компании (например, наиболее популярные категории товаров).

Ведение учета продукции включает в себя, в первую очередь, заполнение пользователями данных о поставках/заказах, информация о которых отражается в базе данных. Сами товары представляют собой совокупность товарных характеристик, описывающих их: категория, бренд, цвет, а также размер. Ведется учет количества товаров в наличии, данное значение автоматически изменяется при оформлении поставок и заказов с указанием количественной величины (шт).

Все прибытия/выбытия товаров записываются в транзакции, которые учитывают денежную массу от поступления или отчисления. Затем при необходимости составления отчетности сотрудники, используя необходимые запросы, смогут агрегировать данные за конкретный промежуток времени.

Процесс доставки осуществляется со склада, на котором в наличии есть наибольшее число товаров из сформированного заказа (склад определяется автоматически).

Пользователи БД могут использовать данные о заказанных товарах и сформированных корзинах для проведения анализа работы сайта, выявления потребностей покупателей (посредством маркетингового исследования), и последующей регуляции товарного обращения в зависимости от этого.

Приложение:

Потенциально разработка приложения будет осуществляться на программном языке Python. Приложение будет иметь окно аутентификации с полями, в которые пользователь должен будет ввести собственные логин и пароль, чтобы получить доступ.

Программа позволит сотрудникам, ведущим учет, заполнять новыми данными о новых поставках товаров. А также получать агрегированную информацию, которая позволит удобно вести отчетность.

Для пользователей-маркетологов будут доступны и удобны для использования данные о покупках, заказах и составах потребительской корзины - в целом о том, что представляет пользу для проведения анализа и создания будущей стратегии продаж. Храниться эти данные будут в соответствующих таблицах.

Также есть в них будет возможность отфильтровать выбранную информацию по дате, чтобы получать наиболее актуальную и, как следствие отслеживать тенденции рынка.

Отчетные формы:

* Результаты маркетингового анализа спроса на товары, с фильтрацией по конкретным характеристикам для оптимизации подбора предпочтений для целевой аудитории и принятии дальнейших решений относительно будущих поставок;
* Результаты анализа закупок товара у поставщиков, для принятия решения относительно целесообразности дальнейшего с ними сотрудничества;
* Результаты анализа целевой аудитории, их предпочтений, финансовых возможностей, для усовершенствования стратегии реализации товаров в интернете;
* Результаты анализов отдела логистики, основанные на информации о доставках и использующиеся для принятия решения о количественном размещении товаров на различных складах

# 

# Проектирование базы данных

После того, как была проанализирована и описана предметная область, необходимо спроектировать инфологическую и даталогическую модели, и для лучшего их понимания также описать сущности, таблицы и связи между ними.

## Инфологическая модель кинотеатра

Инфологическая модель позволяет структурированно отобразить основные сущности, принципы и механизмы взаимосвязи между ними, основываясь на описании предметной области. Данная модель была разработана с помощью CASE-средства ERD Plus, базирующегося на методологии Питера Чена.

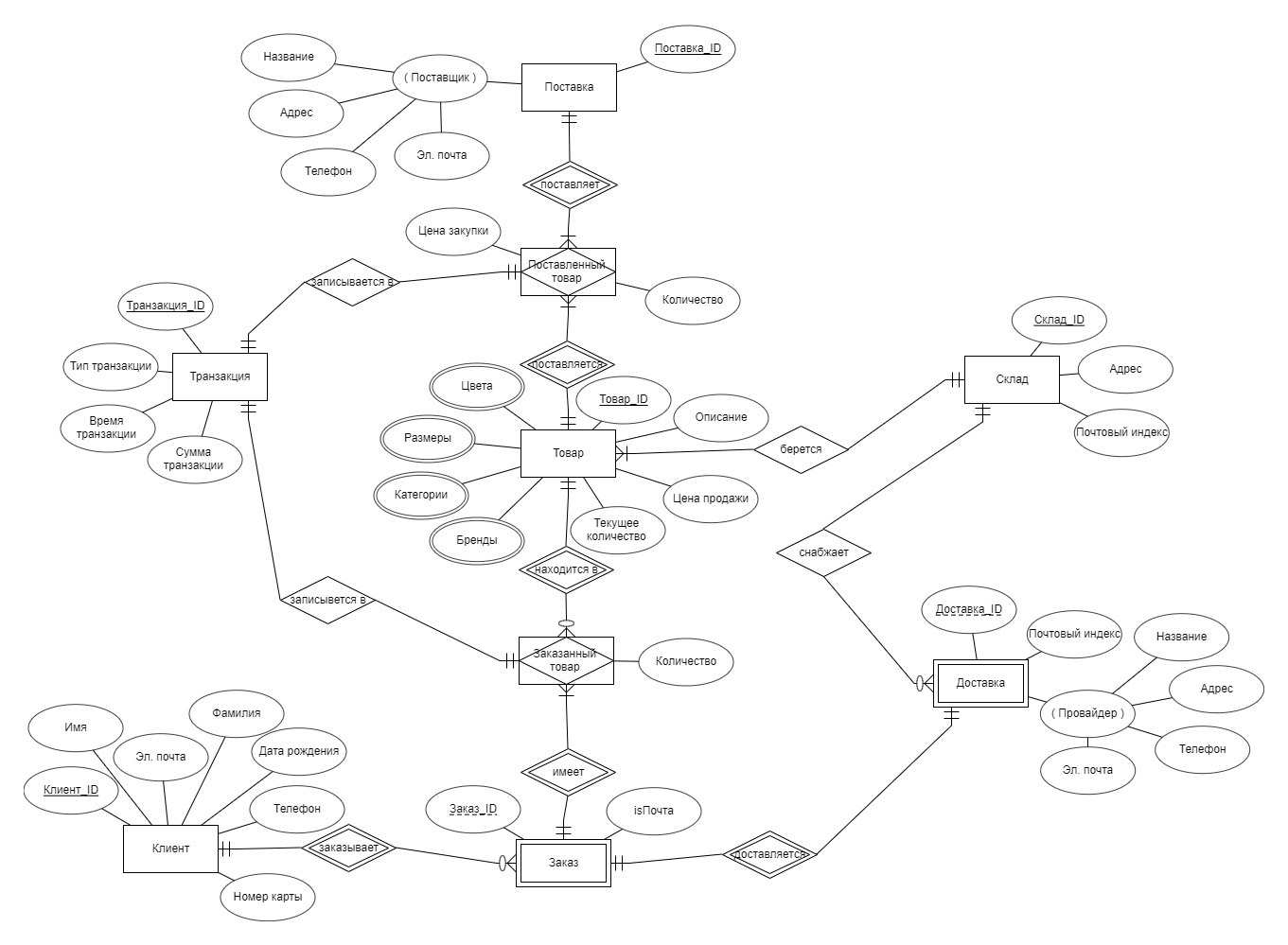


Рисунок 1. Инфологическая модель

### Описание сущностей

При проектировании инфологической модели было создано 9 сущностей, которые делятся на два типа: сущности-справочники и сущности-документы.

Сущности-справочники:

Клиент – справочные данные о клиентах интернет-магазина.

Склад - справочные данные о складах организации (адреса и контактные данные).

Сущности-документы:

1. Поставка - данные о снабжениях компании товарами, а также поставщиках с их контактными данными.

2. Товар – данные о единице продаваемого товара и его характеристики, такие как цвет, размер, категория, цена, бренд и описание, а также его текущее количество в наличии.

3. Заказ – данные о том, что заказал клиент, содержащие некоторые детали доставки (клиент в праве выбрать почтовую или курьерскую службу доставки).

4. Поставленный товар – хранит данные о том, в каком количестве, по какой цене и в какой поставке был поставлен конкретный товар.

5. Заказанный товар – содержит данные о том, в каком количестве и в каком заказе был заказан конкретный товар.

6. Транзакция - хранит данные о прибытиях и выбытиях товаров, их времени совершения и стоимости.

7. Доставка – содержит данные о том, каким способом доставляется конкретный заказ и с помощью какой службы доставки с ее контактными данными.

## Даталогическая модель

При создании данной модели необходимо было сопоставить сущностям из инфологической модели таблицы из даталогической модели, а также соблюсти требования, соответствующие третьей нормальной форме базы данных. В частности, сущность Товар стала совокупностью таких таблиц, как item, item\_list, color, category, brand, size. Атрибут Поставщик сущности Поставка стал самостоятельной таблицей, хранящей данные обо всех поставщиках и их контактных данных, аналогично стал отдельной таблицей атрибут Провайдер сущности Доставка. Остальные таблицы соответствуют одноименным сущностям из инфологической модели. Данная схема была создана с помощью CASE-средства от Erwin Data Modeler.

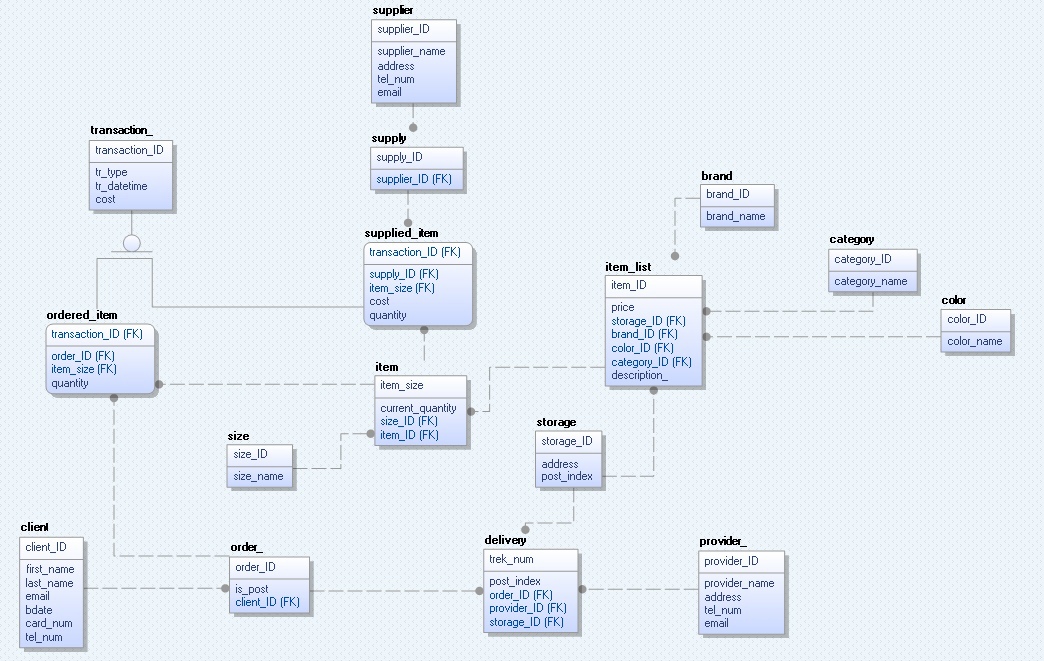


Рисунок 2. Даталогическая модель

### Описание таблиц

*Таблица 1. Описание таблиц*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблица | Атрибут | Тип данных | Описание | PK/FK |
| brand | brand\_ID | int | Идентификатор бренда | Primary key |
| brand\_name | nvarchar(50) | Название бренда |  |
| color | color\_ID | int | Идентификатор цвета | Primary key |
| color\_name | nvarchar(30) | Название цвета |  |
| category | category\_ID | int | Идентификатор категории | Primary key |
| category\_name | nvarchar(30) | Название категории |  |
| size | size\_ID | int | Идентификатор размера | Primary key |
| size\_name | nvarchar(5) | Код размера |  |
| supplier | supplier\_ID | int | Идентификатор поставщика | Primary key |
| supplier\_name | nvarchar(30) | Название компании поставщика |  |
| address | nvarchar(70) | Адрес главного офиса поставщика |  |
| tel\_num | nvarchar(20) | Контактный номер поставщика |  |
| email | nvarchar(60) | Электронная почта поставщика |  |
| storage | storage\_ID | int | Номер склада | Primary key |
| address | nvarchar(70) | Адрес склада |  |
| post\_index | nchar(10) | Почтовый индекс склада |  |
| item\_list | item\_ID | int | Идентификатор типа товара | Primary key |
| color\_ID | int | Идентификатор цвета | Foreign key |
| brand\_ID | int | Идентификатор бренда | Foreign Key |
| category\_ID | int | Идентификатор категории | Foreign Key |
| storage\_ID | int | Идентификатор склада | Foreign key |
| description\_ | nvarchar(70) | Описание типа товара |  |
| price | money | Цена продажи одной единицы товара данного типа |  |
| item | item\_size | int | Идентификатор товара | Primary key |
| item\_ID | int | Идентификатор типа товара | Foreign key |
| size\_ID | int | Идентификатор размера | Foreign key |
| current\_quantity | int | Текущее количество товара в наличии |  |
| transaction | transaction\_ID | int | Идентификатор денежной транзакции |  |
| tr\_type | nvarchar(50) | Тип денежной транзакции |  |
| tr\_datetime | datetime | Дата и время денежной транзакции |  |
| supply | supply\_ID | int | Идентификатор поставки | Primary key |
| supplier\_ID | int | Идентификатор поставщика | Foreign key |
| supplied\_item | transaction\_ID | int | Идентификатор транзакции | Primary key & Foreign key |
| item\_size | int | Имя сотрудника | Foreign key |
| supply\_ID | int | Фамилия сотрудника | Foreign key |
| quantity | int | Дата рождения сотрудника |  |
| client | client\_ID | int | Идентификатор клиента | Primary key |
| first\_name | nvarchar(50) | Имя клиента |  |
| last\_name | nvarchar(50) | Фамилия клиента |  |
| bdate | date | Дата рождения клиента |  |
| email | nvarchar(60) | Электронная почта клиента |  |
| card\_num | nchar(16) | Банковская карта клиента |  |
| tel\_num | nvarchar(20) | Телефон клиента |  |
| order\_ | order\_ID | int | Идентификатор заказа | Primary key |
| client\_ID | int | Идентификатор клиента | Foreign key |
| is\_post | bit | Доставляется ли заказ почтой |  |
| ordered\_item | transaction\_ID | int | Идентификатор смены сотрудника | Primary key & Foreign key |
| item\_size | int | Идентификатор рабочего места | Foreign key |
| quantity | int | Идентификатор сотрудника |  |
| provider\_ | provider\_ID | int | Идентификатор службы доставки | Primary key |
| provider\_name | nvarchar(50) | Название службы доставки | Foreign key |
| address | nvarchar(70) | Адрес главного офиса службы доставки |  |
| tel\_num | nvarchar(20) | Контактный телефон службы доставки |  |
| email | nvarchar(60) | Электронная почта службы доставки |  |
| delivery | trek\_num | int | Идентификатор доставки | Primary key |
| provider\_ID | int | Идентификатор службы доставки | Foreign key |
| storage\_ID | int | Идентификатор склада, с которого осуществляется доставка | Foreign key |
| post\_index | nchar(10) | Почтовый индекс прибытия заказа |  |
| order\_ID | int | Идентификатор заказа | Foreign key |

### Описание связей между таблицами

*Таблица 2. Описание связей*

|  |  |  |  |
| --- | --- | --- | --- |
| Связь | Отношение | Родительская таблица | Дочерняя таблица |
| client\_order | один ко многим | client | order |
| color\_item\_list | один ко многим | color | item\_list |
| category\_item\_list | один ко многим | category | item\_list |
| supplier\_supply | один ко многим | supplier | supply |
| supply\_supplied\_item | один ко многим | supply | supplied\_item |
| item\_supplied\_item | один ко многим | item | supplied\_item |
| order\_delivery | один к одному | order\_ | delivery |
| item\_list\_item | один ко многим | item\_list | item |
| transaction\_supplied | один к одному | transaction | supplied\_item |
| transaction\_ordered | один к одному | transaction | ordered\_item |
| provider\_delivery | один ко многим | provider\_ | delivery |
| storage\_delivery | один ко многим | storage | delivery |
| order\_ordered\_item | один ко многим | order\_ | ordered\_item |
| item\_ordered\_item | один ко многим | order\_ | ordered\_item |
| size\_item | один ко многим | size | item |
| brand\_item\_list | один ко многим | brand | item\_list |

1. client\_order – клиент может создать множество заказов, заказ привязан к одному конкретному клиенту. Не идентифицирующая связь.
2. color\_item\_list – на сайте могут быть вещи одинакового цвета, для вещи отмечается один превалирующий цвет. Не идентифицирующая связь.
3. category\_item\_list -на сайте могут быть вещи из одной и той же категории, каждая вещь относится к одной конкретной категории. Не идентифицирующая связь.
4. supplier\_supply – поставщик может обеспечить множеством поставок, каждая поставка относится к одному конкретному поставщику. Не идентифицирующая связь.
5. supply\_supplied\_item –в одной поставке может быть множество поставленных товаров, поставленный товар соответствует одной конкретной поставке. Не идентифицирующая связь.
6. item\_supplied\_item – один и тот же товар может поставляться множество раз, поставленный товар соответствует одному конкретному товару из наличия. Не идентифицирующая связь.
7. order\_delivery – для каждого заказа должна быть оформлена одна конкретная доставка. Не идентифицирующая связь.
8. item\_list\_item – в наличии может быть множество товаров одного типа (но разных размеров), каждый товар соответствует одному конкретному типу. Не идентифицирующая связь.
9. transaction\_supplied – каждая денежная транзакция соответствует поступлению товара одного типа из поставки. Идентифицирующая связь.
10. transaction\_ordered – каждая денежная транзакция соответствует заказу товара одного типа из поставки. Идентифицирующая связь.
11. provider\_delivery –служба доставки может обеспечить множество доставок, доставка осуществляется одной конкретной службой доставки. Не идентифицирующая связь.
12. storage\_delivery – доставка с какого-то склада может осуществляться множество раз, доставка происходит из одного конкретного склада. Не идентифицирующая связь.
13. order\_ordered\_item –в заказе может быть множество заказанных товаров, заказанный товар принадлежит одному конкретному заказу. Не идентифицирующая связь.
14. item\_ordered\_item – каждый товар может заказываться множество раз, заказанный товар является одним из товаров в наличии. Не идентифицирующая связь.
15. size\_item – может быть множество вещей одного размера, вещи присваивается один конкретный размер. Не идентифицирующая связь.
16. brand\_item\_list – может быть множество вещей одного бренда, вещи присваивается один конкретный бренд. Не идентифицирующая связь.

# Реализация базы данных

При построении диаграммы базы данных в DBeaver была использована функция «Schema Generation» из Erwin Data Modeler, генерирующая скрипт для ее создания. Содержимое скрипта было перенесено в запрос к конкретной базе данных Asos, также были вручную проставлены ограничения IDENTITY (1, 1) для первичных ключей (кроме таблиц supplied\_item и ordered\_item т. к. они являются подтипами таблицы transaction\_), чтобы снять с пользователей задачу в их мануальном заполнении.

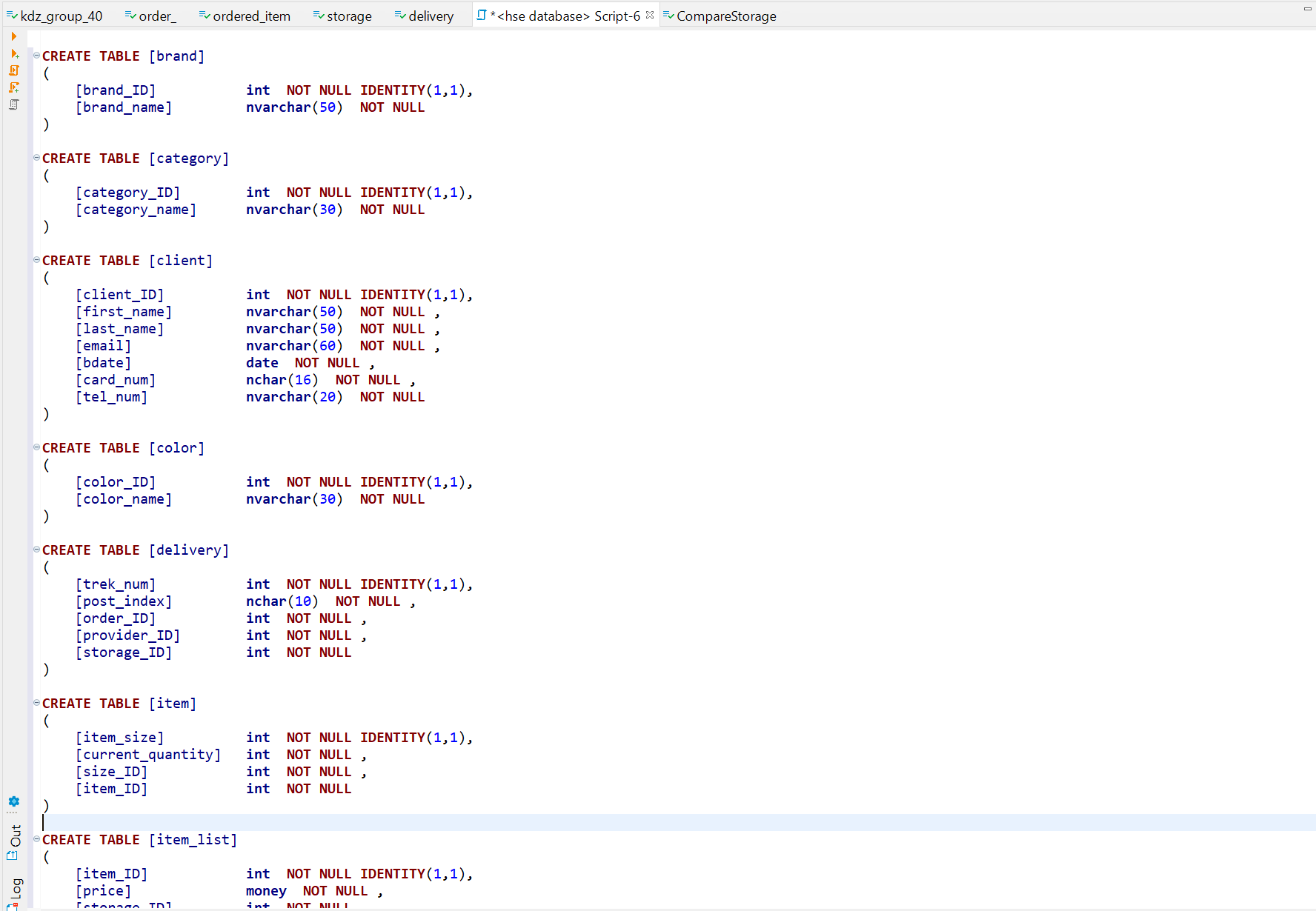


Рисунок 3. Создание запроса на основе скрипта

Также была создана диаграмма в DBeaver для проверки результата выполнения скрипта.

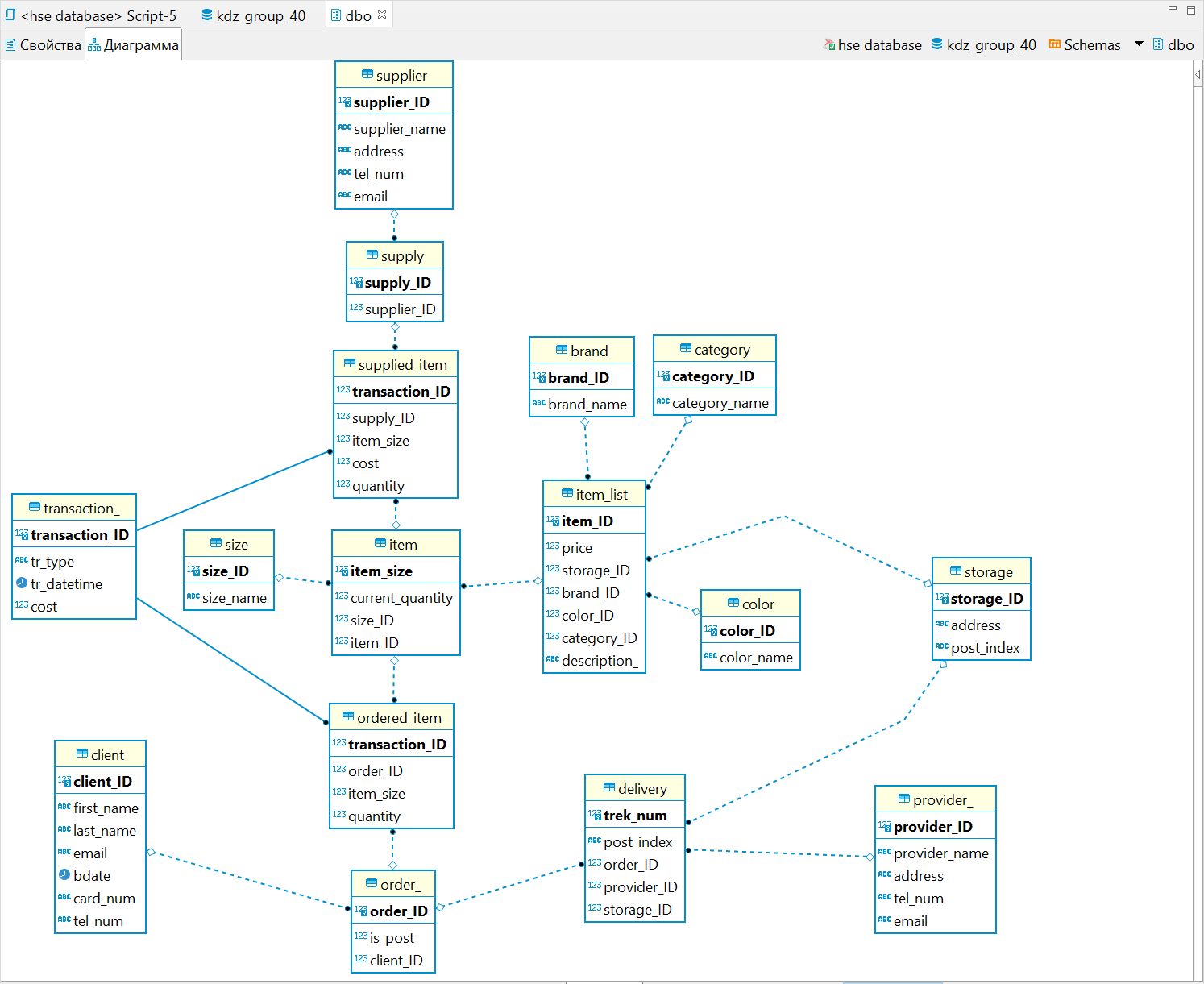


Рисунок 4. Диаграмма базы данных, созданная в DBeaver

## Заполнение начальными данными

На данном этапе работы таблицы были заполнены значениями с помощью оператора INSERT. Таблицы item, transaction\_, supplied\_item, ordered\_item, а также delivery остались без изменений, поскольку для их заполнения желательно пользоваться триггерами и хранимыми процедурами для обеспечения корректности ввода данных и автоматического определения ряда атрибутов. Результаты их выполнения будут продемонстрированы позже.

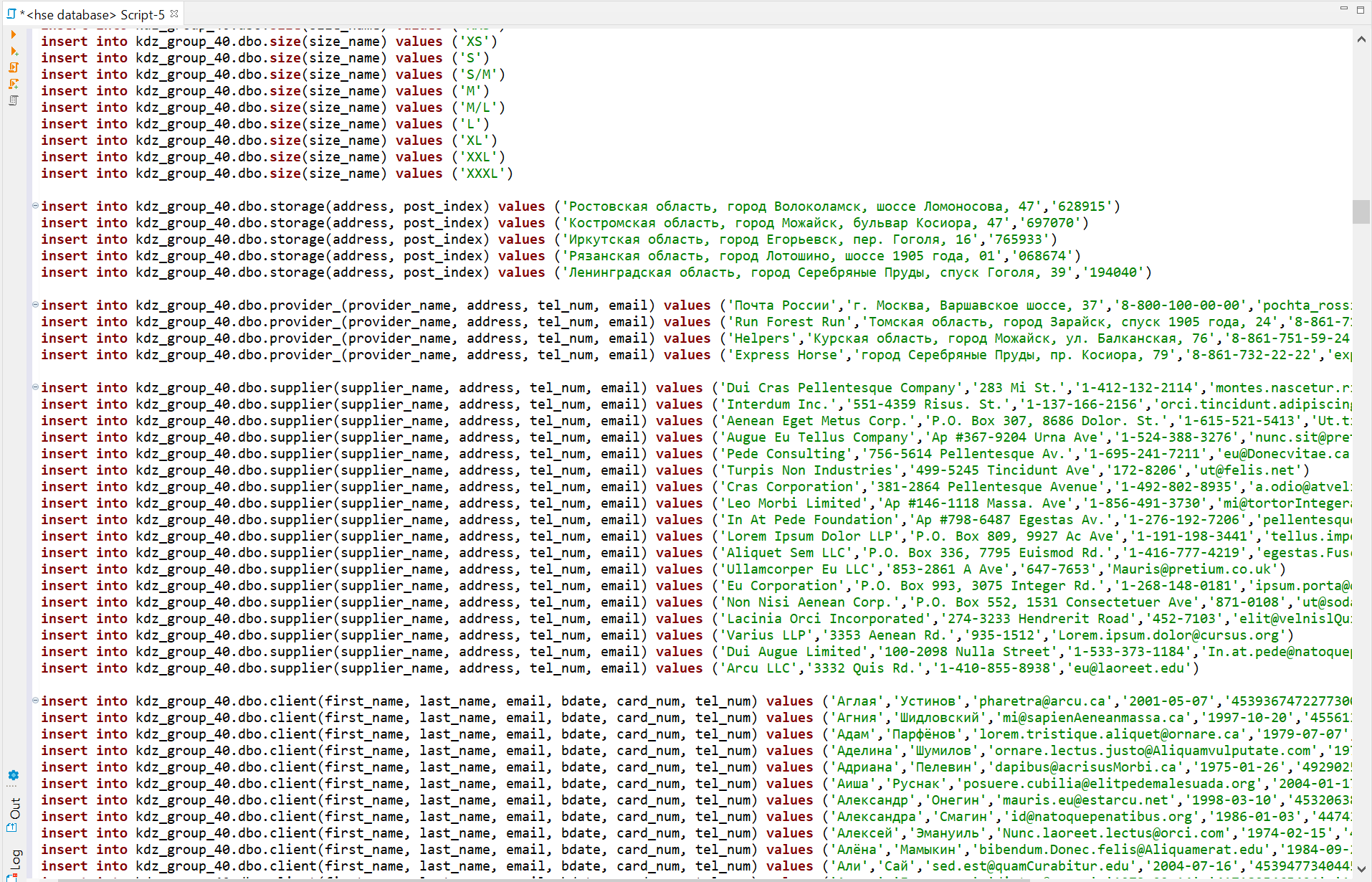


Рисунок 5. Пример заполнения таблиц.

Результат запроса был проверен при рассмотрении строк в таблице client.

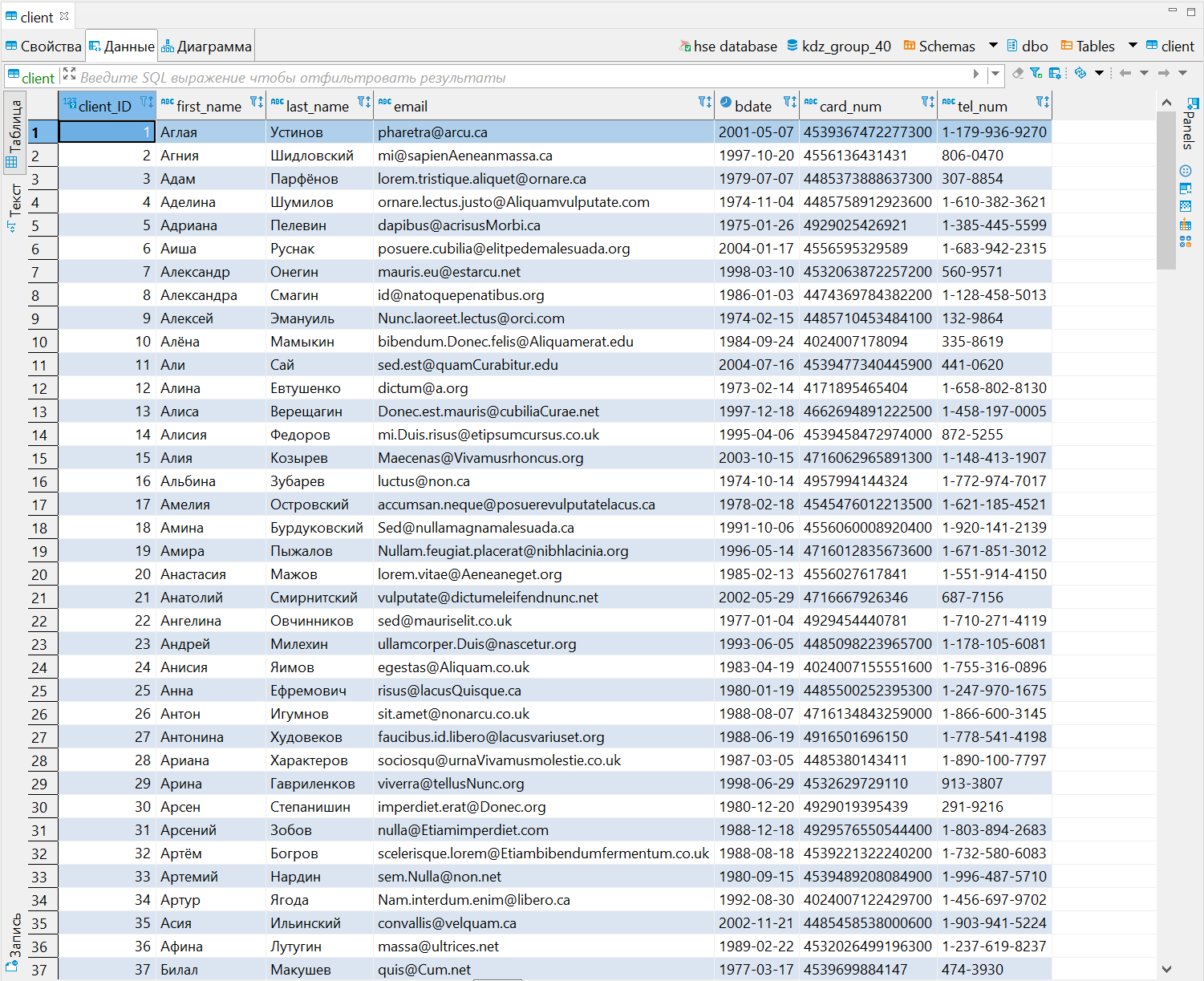


Рисунок 6. Проверка вставки данных в таблицу client.

## Функции

Ниже приведены функции, которые призваны ускорить и упростить работу пользователя с базой данных.

1. Для того, чтобы оформить и совершить доставку вещей, находящихся на разных складах, необходимо определить склад, с которого должен быть отправлен уже собранный заказ. Целесообразно предположить, что таким складом является тот, в котором находится больше всего вещей из заказа. Функция define\_storage принимает в качестве параметра заказ, и возвращает оптимальный для доставки склад.

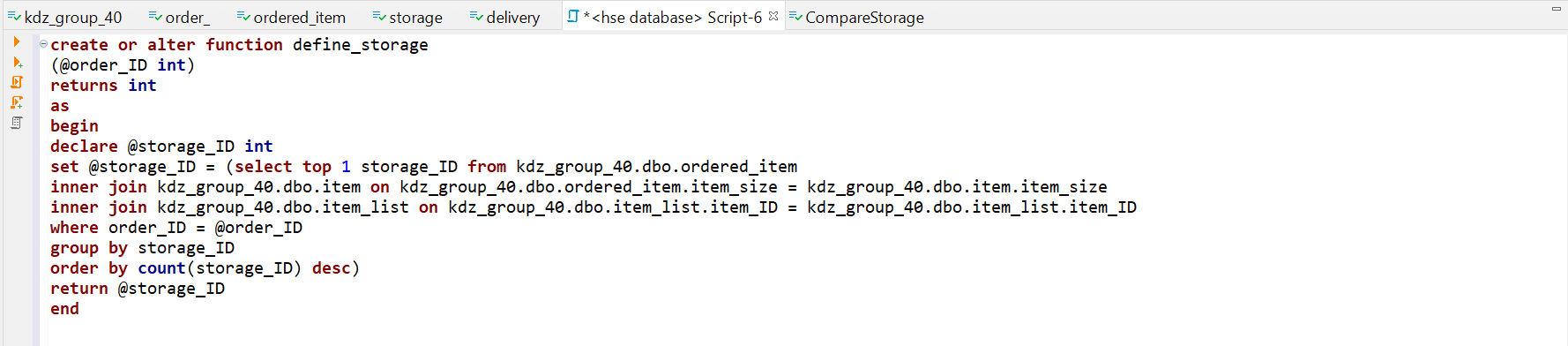


Рисунок 7. Код функции define\_storage

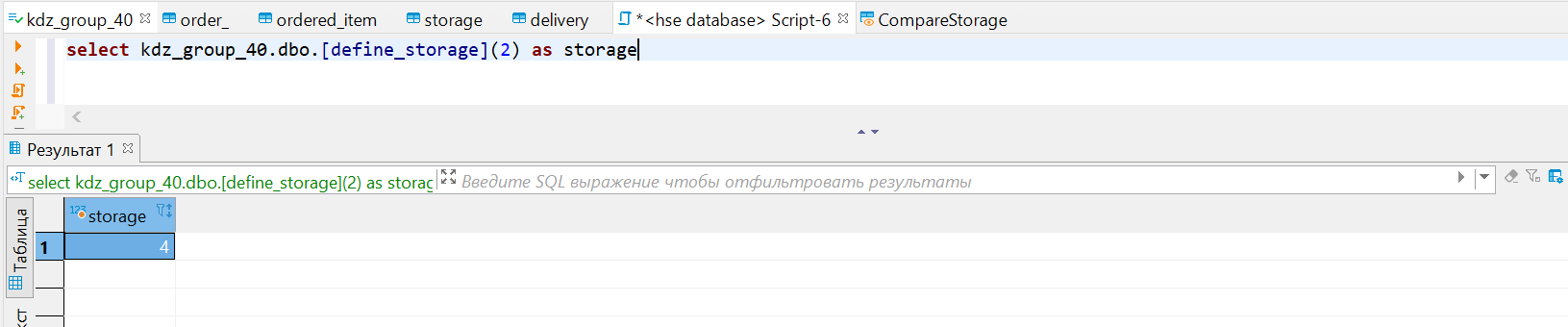


Рисунок 7.1. Результат выполнения функции define\_storage

Чтобы убедиться в корректности результата, были получены данные о складах, на которых находятся заказанные вещи.

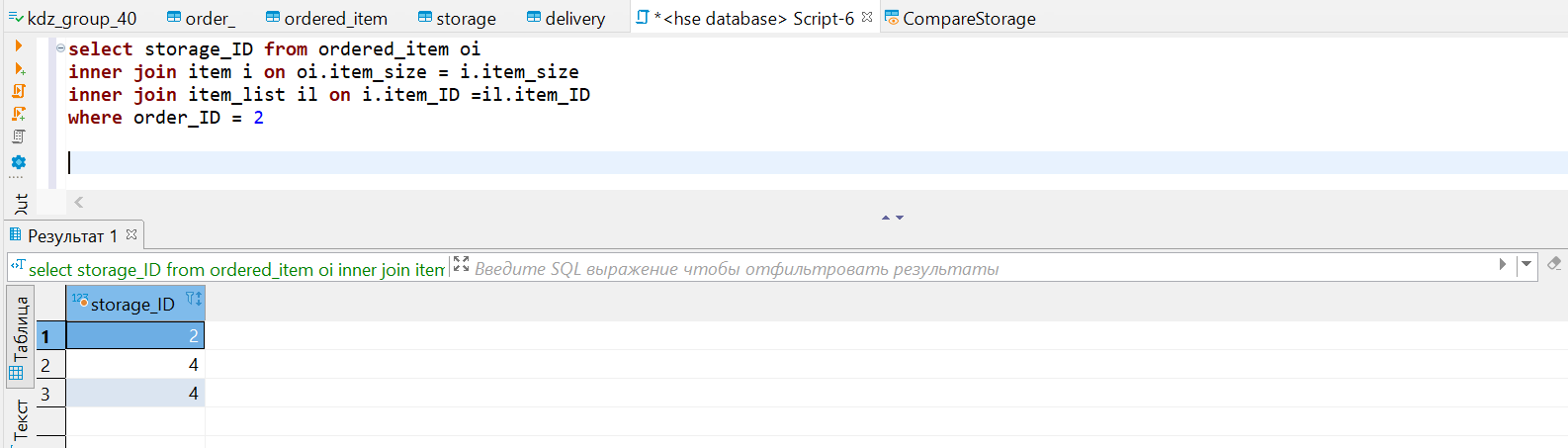
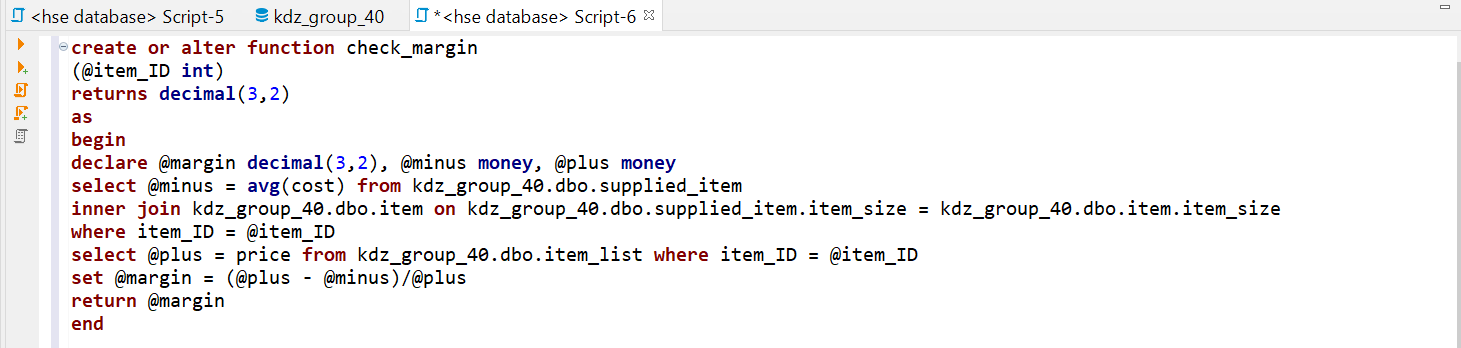
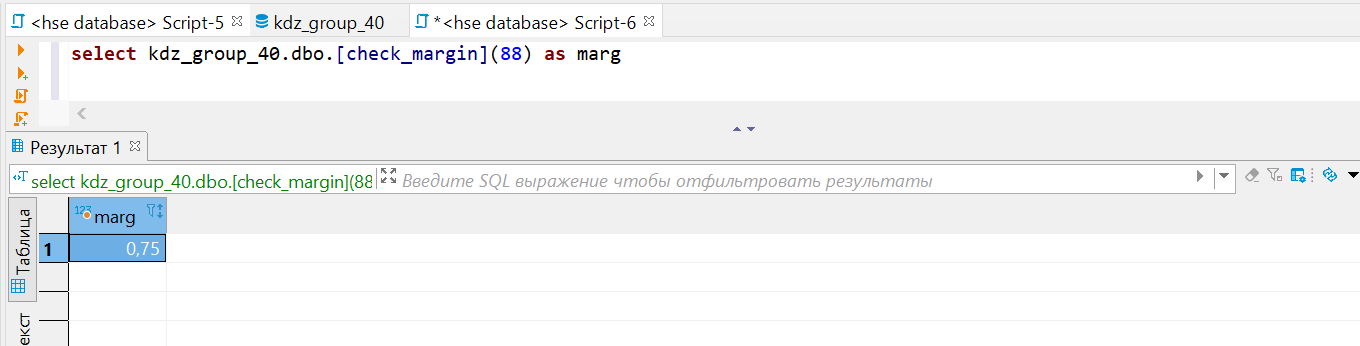


Рисунок 7.2. Запрос для получения данных о складах заказанных товаров и его результат выполнения

1. Цены поставщиков могут постоянно меняться, поэтому необходимо отслеживать, какую прибыль приносит каждый тип товара. Если в среднем цена на поставку повысится, то высчитываемое данной функцией значение нормы маржинальной прибыли снизится, что даст сигнал к повышению цены продажи товара определенного типа.



Изображение 8. Код функции check\_margin



Изображение 8.1. Результат выполнения функции check\_margin

## Представления

Ниже показаны представления, с помощью которых можно визуализировать агрегированные данные, а также упростить выполнение некоторых запросов.

1. Представление ShowMargin выводит для каждого типа товара его цену продажи, среднюю стоимость закупки и норму маржинальности, полученную через функцию check\_margin, описанную в предыдущем пункте.

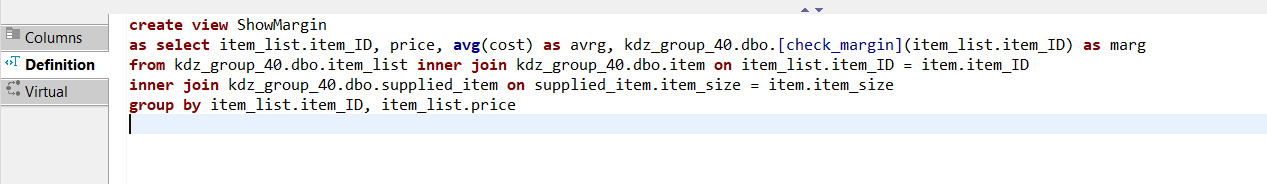


Рисунок 9. Код представления ShowMargin

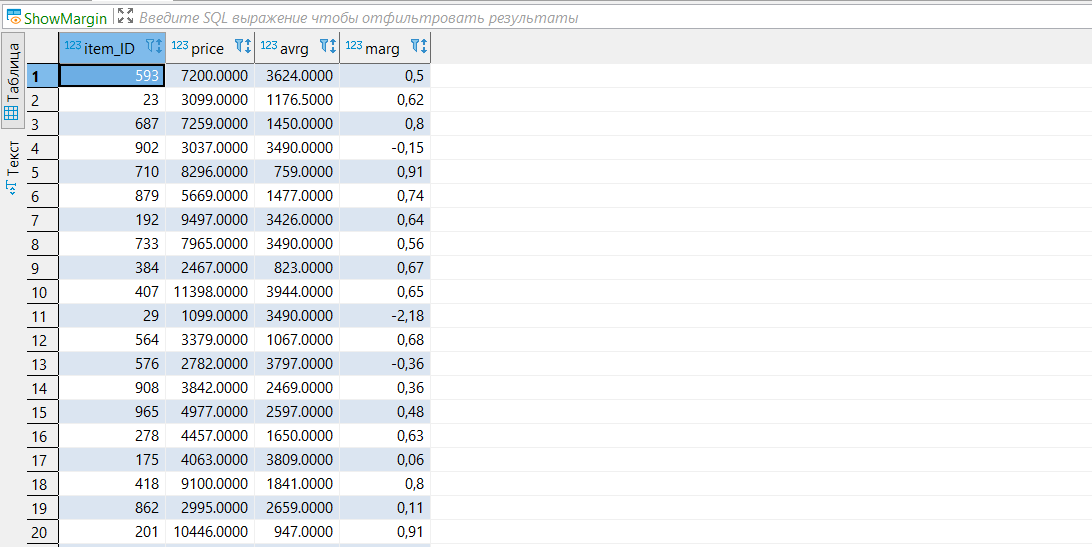


Рисунок 9.1. Данные представления ShowMargin

1. Представление CompareStorage показывает, где находится товар, а также сколько раз и откуда он отгружался.

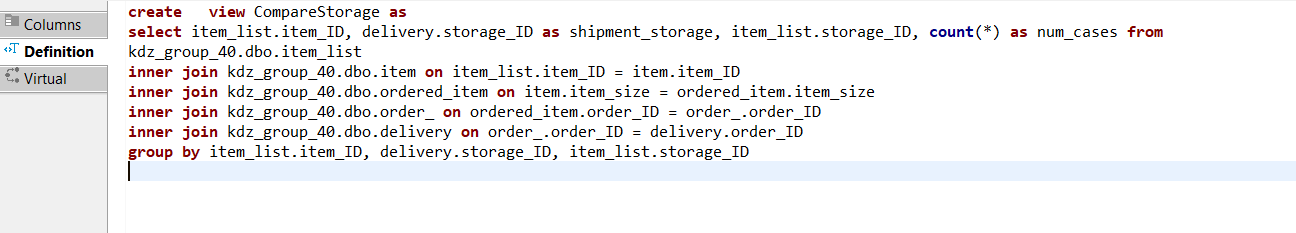
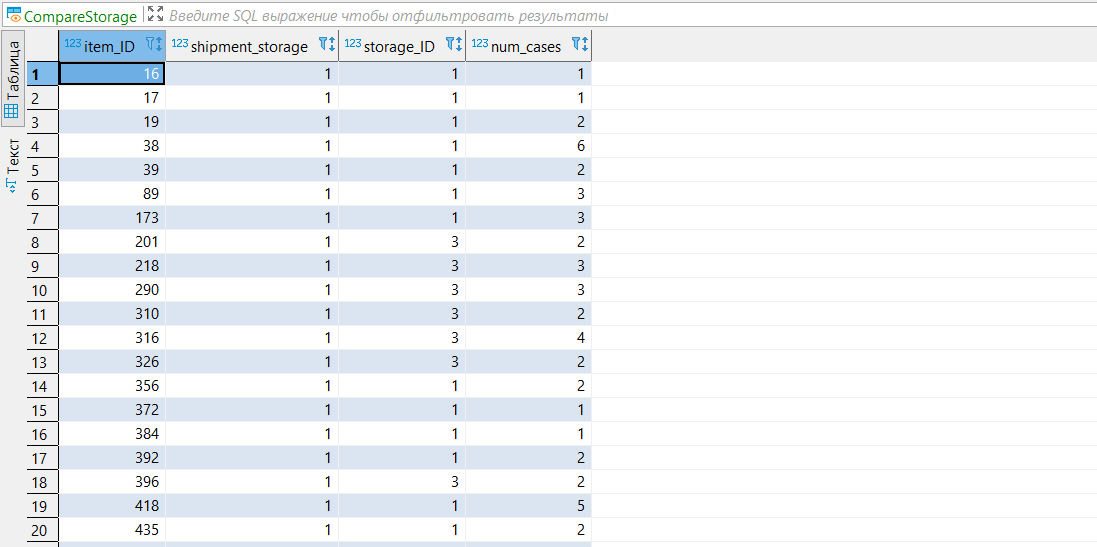
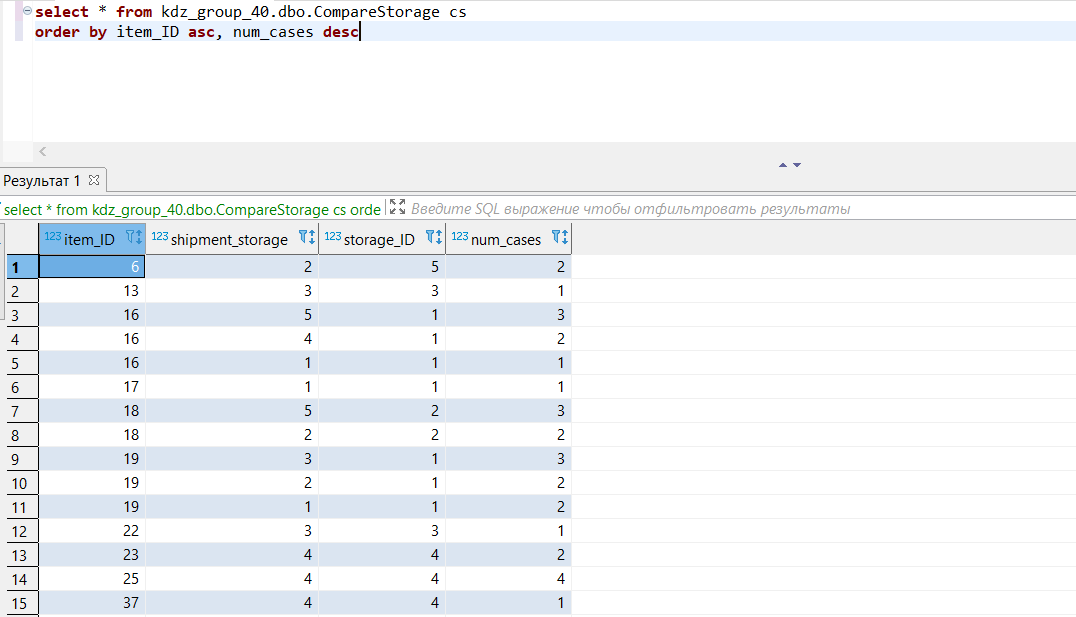


Рисунок 10. Код представления CompareStorage



*Рисунок 10.1. Данные представления CompareStorage*

С помощью созданного представления получилось упростить процесс выполнения запроса. Запрос показывает отсортированные склады по количеству отгрузок, чтобы понять откуда товар отгружался чаще всего.



*Рисунок 10.2. Запрос с использованием представления CompareStorage*

## Триггеры

В данной работе триггер служит для проверки корректности ввода данных. Таким образом, при оформлении доставки пользователем базы данных происходит проверка занесенного в строку провайдера. Если клиент указал почту в качестве службы доставки, а пользователь занес данные курьерской службы, то провайдер автоматически поменяется на почту.

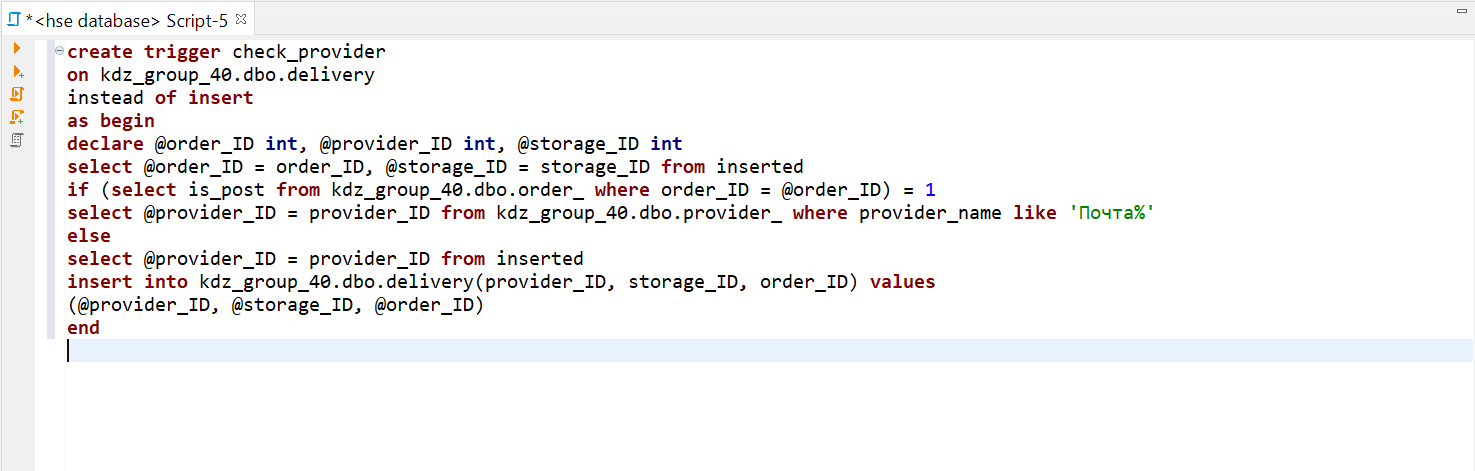


Рисунок 11. Код триггера check\_provider

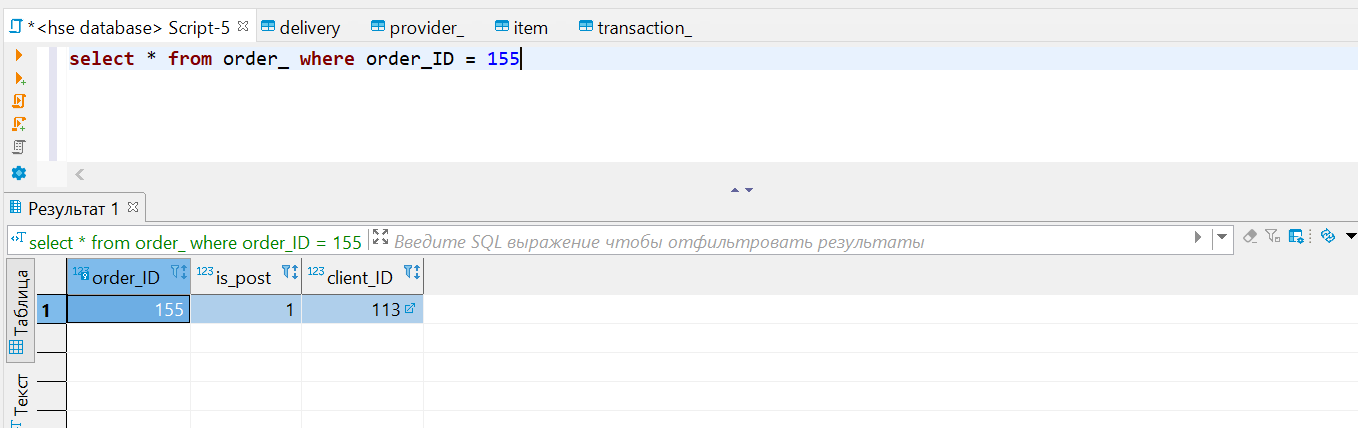
Чтобы убедиться, что триггер работает правильно, была оформлена доставка для заказа, в котором указана доставка почтой. 

Рисунок 11.1. Пример заказа с доставкой почтой

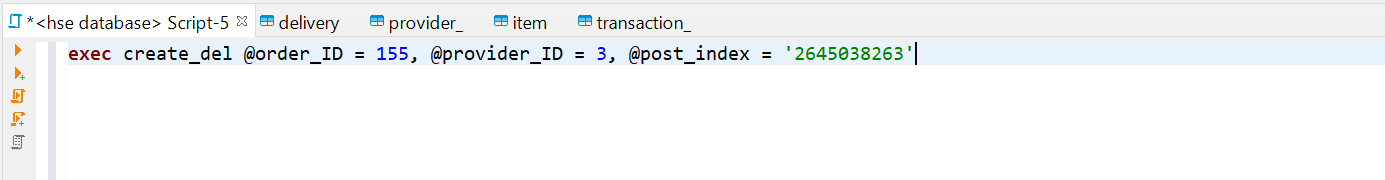


Рисунок 11.2. Пример запроса с применением триггера check\_provider

Теперь можно убедиться наглядно при анализе строк таблицы delivery. Идентификатор провайдера стал равным единице несмотря на то, что вводился провайдер под номером 3.

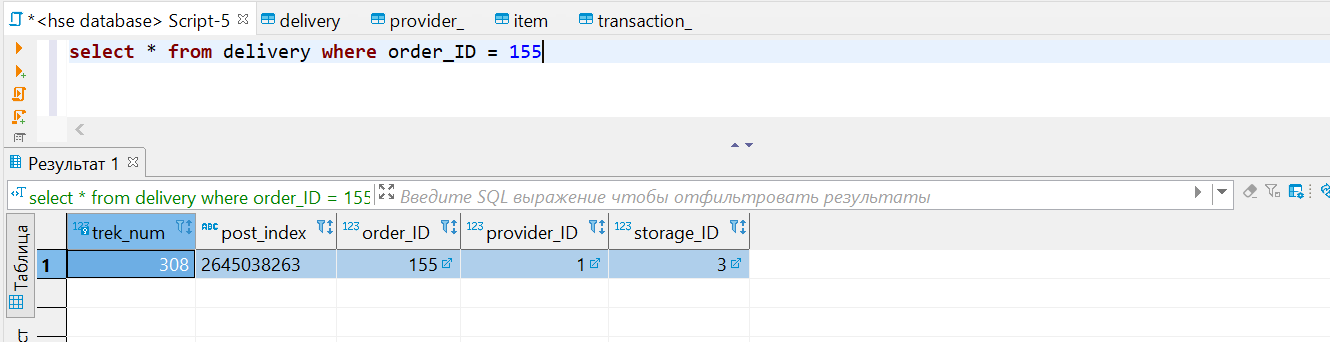


Рисунок 11.3. Результат запроса с применением триггера check\_provider

Провайдеру под номером 1 соответствует Почта России. Триггер отработал корректно.

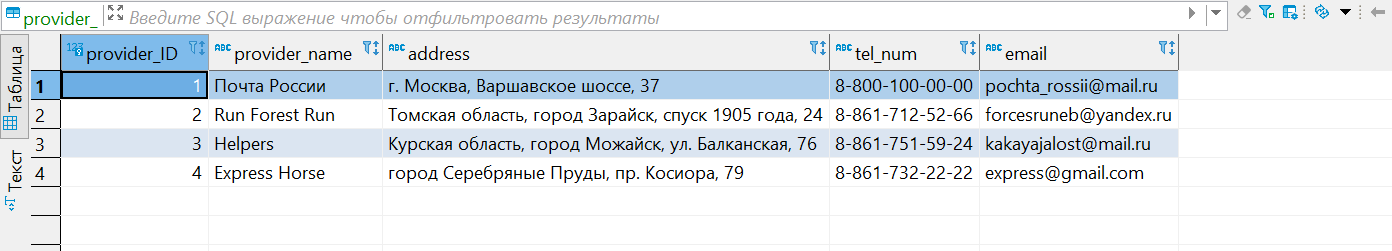


Рисунок 11.4. Строки таблицы provider\_

## Хранимые процедуры

В данной работе хранимые процедуры призваны доопределять некоторые атрибуты при вставке новой строки в ряде таблиц, проверять корректность ввода данных, а также фиксировать (или логировать) сопутствующие денежные потоки.

1. После оформления поставки необходимо заполнить данные о том, какие в ней прибыли товары, в каком количестве и по какой цене. Функция автоматически прибавляет к значению текущего количества в наличии (обновляет таблицу item) количество прибывших товаров, или добавляет новую строку с поставленным количеством (если товар прибыл впервые). Также плата (и время ее осуществления) за каждый поставленный тип товара, рассчитывающаяся как произведение стоимости поставленной единицы товара на количество товара, заносится в таблицу transaction\_.

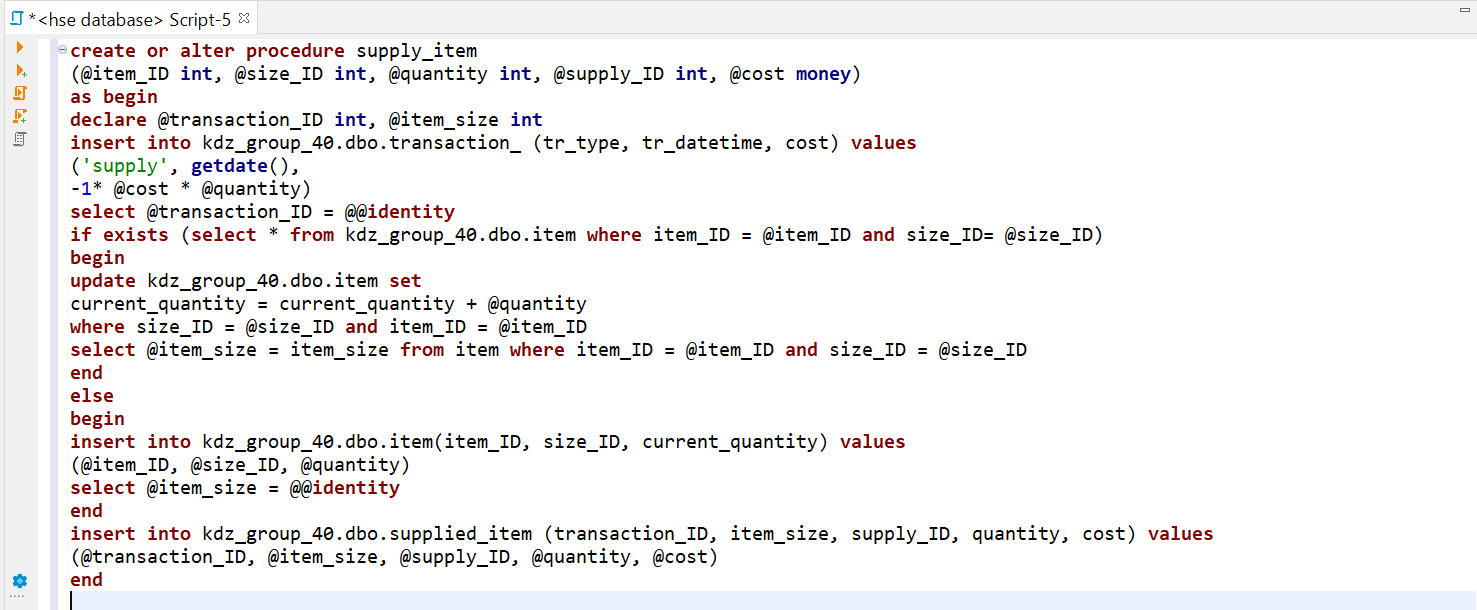


Рисунок 12. Код процедуры supply\_item

В качестве примера работы данной процедуры был выполнен запрос с ее использованием. Перед этим необходимо проверить текущее количество товара в наличии. На данный момент товара под номером 10 всего 580 шт.

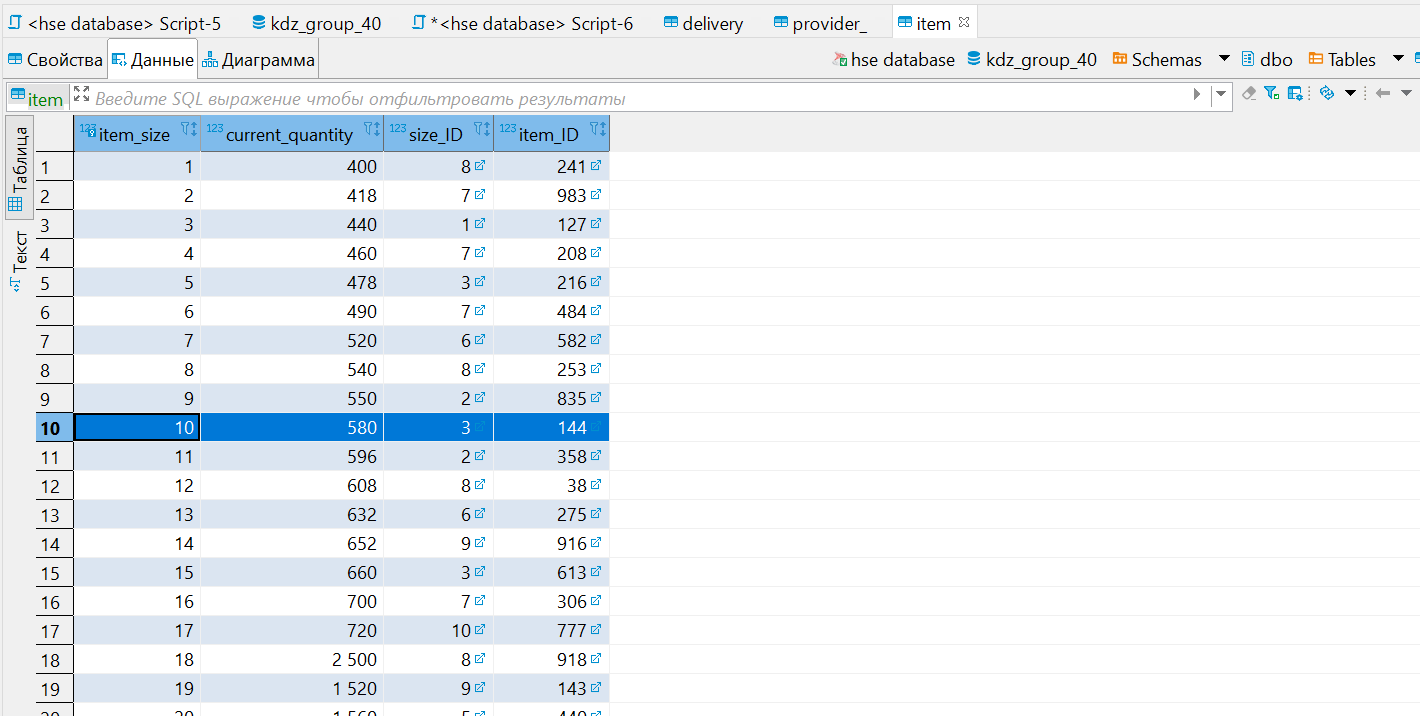


Рисунок 12.1. Вид строк таблицы item

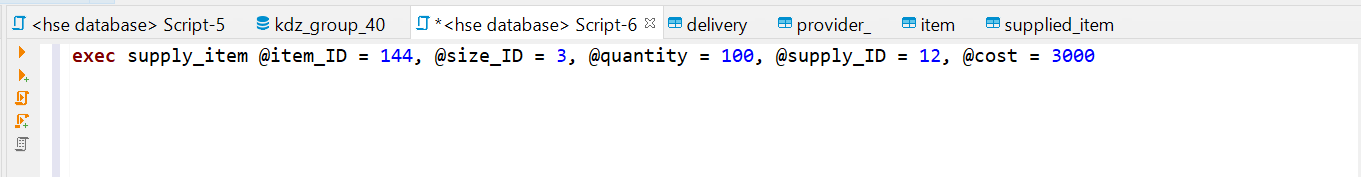


Рисунок 12.2. Запрос с применением процедуры supply\_item

После исполнения запроса были зафиксированы изменения в таблицах supplied\_item, item и transaction\_. В supplied\_item появилась строка с поставленным товаром в размере 100 шт. по 3000 руб. за каждую, в item количество товара 10 увеличилось на 100 шт., а также в transaction\_ появилась запись о списании денежных средств в размере 300 тысяч рублей.

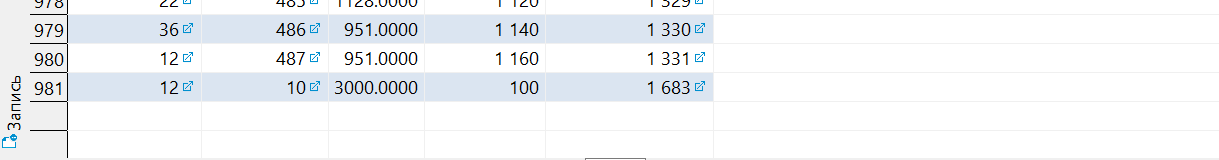


Рисунок 12.3. Результат запроса в таблице supplied\_item

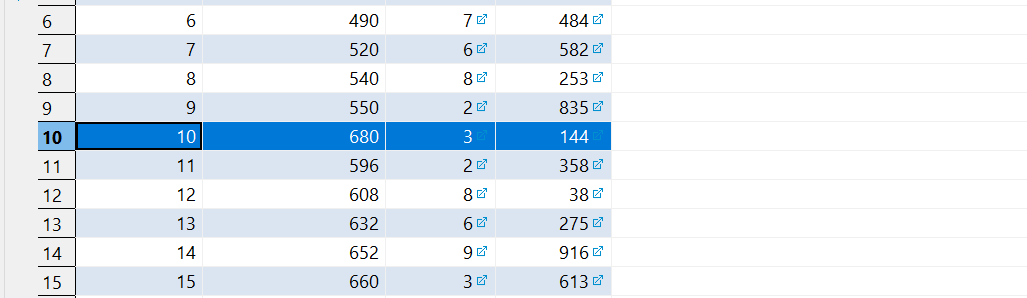


Рисунок 12.4. Результат запроса в таблице item

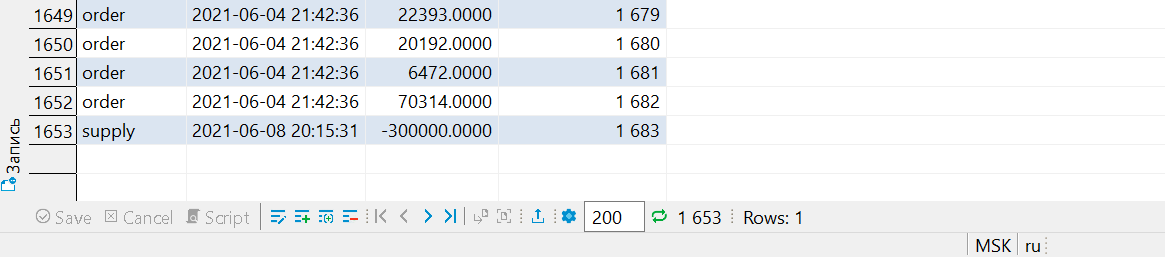


Рисунок 12.5. Результат запроса в таблице transaction\_

1. После оформления заказа необходимо заполнить данные о том, какие в нем товары и в каком они количестве. Функция автоматически вычитает из значения текущего количества в наличии (обновляет таблицу item) количество заказанных товаров, или сообщает об ошибке, если товара меньше, чем в заказе. Также получение оплаты заказа (и время ее осуществления) за каждый поставленный тип товара, рассчитывающаяся как произведение стоимости поставленной единицы товара на количество товара, заносится в таблицу transaction\_.

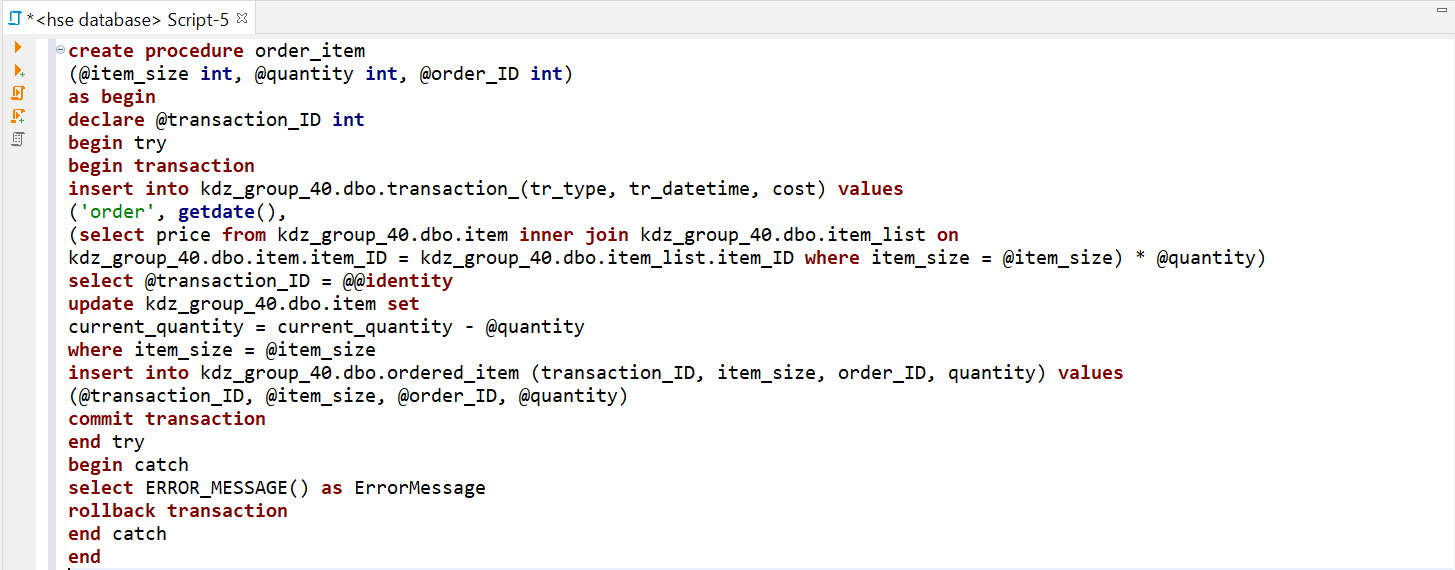


Рисунок 13. Код процедуры order\_item

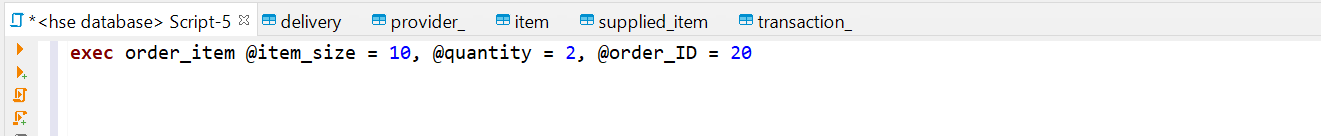


Рисунок 13.1. Запрос с применением процедуры order\_item

После исполнения запроса были зафиксированы изменения в таблицах ordered\_item, item и transaction\_. В ordered\_item появилась строка с заказанным товаром 10 в количестве 2 шт., в item уменьшилось количество товара 10 на 2 шт., в transaction\_ добавилась запись о получении денежных средств в размере 17726 руб.

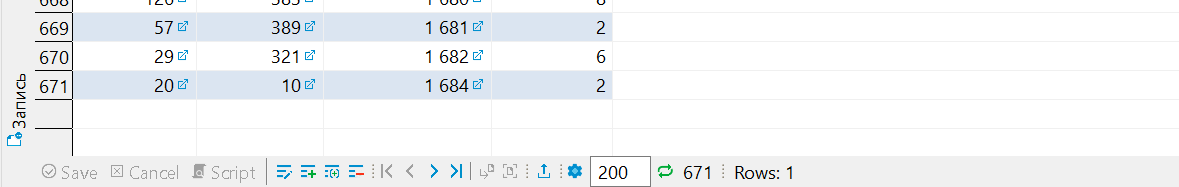


Рисунок 13.2. Результат выполнения запроса в таблице ordered\_item

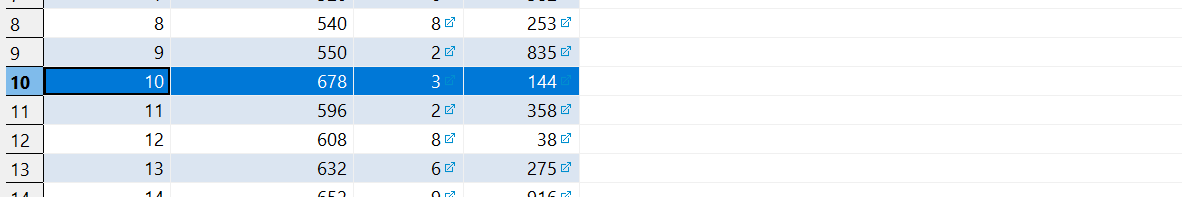


Рисунок 13.3. Результат выполнения запроса в таблице item

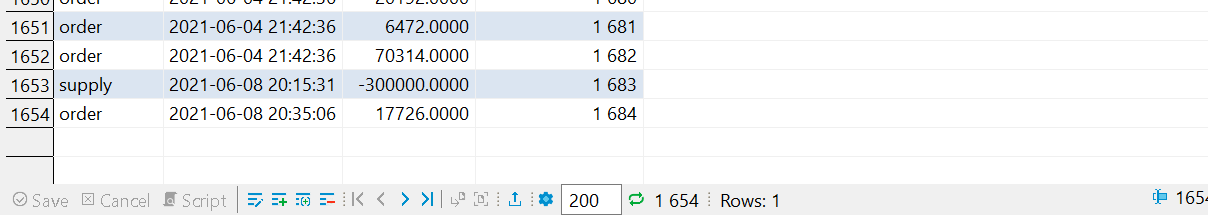


Рисунок 13.4. Результат выполнения запроса в таблице transaction\_

Также была произведена попытка заказа товара в количестве большем, чем имеется в наличии. При попытке выполнения запроса всплывет сообщение об ошибке, которая ссылается на ограничение more\_than\_zero, проверяющее текущее количество в наличии, которое не может быть отрицательным.

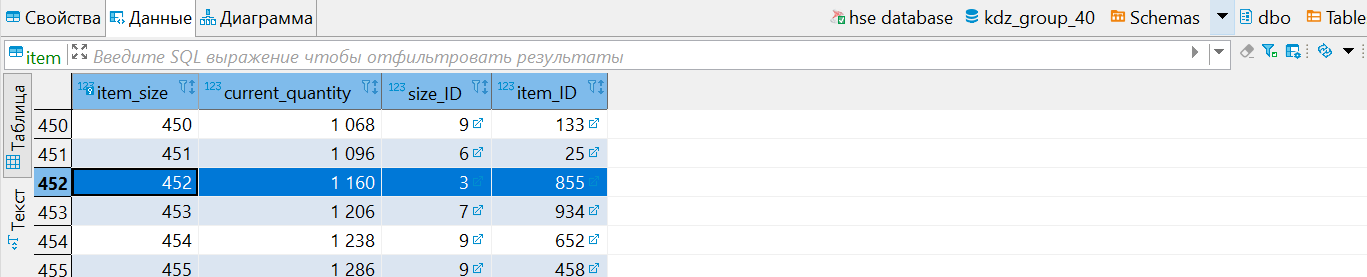


Рисунок 13.5. Вид строк таблицы item

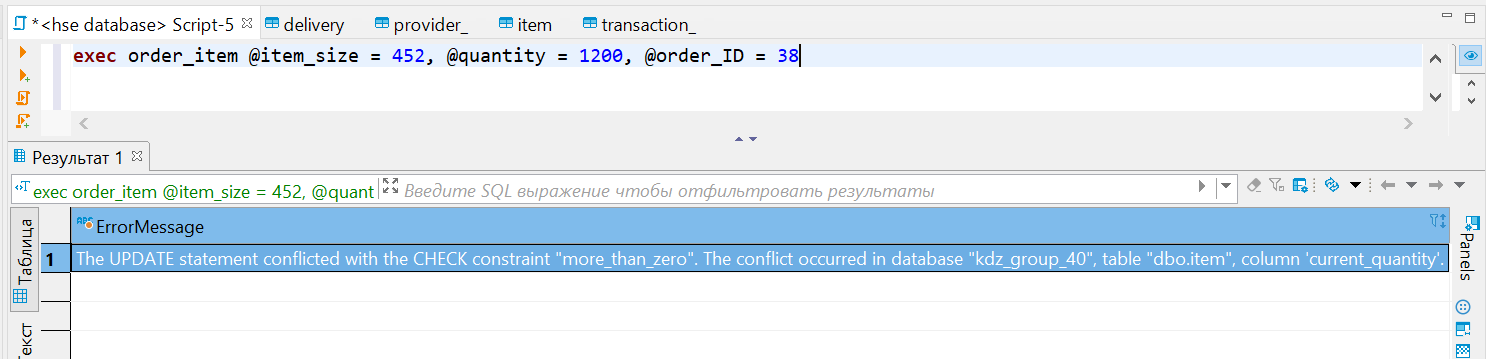


Рисунок 13.6. Пример невыполненного запроса

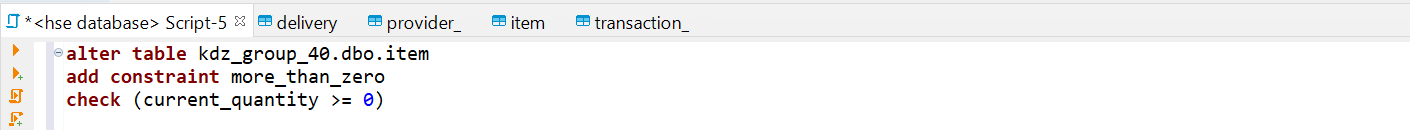


Рисунок 13.7. Код ограничения more\_than\_zero

1. Нередко происходит так, что клиенту может не понравиться качество/фасон/цвет вещи и он захочет ее вернуть. В данном случае при запуске процедуры с входным параметром конкретного товара из конкретного заказа значение количества товара в наличии вновь увеличится на количество, прописанное в заказе. Произойдет удаление строки заказанной вещи, а в таблице transaction\_ появится новая запись о возврате денег клиенту.



Рисунок 14. Код процедуры ticket\_return

Для показа работы процедуры был создан запрос, входным параметром которого является заказанный товар, занесенный в базу на прошлом шаге.

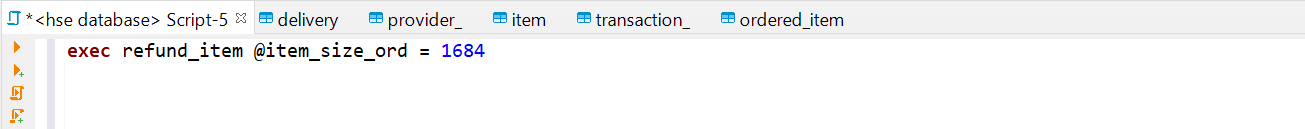
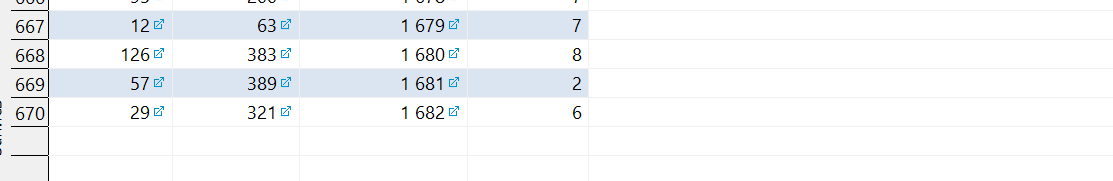


Рисунок 14.1. Запрос с использованием процедуры refund\_item

После исполнения запроса были зафиксированы изменения в таблицах item и transaction\_. Появилась новая строка с типом транзакции refund и суммой денег равной стоимости проданного товара, записи указанного в процедуре товара в таблице ordered\_item нет, в таблице item текущее количество стало прежним.

*Рисунок 14.2. Результат выполнения запроса в таблице ordered\_item*

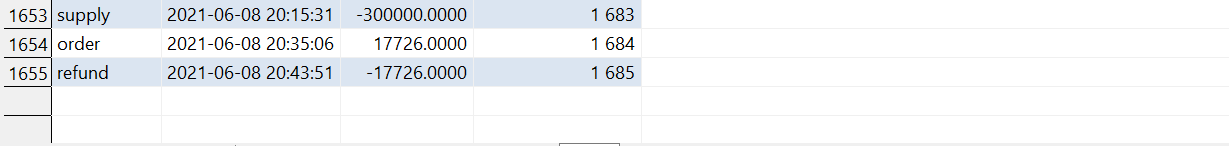


Рисунок 14.3. Результат выполнения запроса в таблице transaction\_

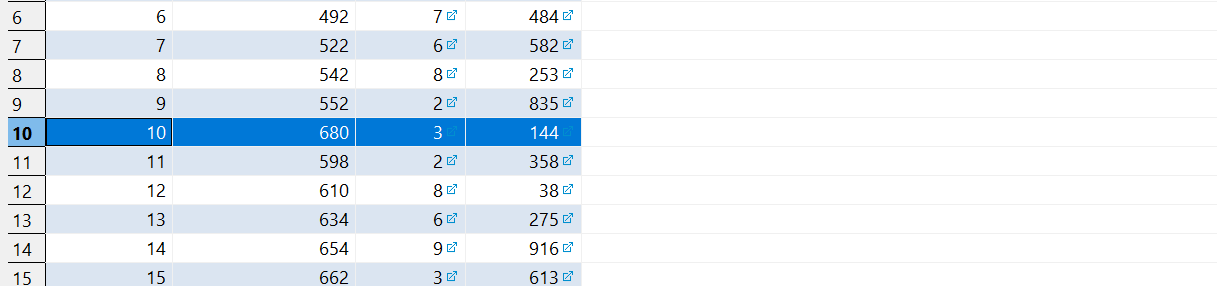


Рисунок 14.4. Результат выполнения запроса в таблице item

1. Процедура create\_del помогает корректно оформить доставку, при занесении всех необходимых данных в ней с помощью обозначенной ранее функции define\_ storage определяется склад, с которого доставка будет производиться. Это облегчает работу пользователя и автоматически определяет оптимальный вариант в выборе склада.

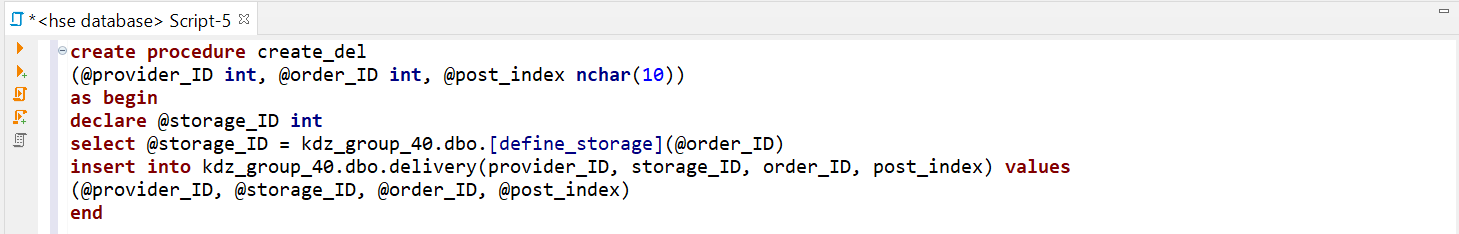


Рисунок 15. Код процедуры create\_del

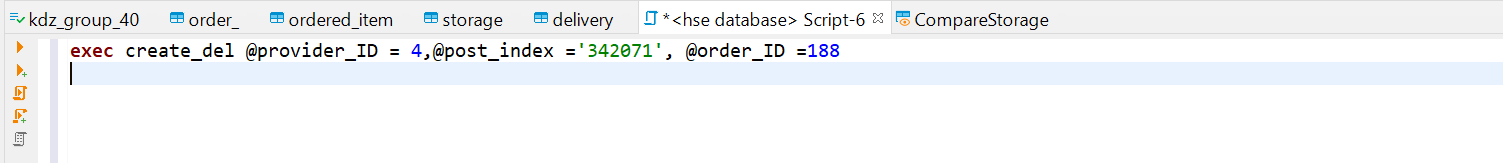


Рисунок 15.1. Запрос с применением процедуры create\_del

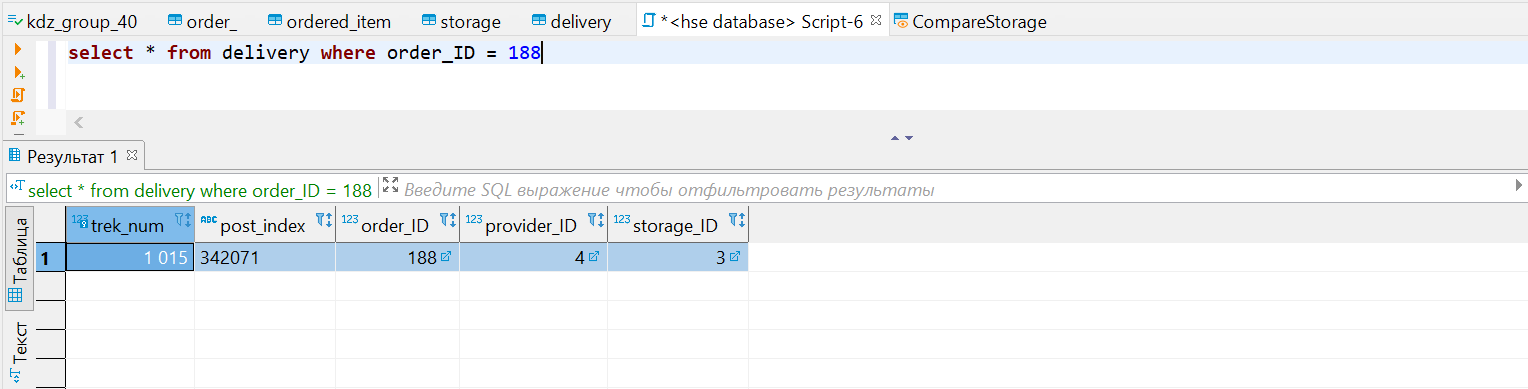


Рисунок 15.2. Результат выполнения запроса в таблице delivery

1. Если же пользователь видит, что товар определенного типа не обладает достаточной маржинальностью, то может завести в параметры процедуры change\_price этот товар с желаемой им значением прибыльности. Процедура обновит значение атрибута price таблицы item\_list, которое теперь будет удовлетворять значению маржинальности.

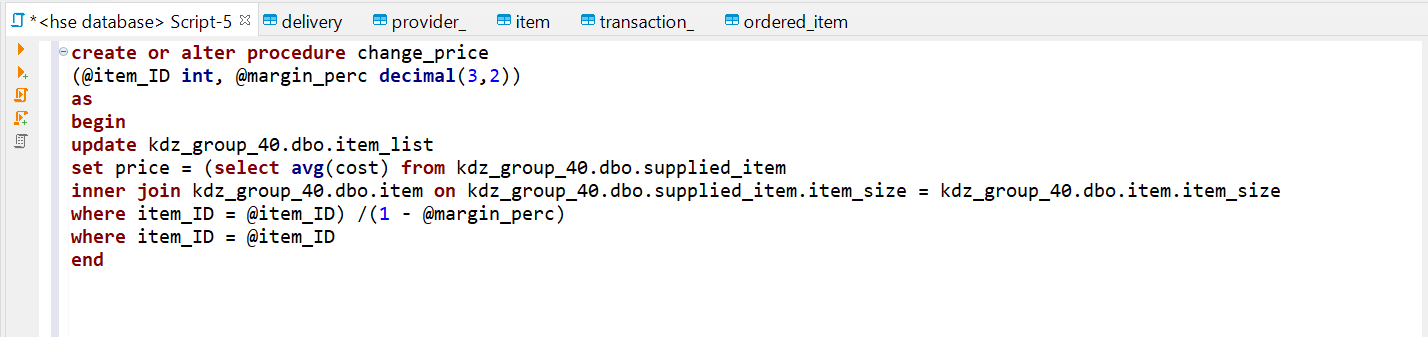


Рисунок 16. Код процедуры change\_price

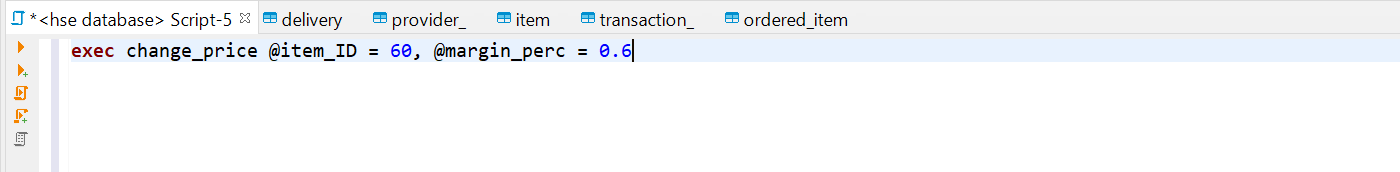


Рисунок 16.1. Запрос с применением процедуры change price

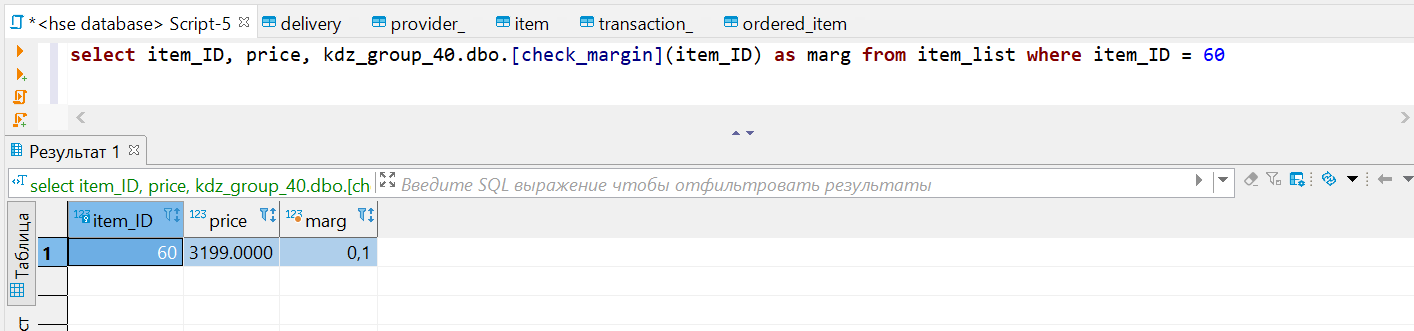


Рисунок 16.2. Строка таблицы до выполнения запроса

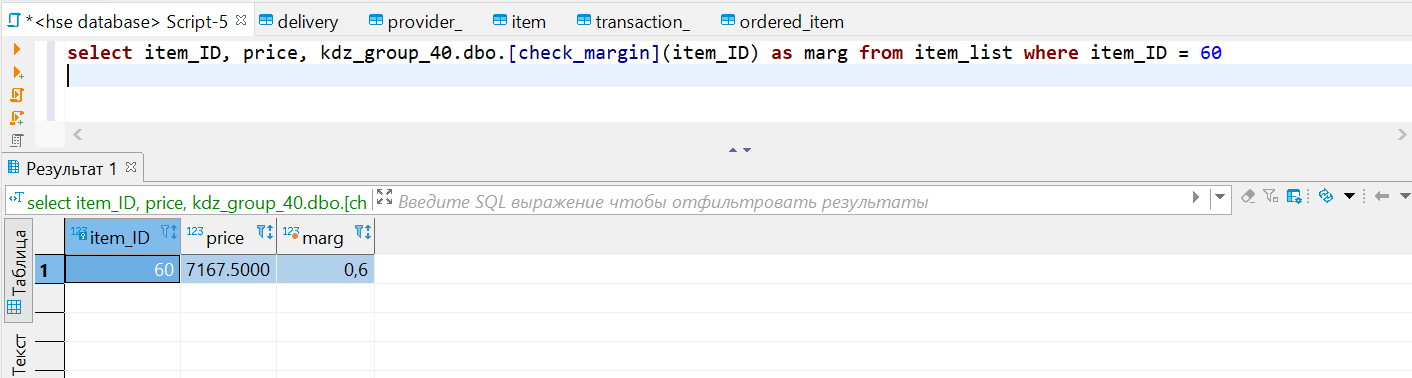


Рисунок 16.3. Строка таблицы после выполнения запроса

## Запросы

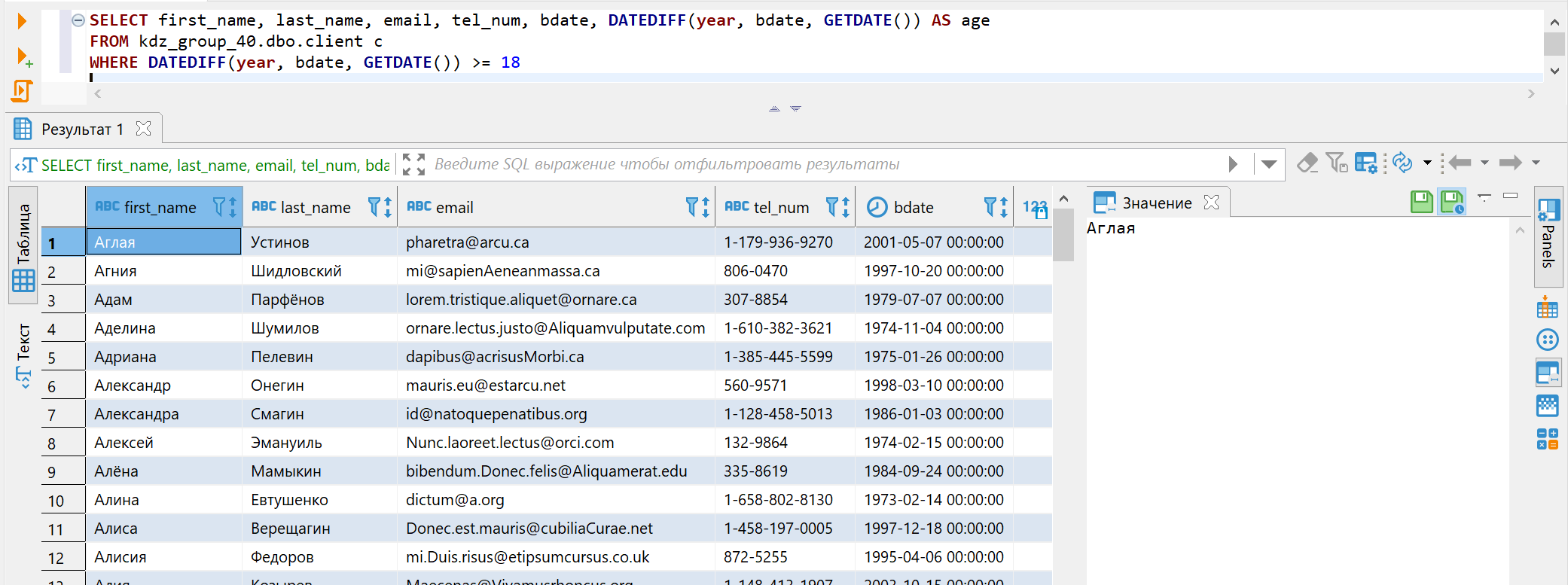
Итак, после реализации базы данных нам необходимо написать запросы, которые могут быть полезными для аналитика, использующего базу данных.

1. Простой запрос с условием и формулами в SELECT.
   1. Вывод оставшегося количества конкретного товара.



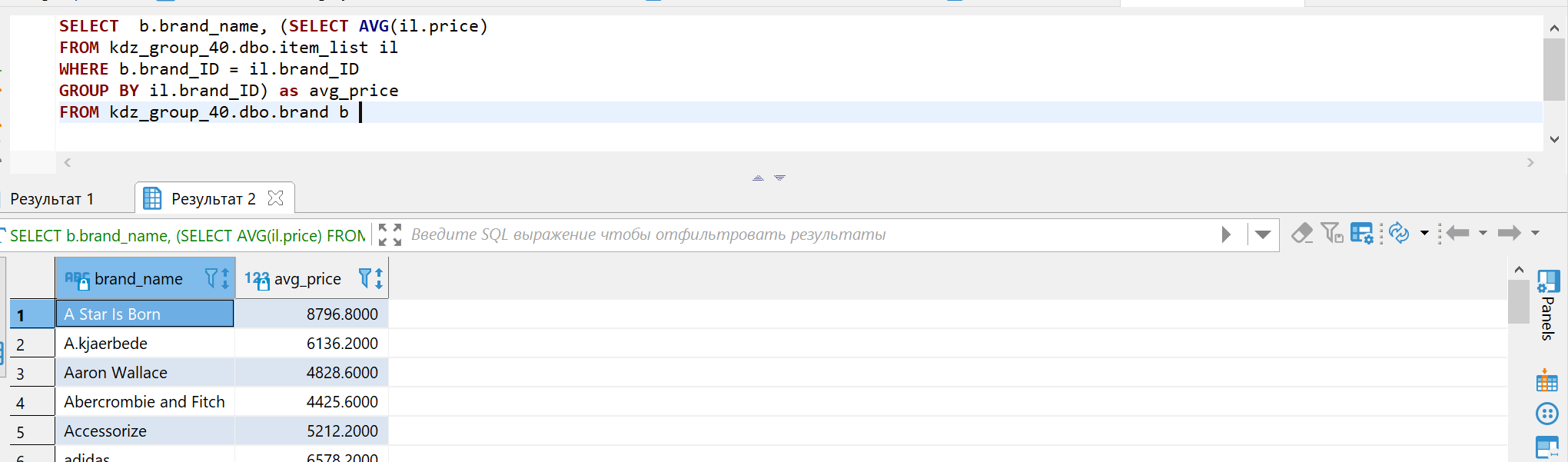
*Рисунок 17. Результат запроса 1.1.*

* 1. Вывод совершеннолетних пользователей.



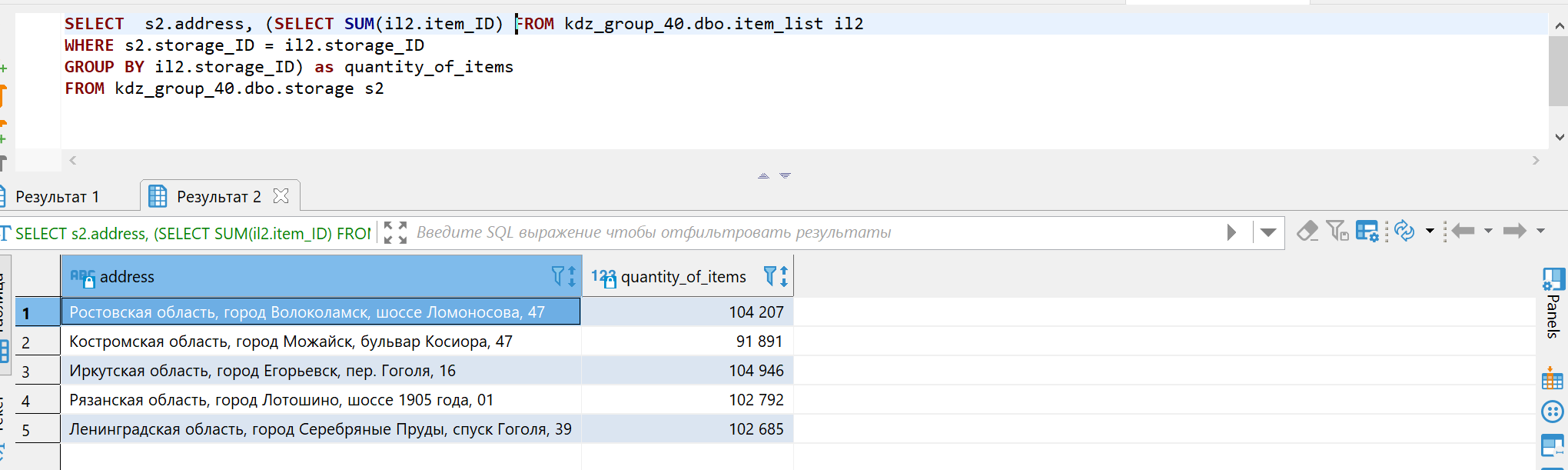
*Рисунок 17.1. Результат запроса 1.2.*

1. Запрос с коррелированным подзапросом в SELECT.
   1. Средняя цена товаров по брендам.



*Рисунок 18. Результат запроса 2.1.*

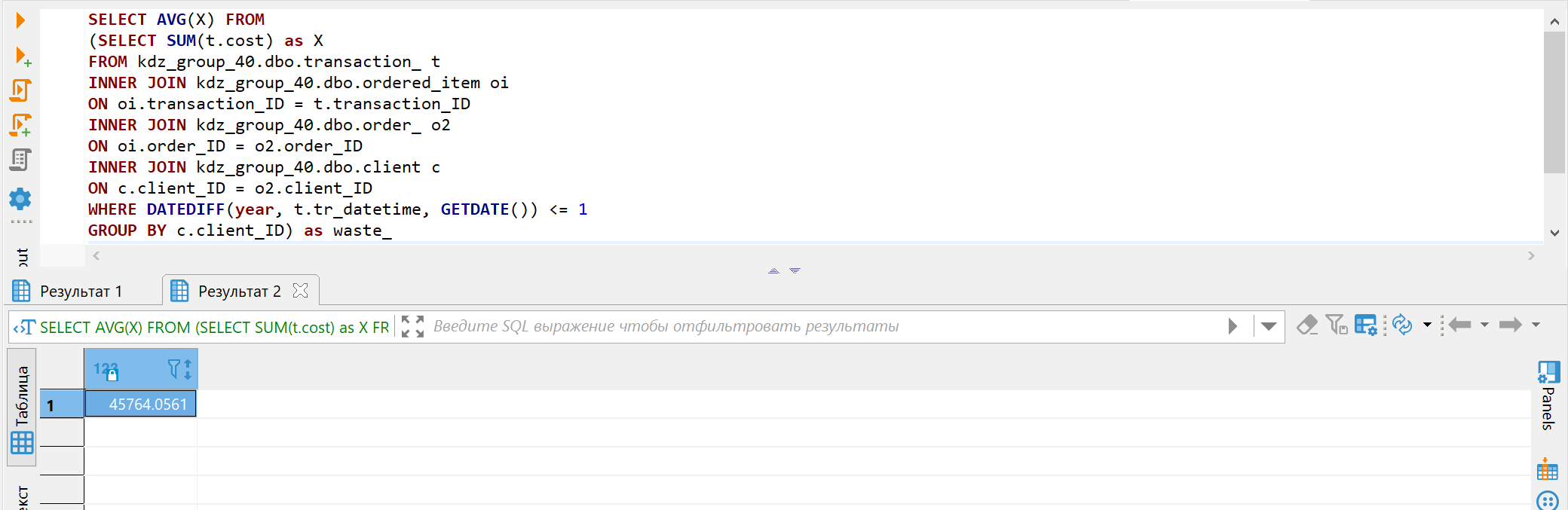
* 1. Количество товаров в складах.



*Рисунок 18.1. Результат запроса 2.2.*

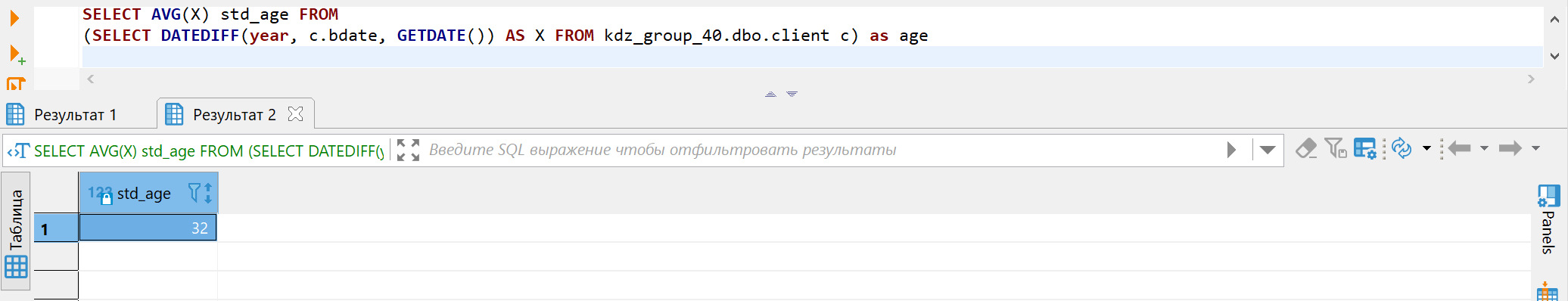
3. Запрос с подзапросом в FROM.

3.1. Сколько в среднем потратили покупатели за год.



*Рисунок 19. Результат запроса 3.1.*

* 1. Средний возраст покупателей.



*Рисунок 19.1. Результат запроса 3.2.*

1. Запрос с использованием LAG LEAD.
   1. Сравнение стоимости следующей и предыдущей стоимости поставки.

Изображение выглядит как текст

Автоматически созданное описание

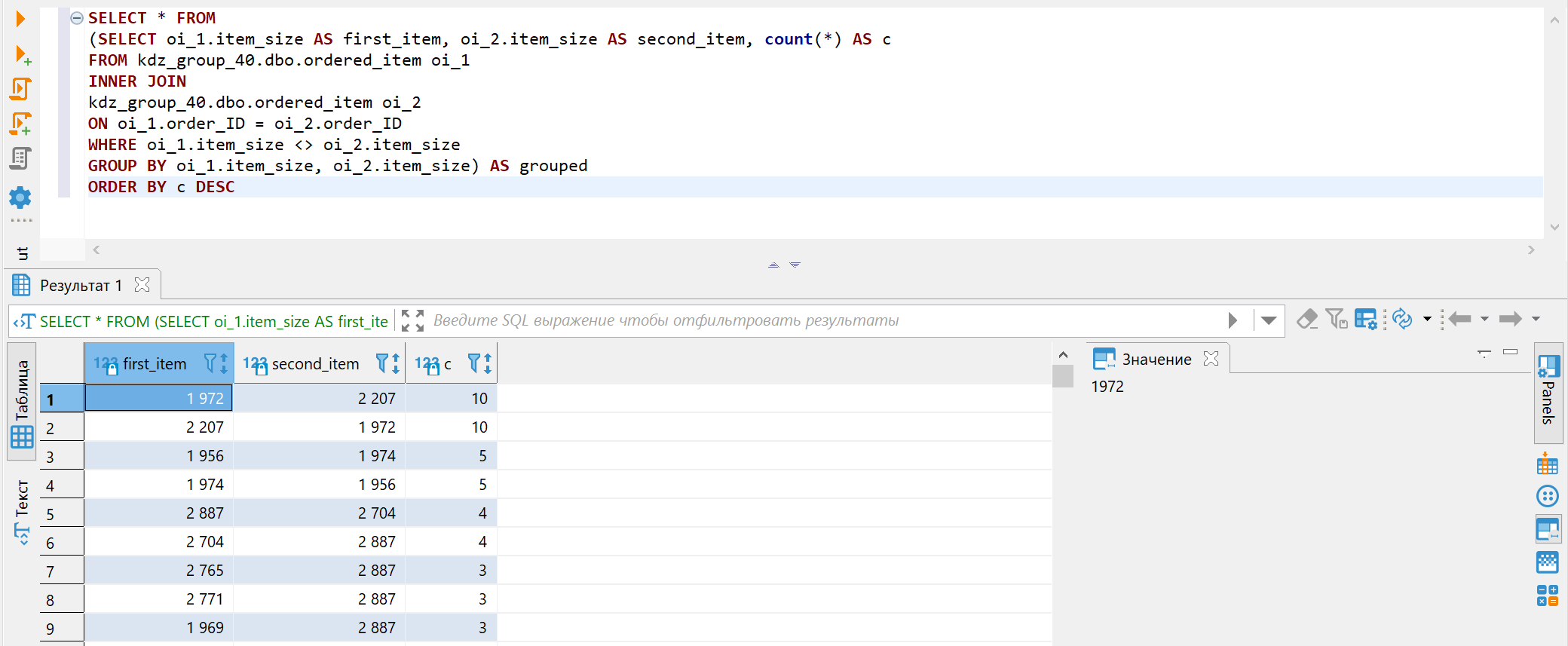
*Рисунок 20. Результат запроса 4.1.*

1. Запрос с подзапросом в FROM, агрегированием, группировкой и сортировкой.
   1. Вывод самых частых перевозок товаров со склада на другой склад для доставки

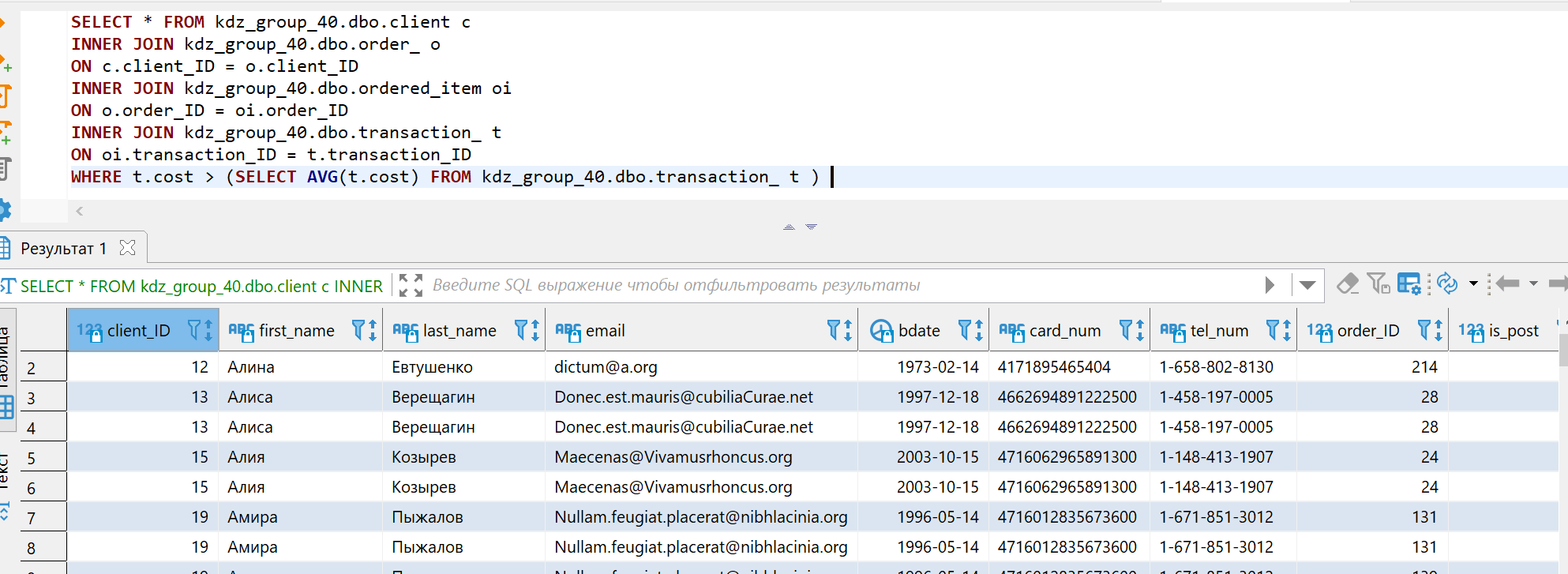


*Рисунок 21. Результат запроса 5.1.*

* 1. Вывод товаров, которые наиболее часто сопутствуют искомому товару в заказах. Поможет для рекомендаций на сайте по принципу «С этим товаром также покупают».

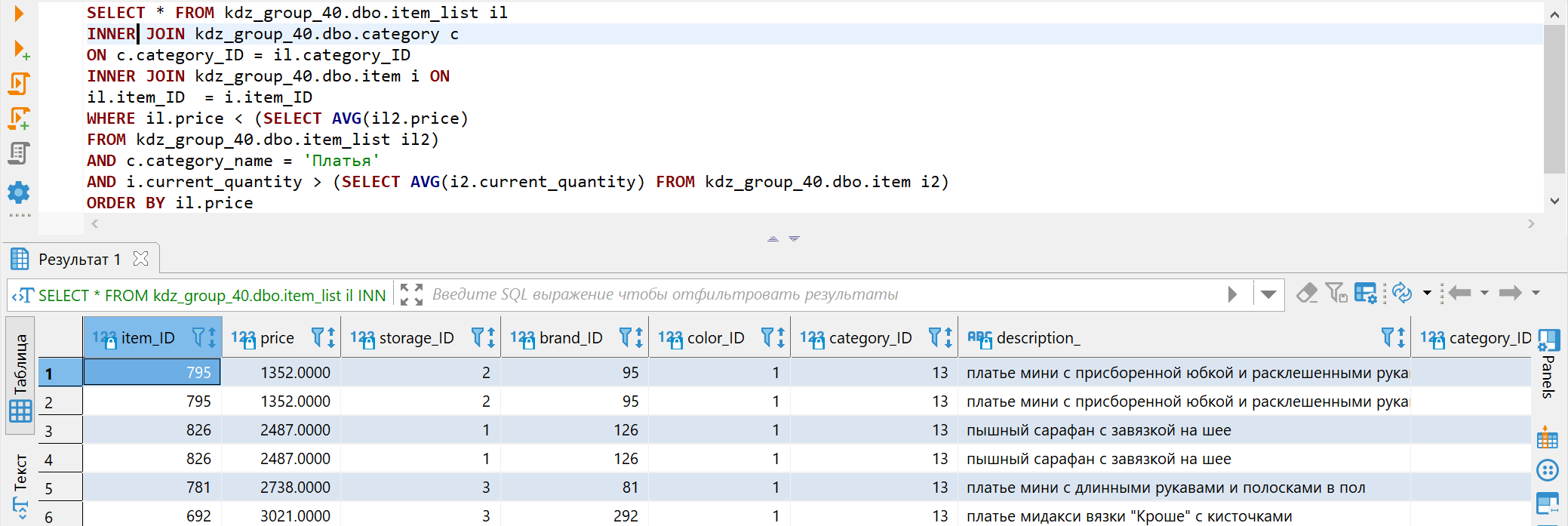
  
*Рисунок 21.1 Результат запроса 5.2.*

1. Запрос с коррелированным подзапросом в WHERE.
   1. Вывод клиентов и списка их заказов со стоимостью выше средней.



*Рисунок 22. Результат запроса 6.1.*

* 1. Вывод товаров конкретной категории, с ценой ниже средней и большим запасом на складе



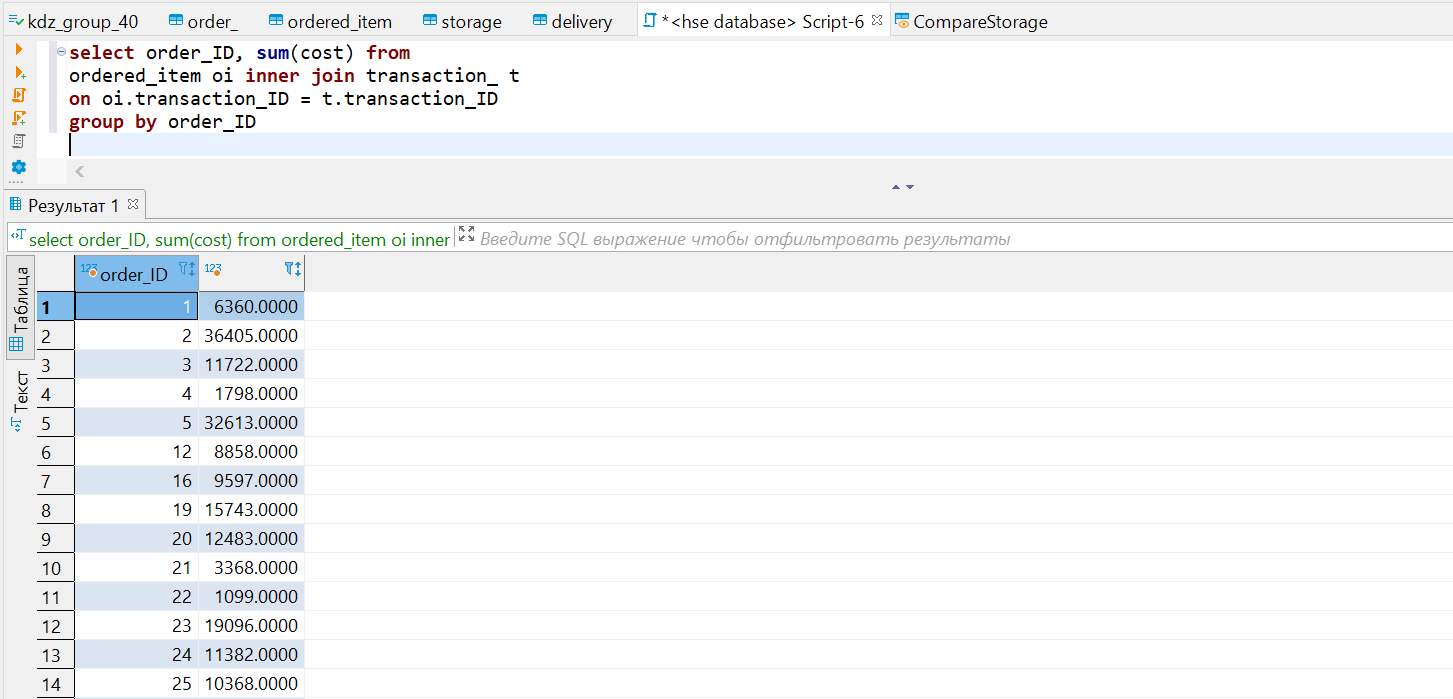
*Рисунок 22.1. Результат запроса 6.2.*

1. Запрос с агрегированием и выражением JOIN, включающим не менее 2 таблиц
   1. Подсчет количества продаж конкретной категории



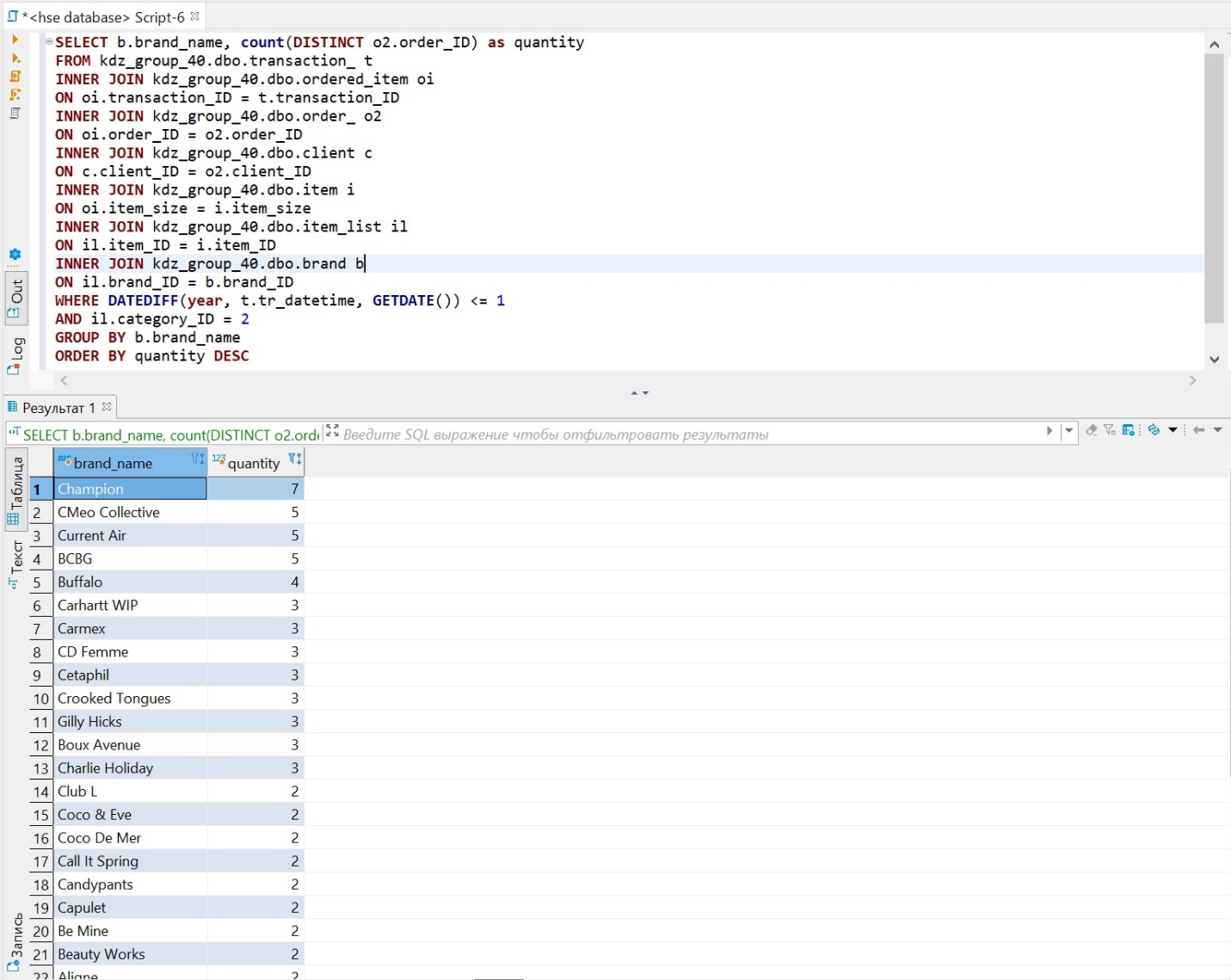
*Рисунок 23. Результат запроса 7.1.*

* 1. Вывод конечной суммы заказов.



*Рисунок 23.1. Результат запроса 7.2.*

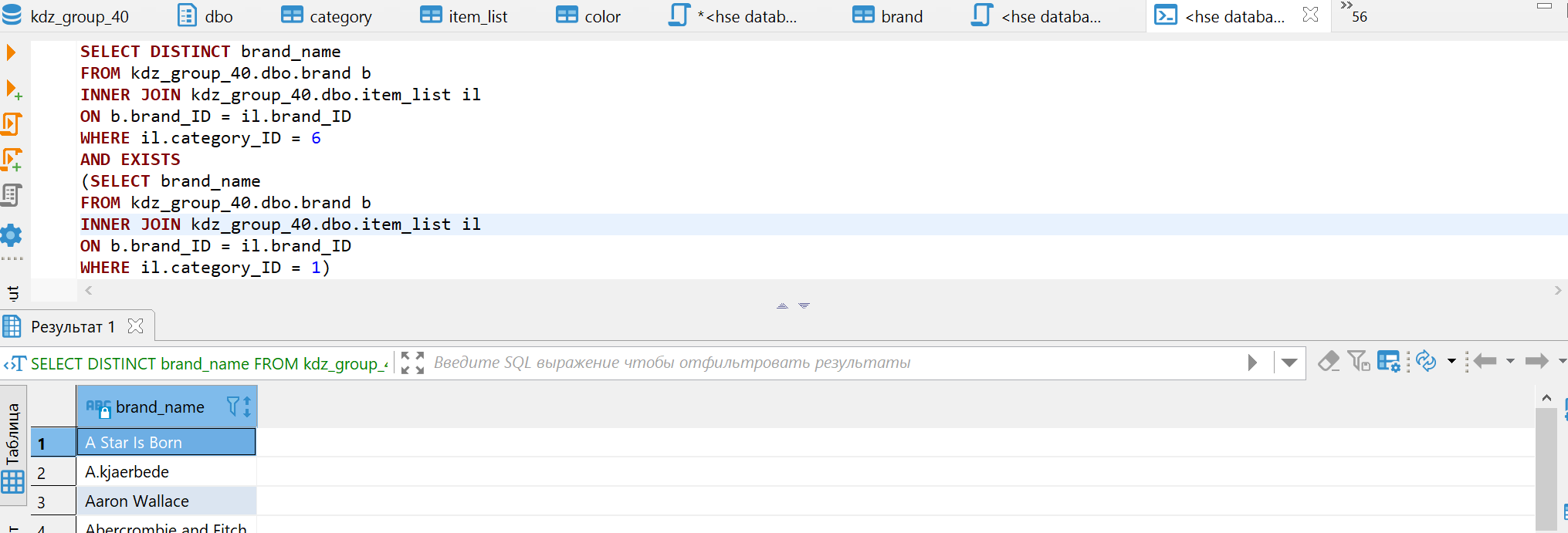
* 1. Количество заказов категории по брендам за год



*Рисунок 23.2. Результат запроса 7.3.*

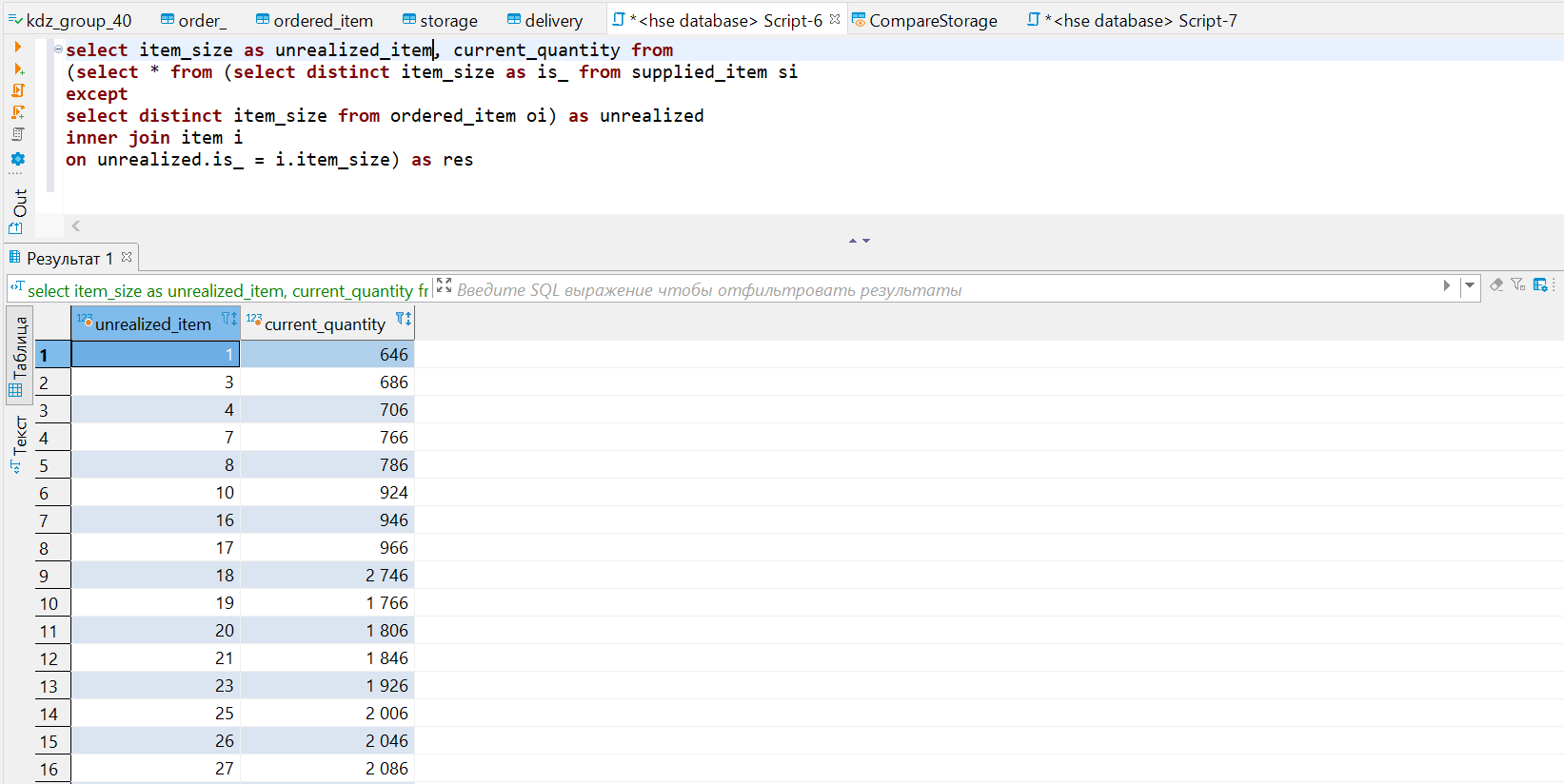
1. Запрос с EXISTS.

8.1 Вывод названия брендов, которые производят товары конкретных категорий



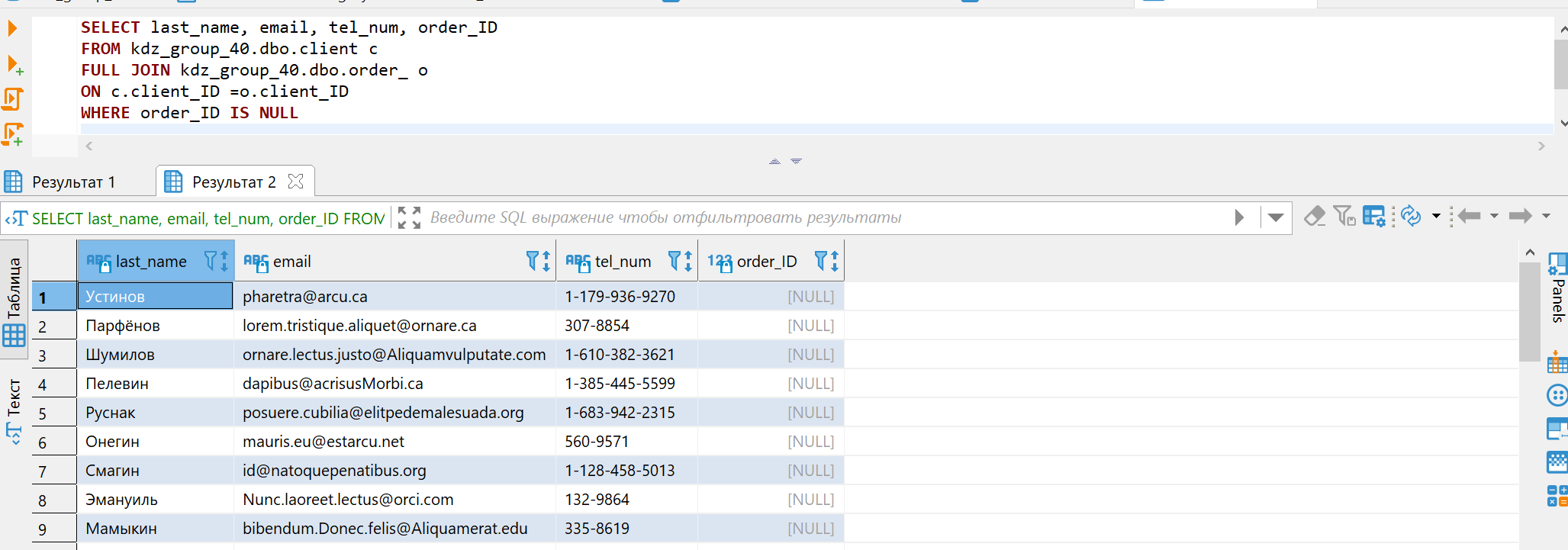
*Рисунок 24. Результат запроса 8.1.*

1. Запрос, использующий манипуляции с множествами.
   1. Вывод нереализованных товаров и их количество в наличии.



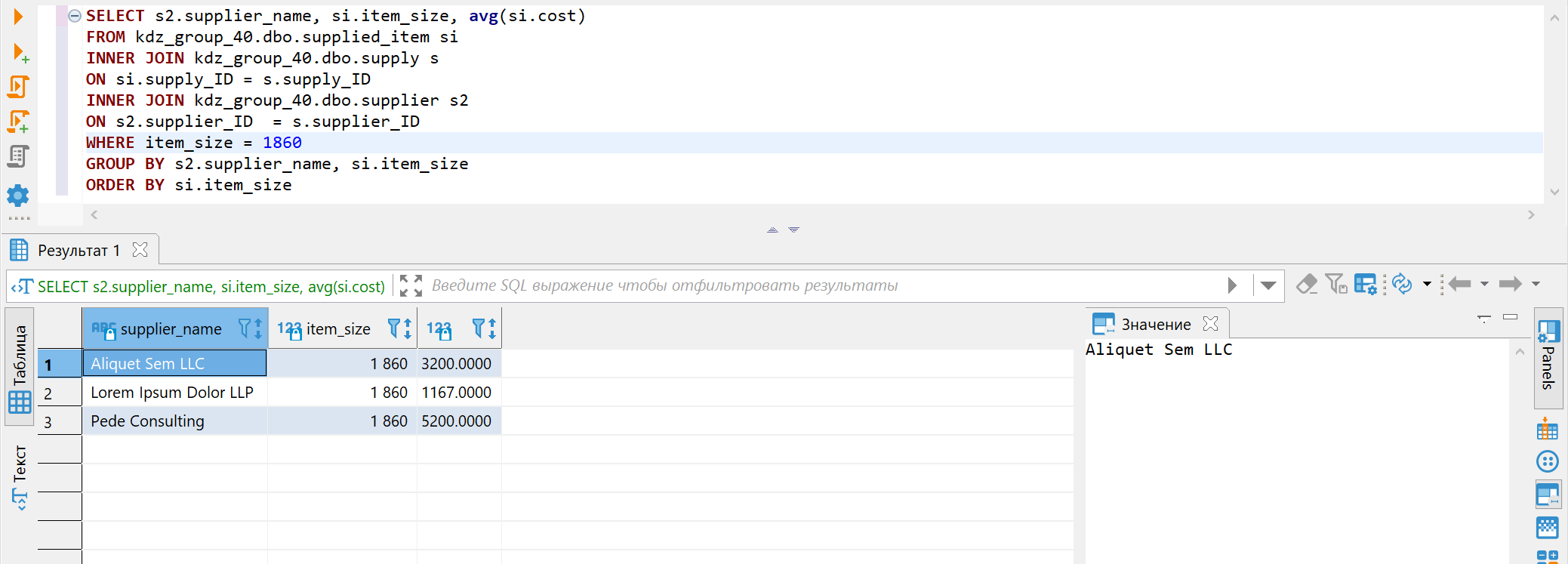
*Рисунок 25. Результат запроса 9.1.*

1. Запрос с внешним соединением и проверкой на наличие NULL
   1. Вывод всех клиентов, которые еще не оформили ни одного заказа.



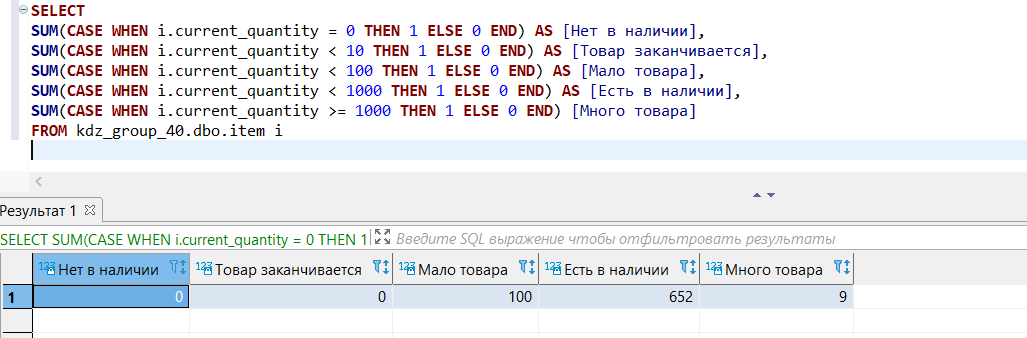
*Рисунок 26. Результат запроса 10.1.*

1. Запрос с агрегированием и выражением JOIN, включающим не менее 3 таблиц.
   1. Средние цены закупок одного и того же товара от разных поставщиков для нахождения более экономного варианта снабжения.



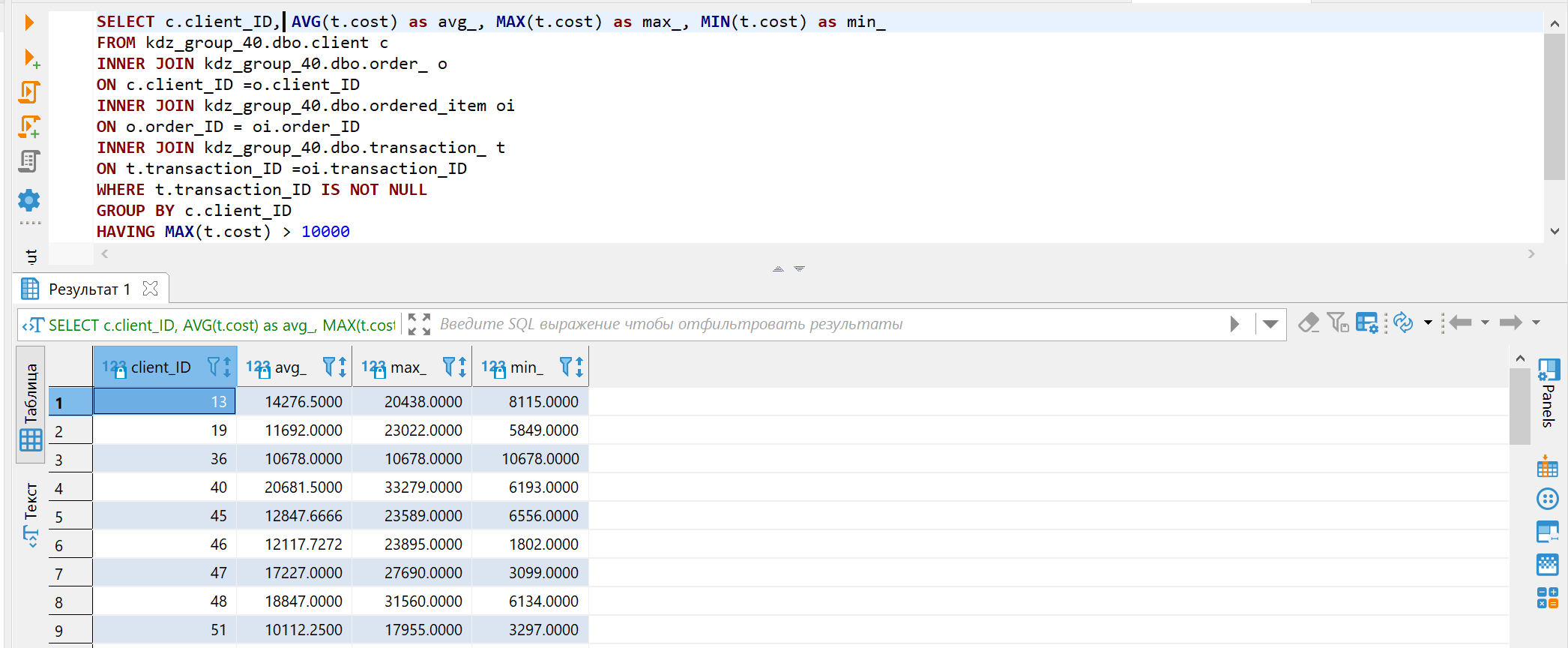
*Рисунок 27. Результат запроса 11.1.*

1. Запрос с CASE и агрегированием.
   1. Подсчет количества товара.



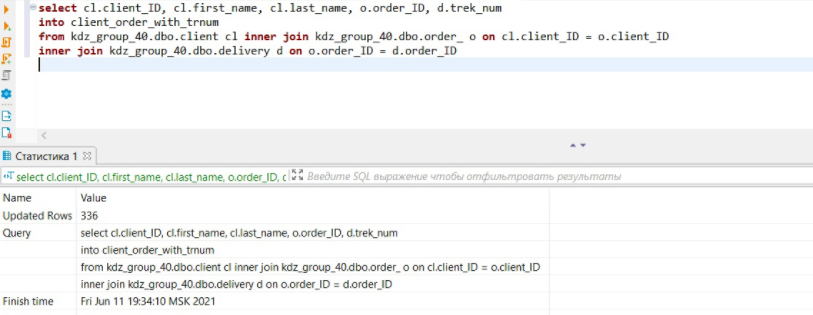
*Рисунок 28. Результат запроса 12.1.*

1. Запрос с HAVING и агрегированием.
   1. Вывод клиентов с заказами, выше определенной суммы.



*Рисунок 29. Результат запроса 13.1.*

1. Запрос SELECT INTO.
   1. Вывод клиентов с номером заказа и номером отслеживания заказа.



*Рисунок 30. SQL код запроса 14.1.*



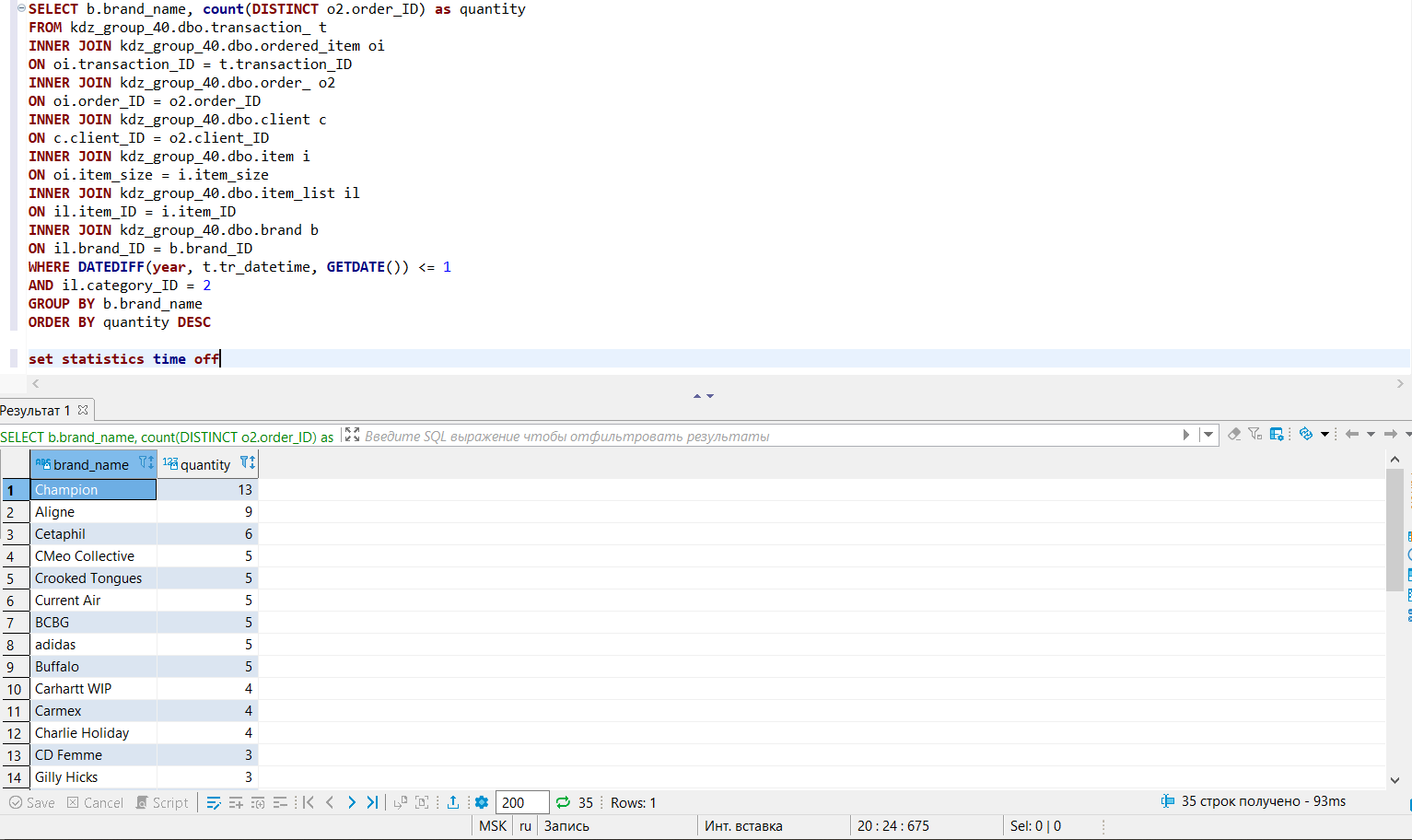
*Рисунок 30.1. Результат запроса 14.1.*

## Индекс

Чтобы ускорить время работы некоторых запросов, в таблицы, которые не часто обновляются и не требуют частых вставок, можно добавить индексы.

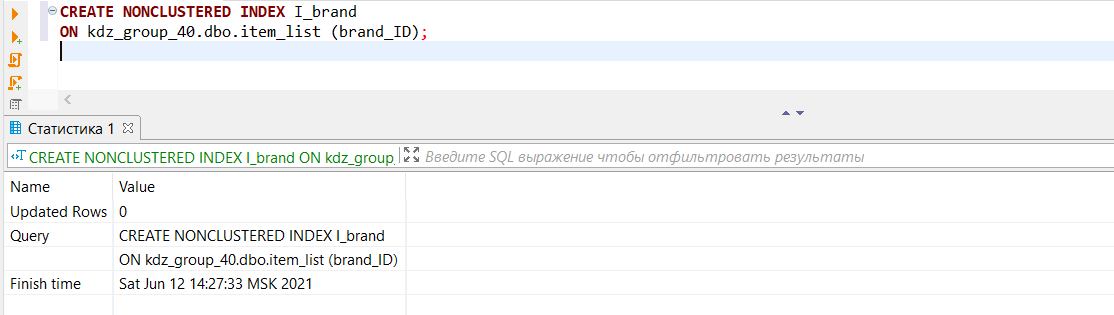
Запрос, где присутствует много объединений таблиц, будет выполнятся долго, поэтому добавим индекс туда.

Возьмем запрос, показывающий количество заказов категории по брендам за год, и посмотрим на время выполнения запроса до добавления индекса:



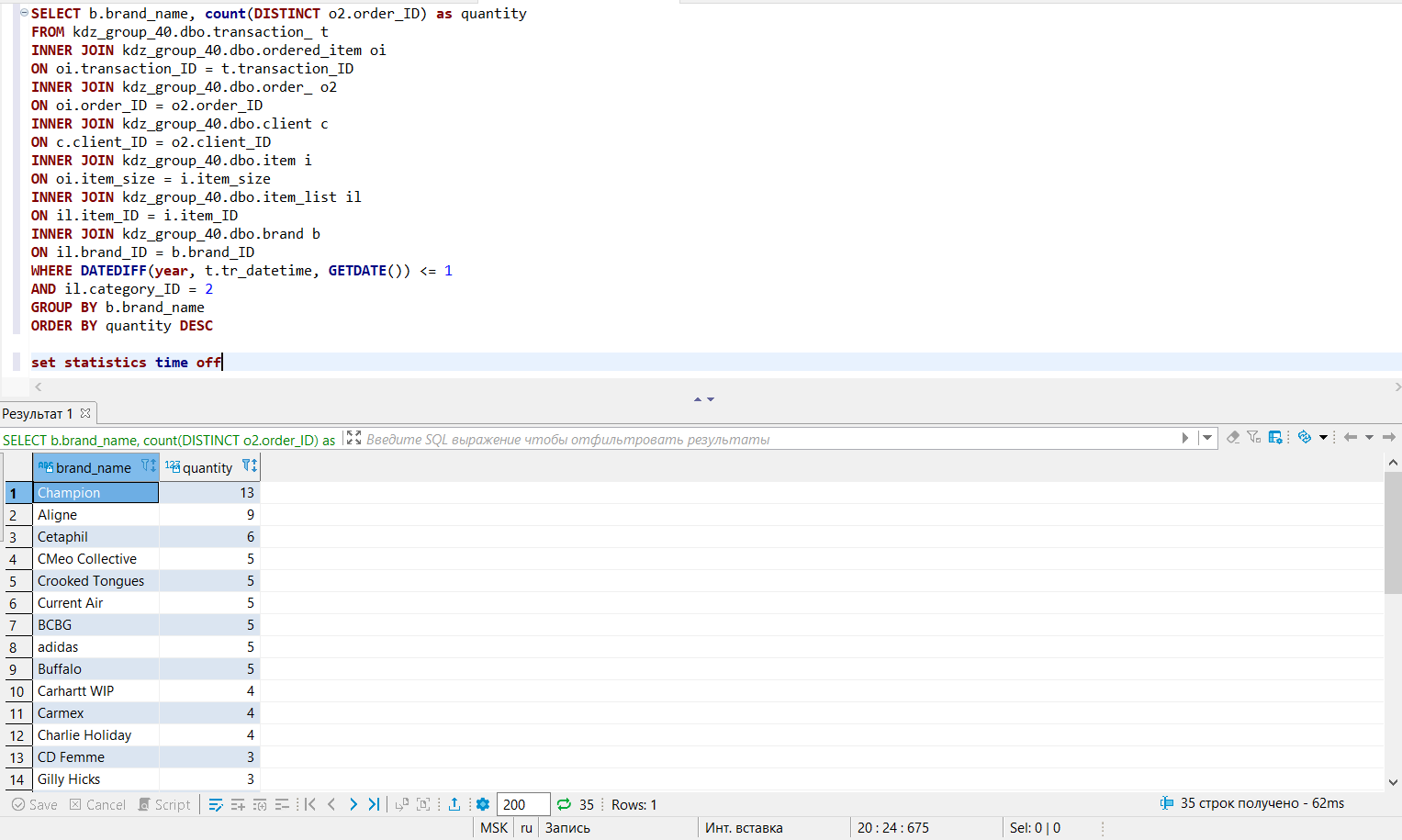
*Рисунок 31. Проверка времени исполнения запроса до создания индекса*

Далее создадим индекс для поля brand\_ID таблицы item\_list:



*Рисунок 31.1. Создание индекса*

Снова проверяем время работы запроса и видим, что оно сократилось с 93 ms до 62:



*Рисунок 31.2. Проверка времени после запроса после создания индекса*

# Отчетные формы Power BI и Excel

## Отчеты в Power BI

В качестве основного отчета был выбран запрос с выводом суммы всех затрат на приобретение и дохода от реализации по товару конкретного типа. Благодаря нему можно сразу понять окупается ли вложение в этот товар. Таблица обновляется сразу при открытии данного отчета.

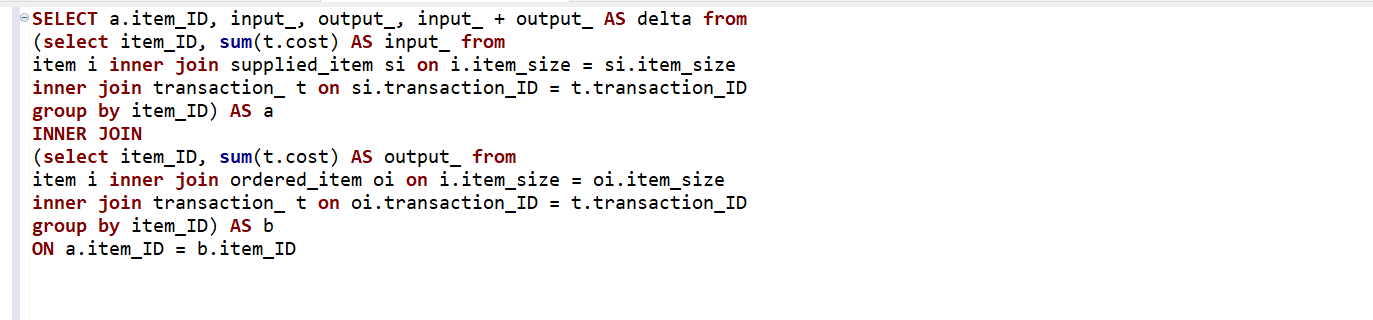


Рисунок 32. Код запроса для основного отчета

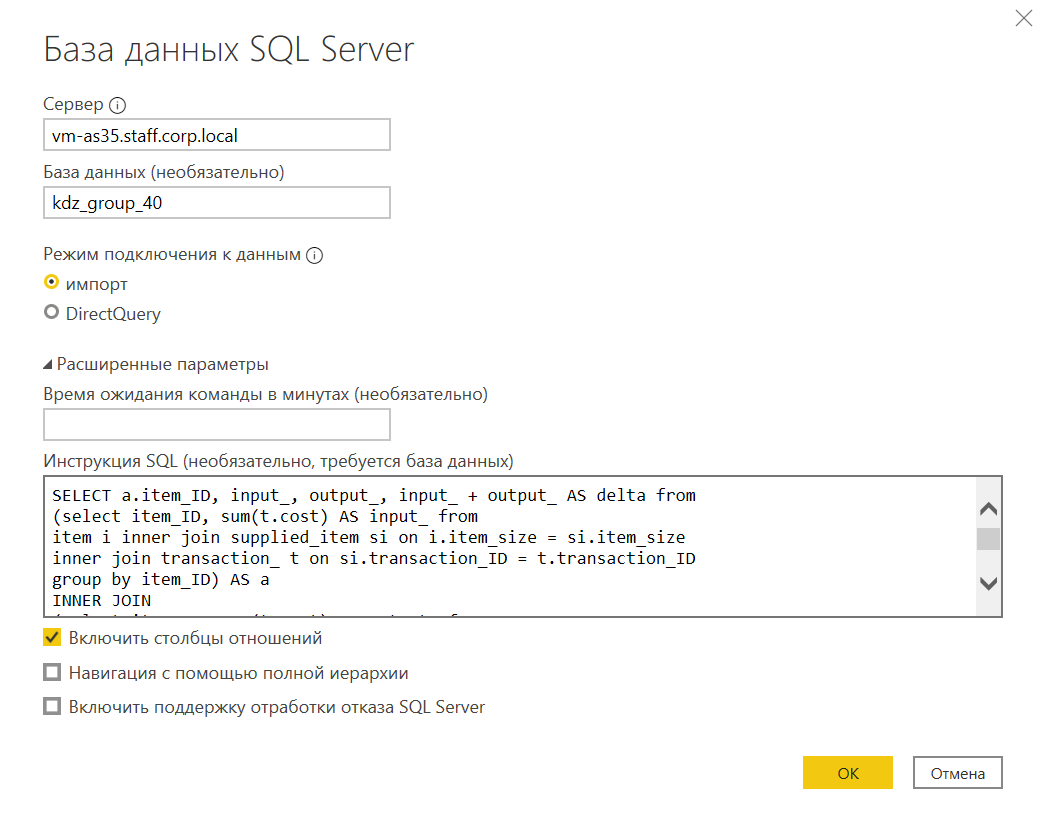
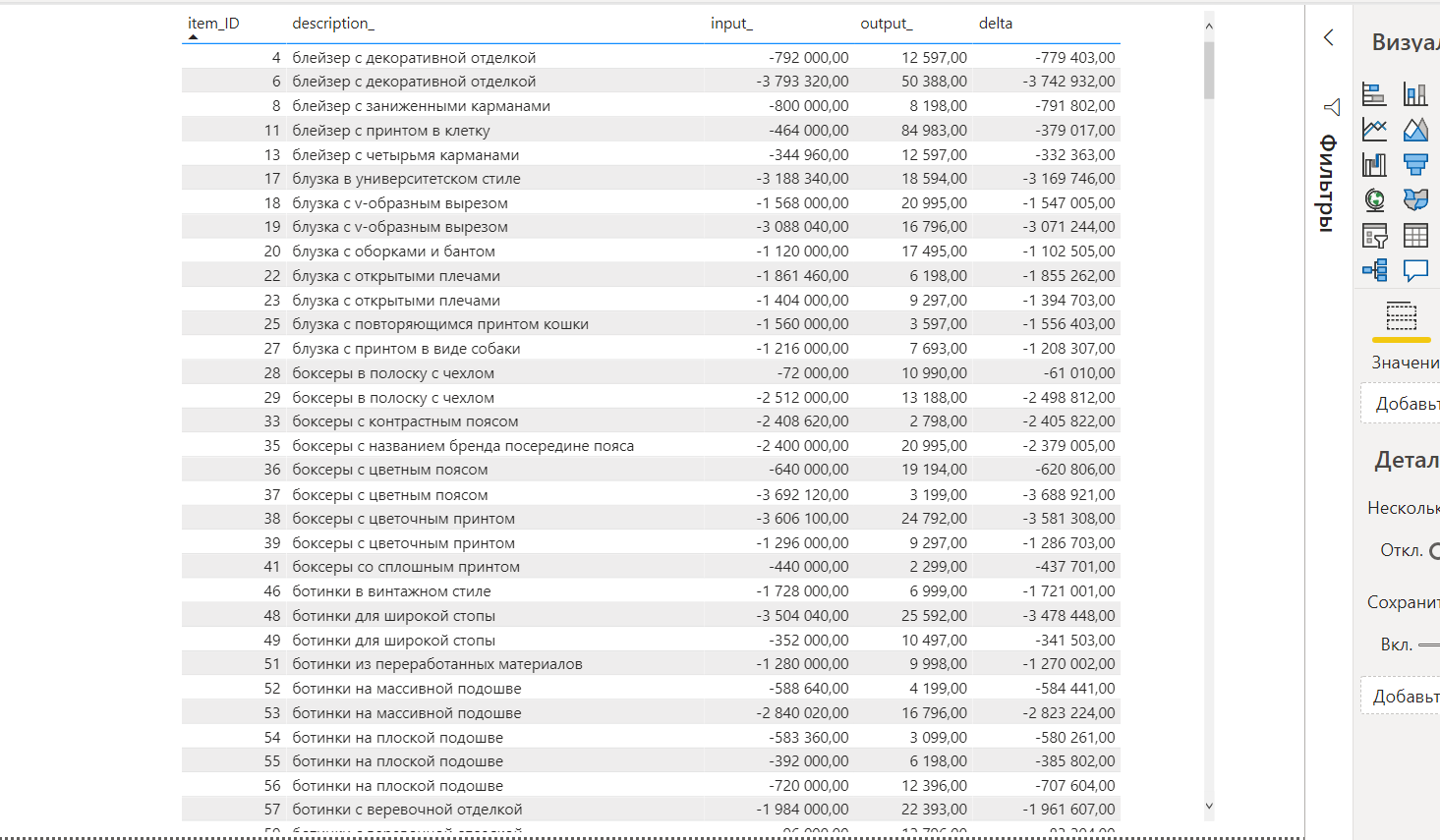


Рисунок 33. Инструкция при открытии отчета с использованием запроса



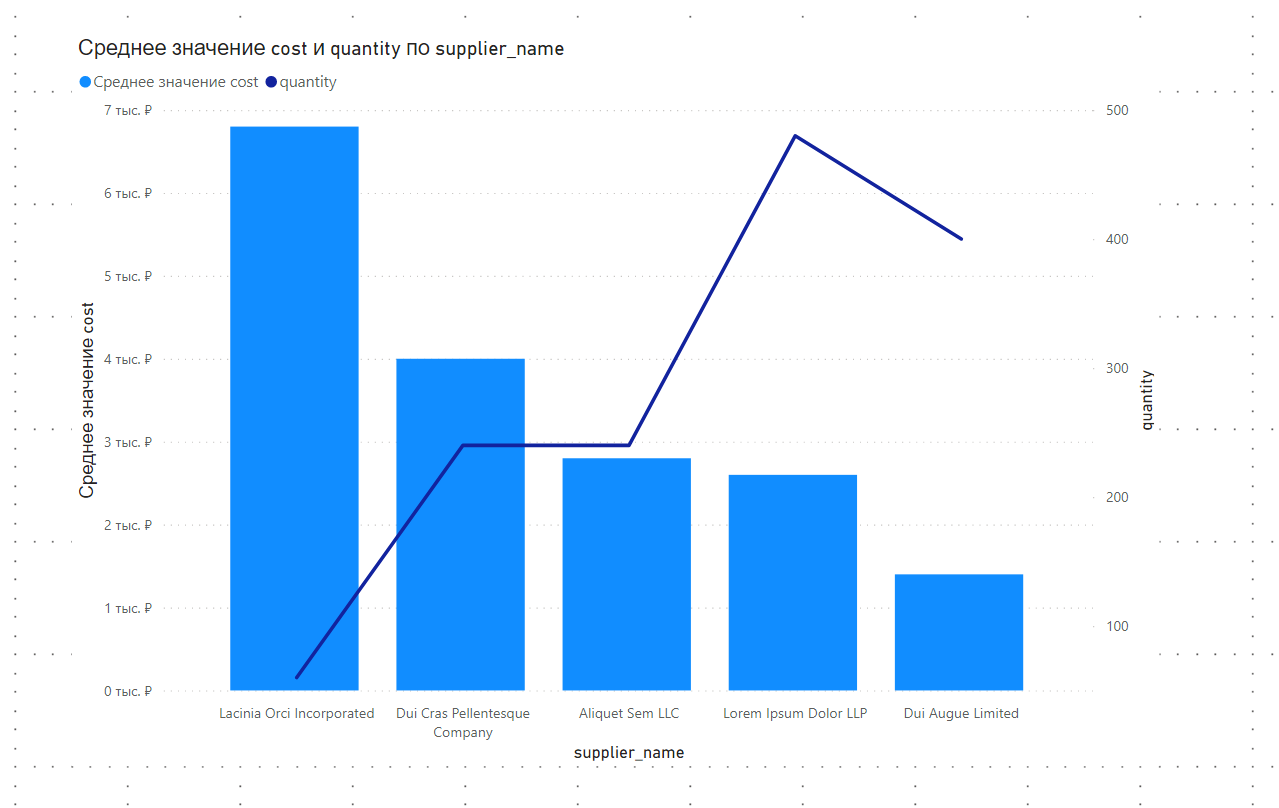
*Рисунок 34. Отчет о суммарных затратах на приобретение товаров и выручке от них*

При выборе строчки с интересующим нас товаром появляется возможность детализировать отчет под названием «Supply info», предоставляющая ценную для аналитиков информацию о поставщиках.



*Рисунок 35. Пример получения детализации*

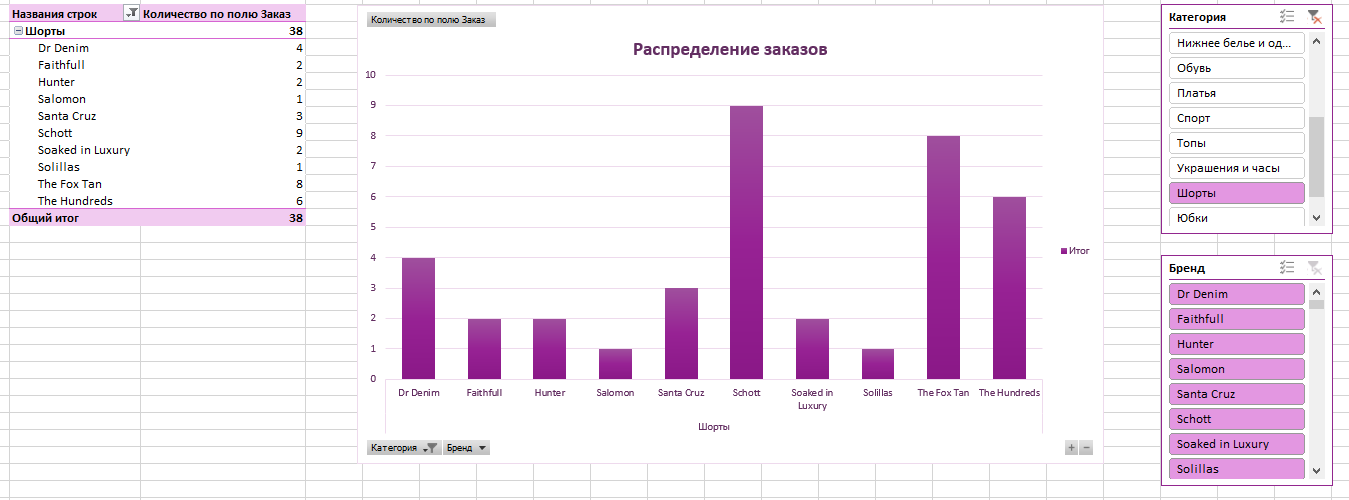
После нажатия на «Supply Info» появляется визуализирующий данные график. Столбцы соответствуют средней закупочной цене выбранного товара от поставщиков, линия показывает объемы закупок товаров от тех же поставщиков. Таким образом, детализированный отчет помогает принять решение о корректировке сотрудничества с поставщиками, т. к. большие объемы закупок товара по высокой цене (при возможности закупить тот же товар дешевле у другого поставщика) не желательны.



*Рисунок 36. Пример детализированного отчета по поставкам*

## Отчет в MS Excel

Информационная панель в Excel с графиком и "срезами".



*Рисунок 37. Сводная таблица Excel*

Для визуализации данных мы построили сводную таблицу заказов по брендам и категориям, чтобы понять, какие бренды чаще всего покупают в выбранной категории.

Настроили срезы по категории и бренду, а также вывели диаграмму для наглядности.

# Описание роли участников проекта

Обязанности в нашем проекте были распределены равномерно, каждая участница внесла свой вклад в выполнение задания, используя свои сильные стороны. Хочется отметить, что, хотя и обязанности были четко распределены, мы активно поддерживали коммуникацию и помогали друг другу, если возникали вопросы, и решали их совместно.

Распределение по заданиям следующее:

1. Ради Мария – описание и анализ предметной области, монтаж видеоролика.
2. Бокова Полина – проект базы данных (инфологическая модель, даталогическая модель и приведение ее к 3НФ), вставка данных, создание представлений, отчет в MS Excel
3. Калякина Мария – разработка хранимых процедур, функций, триггеров, отчеты в Power BI
4. Петяева Елизавета – создание различных запросов, разработка индекса.