|  |
| --- |
| МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ  ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА» (РУТ (МИИТ))  ИНСТИТУТ ТРАНСПОРТНОЙ ТЕХНИКИ И СИСТЕМ УПРАВЛЕНИЯ   \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Кафедра «Управление и защита информации» |
|  |
| **КУРСОВОЙ ПРОЕКТ**  **по дисциплине «Основы построения защищенных баз данных»** |
|  |
|  |
|  |
| Выполнил: ст. гр. ТКИ – 441  Комаричев Г.Ю. |
| Проверил: к.т.н., доц.  Васильева М.А. |
|  |
|  |
|  |
| Москва 2024 |

**Оглавление**

[Введение 3](#_Toc167878894)

[Проектирование базы данных 4](#_Toc167878895)

[Нормализация полученных отношений 8](#_Toc167878896)

[Создание базы данных 9](#_Toc167878897)

[Создание таблиц 9](#_Toc167878898)

[Создание ограничений на пустые значения 10](#_Toc167878899)

[Добавление первичных ключей 11](#_Toc167878900)

[Добавление внешних ключей 11](#_Toc167878901)

[Создание ограничений на уникальное значение 12](#_Toc167878902)

[Заполнение базы данных 13](#_Toc167878903)

[Диаграмма базы данных 14](#_Toc167878904)

[Формирование запросов 15](#_Toc167878905)

[Формирование процедур 18](#_Toc167878906)

[Формирование триггера 22](#_Toc167878907)

[Вывод 23](#_Toc167878908)

Введение

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

В данном курсовом проекте ставится задача разработать РБД в СУБД PostgreSQL по теме «Мебельный магазин», необходимо разработать скрипты на создание и заполнение РБД.

База данных создается для информационного обслуживания сотрудников магазина. Магазин продает различные мебельные товары.

Готовые запросы:

* Показывать ассортимент предлагаемой мебели и цену товара.
* Показывать количество проданной мебели выбранного образца за
* отчетный период времени.
* Показывать список сделанных заказов за отчетный период времени.
* Рассчитывать стоимость заказа клиента по индивидуальному проекту.

Проектирование базы данных

Выделим базовые сущности базы данных «Мебельный магазин»:

* Заказы;
* Товары;
* Покупатели;
* Наличие товара;
* Продажи.

Были созданы следующие сущности:

1. Таблица "Orders" - содержит информацию о заказах.
2. Таблица "Product" – содержит информацию о товарах.
3. Таблица "Buyers" - содержит информацию о покупателях.
4. Таблица "Availability" - содержит информацию о наличие товара.
5. Таблица "Sales" - содержит информацию о продажах.

На рисунке 1 представлена ER-диаграмма БД.

|  |
| --- |
|  |
| 1. – ER-диаграмма базы данных «Мебельный магазин» |

1. - Схема отношения Заказы (Orders)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор заказа | Order\_ID | int4 | Первичный ключ, обязательное поле |
| Идентификатор покупателя | Buyers\_ID | Int4 | Внешний ключ, обязательное поле |
| Идентификатор товара | Product\_ID | int4 | Внешний ключ, обязательное поле |
| Дата заказа | Order\_Date | Date | Обязательное поле |
| Сумма заказа | Total | Numeric | Обязательное поле |
| Статус заказа | Status | varchar(200) | Обязательное поле |
| Дата исполнения заказа | Completion\_Date | Date | Обязательное поле |
| Наценка | Markup | Int4 | Обязательное поле |

1. - Схема отношения Товаров (Product)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор товара | Product\_ID | int4 | Первичный ключ, обязательное поле |
| Название товара | Product\_Name | varchar(200) | Обязательное поле |
| Цена товара | Product\_Price | numeric | Обязательное поле |

1. - Схема отношения Покупателей (Buyers)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор покупателя | ­ Buyers\_ID | int4 | Первичный ключ, обязательное поле |
| Имя покупателя | Buyers\_Name | varchar(200) | Обязательное поле |
| Номер телефона | Buyers\_PhoneNumber | Int4 | Обязательное поле |

1. - Схема отношения Наличия товара (Availability)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Кол-во товара | Quantity | int4 | Первичный ключ, обязательное поле |
| Идентификатор товара | Product\_ID | int4 | Внешний ключ, обязательное поле |
| Название товара | Product\_Price | numeric | Внешний ключ, обязательное поле |

1. - Схема отношения Продаж (Sales)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор продажи | Sales\_ID | int4 | Первичный ключ, обязательное поле |
| Идентификатор товара | Product\_ID | int4 | Внешний ключ, обязательное поле |
| Дата продажи | Sales\_Date | Date | Обязательное поле |
| Сумма продаж | Sales\_Total | numeric | Обязательное поле |

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.

**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, т.к:

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

**3. Третья нормальная форма (3NF)**

Все три таблицы находятся в третьей нормальной форме, т.к:

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

Создание базы данных

Ниже представлены скрипты для создания базы данных и таблиц.

Создание базы данных «Мебельный Магазин»:

**create** **database** **"FurnitureStore"**;

Создание таблиц

Создание таблицы Заказы:

**CREATE** **TABLE** ORDERS (

Order\_ID **INT**,

Buyers\_ID **INT**,

Product\_ID **INT**,

Order\_Date **DATE**,

Total **numeric** (8,4),

Status **VARCHAR**(20),

Completion\_date **DATE**,

Markup **INT**

);

Создание таблицы с информацией о товаре:

**CREATE** **TABLE** PRODUCT (

Product\_ID **INT**,

Product\_Name **VARCHAR**(200),

Product\_Price **numeric** (8,4),

);

Создание таблицы с информацией о покупателе:

**CREATE** **TABLE** BUYERS (

Buyers\_ID **INT**,

Buyers\_Name **VARCHAR**(200),

Buyers\_PhoneNumber **INT**

);

Создание таблицы с информацией о наличии товара:

**CREATE** **TABLE** Availability (

Product\_ID **INT**,

Product\_Price **numeric** (8,4),

Quantity **INT**

);

Создание таблицы с информацией о продажах:

**CREATE** **TABLE** SALES (

Sales\_ID **INT**,

Product\_ID **INT**,

Sales\_Total **numeric** (8,4),

Sales\_Date **DATE**

);

Создание ограничений на пустые значения

В каждой из таблиц добавим ограничения на пустые значения для столбцов, которые не могут являться пустыми.

**ALTER** **TABLE** orders

**ALTER** **COLUMN** order\_ID **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Buyers\_ID **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Product\_ID **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Order\_Date **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Total **SET** **NOT** **null**;

**ALTER** **TABLE** product

**ALTER** **COLUMN** product\_ID **SET** **NOT** **NULL**,

**ALTER** **COLUMN** product\_name **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Product\_price **SET** **NOT** **null**;

**ALTER** **TABLE** buyers

**ALTER** **COLUMN** Buyers\_ID **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Buyers\_Name **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Buyers\_PhoneNumber **SET** **NOT** **null**;

**ALTER** **TABLE** Availability

**ALTER** **COLUMN** product\_ID **drop** **NOT** **NULL**,

**ALTER** **COLUMN** Product\_price **drop** **NOT** **NULL**,

**ALTER** **COLUMN** Quantity **drop** **NOT** **null**;

**ALTER** **TABLE** sales

**ALTER** **COLUMN** product\_ID **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Sales\_ID **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Sales\_total **SET** **NOT** **NULL**,

**ALTER** **COLUMN** Sales\_date **SET** **NOT** **null**;

Добавление первичных ключей

Ниже представлены ограничение первичного ключа.

**ALTER** **TABLE** orders

**ADD** **CONSTRAINT** pk\_orders **PRIMARY** **KEY** (Order\_ID);

**ALTER** **TABLE** product

**ADD** **CONSTRAINT** pk\_product **PRIMARY** **KEY** (Product\_ID)

**ALTER** **TABLE** buyers

**ADD** **CONSTRAINT** pk\_buyers **PRIMARY** **KEY** (buyers\_ID)

**ALTER** **TABLE** Availability

**ADD** **CONSTRAINT** pk\_Availability **PRIMARY** **KEY** (Quantity)

**ALTER** **TABLE** sales

**ADD** **CONSTRAINT** pk\_sales **PRIMARY** **KEY** (Sales\_ID)

Добавление внешних ключей

Ниже представлены связи между таблицами с помощью добавления внешних ключей.

**ALTER** **TABLE** orders

**ADD** **CONSTRAINT** fk\_orders\_buyers **FOREIGN** **KEY** (buyers\_ID) **REFERENCES** Buyers(buyers\_ID),

**ADD** **CONSTRAINT** fk\_orders\_product **FOREIGN** **KEY** (product\_ID) **REFERENCES** Product(product\_ID);

**ALTER** **TABLE** availability

**ADD** **CONSTRAINT** fk\_avail\_product **FOREIGN** **KEY** (Product\_ID) **REFERENCES** Product(Product\_ID),

**ADD** **CONSTRAINT** fk\_avail\_price **FOREIGN** **KEY** (Product\_Price) **REFERENCES** Product(Product\_price);

**ALTER** **TABLE** sales

**ADD** **CONSTRAINT** fk\_sales\_product **FOREIGN** **KEY** (product\_ID) **REFERENCES** Product(product\_ID);

Создание ограничений на уникальное значение

Ниже представлены ограничения уникальности для полей.

**ALTER** **TABLE** orders

**ADD** **CONSTRAINT** uk\_orders\_id **UNIQUE** (order\_ID);

**ALTER** **TABLE** product

**ADD** **CONSTRAINT** uk\_product\_id **UNIQUE** (product\_ID);

**ALTER** **TABLE** buyers

**ADD** **CONSTRAINT** uk\_buyers\_id **UNIQUE** (buyers\_ID),

**ADD** **CONSTRAINT** uk\_buyers\_phone **UNIQUE** (Buyers\_PhoneNumber);

**ALTER** **TABLE** sales

**ADD** **CONSTRAINT** uk\_sales\_id **UNIQUE** (sales\_ID);

Заполнение базы данных

Ниже представлены скрипты для заполнения базы данных «Мебельный магазин».

**INSERT** **INTO** orders (Order\_ID, buyers\_id , product\_id,order\_date,total,status,completion\_date,markup) **VALUES**

(1, 1, 1,**'2024-04-15'**,1998.75,**'Готов'**,**'2024-04-25'**, 25),

(2, 2, 2,**'2024-04-01'**, 2835.8, **'В работе'**,**'2024-03-10'**, 10),

(3, 3, 3,**'2024-03-02'**, 3325, **'Готов'**,**'2024-03-12'**, 33),

(4, 4, 4,**'2024-05-17'**, 3988.95, **'Готов'**,**'2024-05-27'**,5 ),

(5, 5, 5,**'2024-01-11'**, 1127, **'Готов'**,**'2024-01-21'**, 27);

**INSERT** **INTO** PRODUCT (Product\_ID, Product\_Name, Product\_Price) **VALUES**

(1, **'Дубовый обеденный стол'**, 1599),

(2, **'Стул Модерн'**, 2578),

(3, **'Мягкий диван Семейный уют'**, 2500),

(4, **'Шкаф Эко-стиль '**, 3799),

(5, **'Кровать Бархат'**, 1100);

**INSERT** **INTO** BUYERS (Buyers\_ID, Buyers\_Name, Buyers\_PhoneNumber) **VALUES**

(1, **'Иванов Иван'**, **'123-456-7890'**),

(2, **'Петров Петр'**, **'456-789-0123'**),

(3, **'Сидоров Сидор'**, **'789-012-3456'**),

(4, **'Александров Александр'**, **'012-345-6789'**),

(5, **'Егоров Егор'**, **'234-567-8901'**);

**INSERT** **INTO** availability (Product\_ID,Product\_Price,quantity) **values**

(1,1599,20),

(2,2578,47),

(3,2500,56),

(4,3799,2),

(5,1100,71);

**INSERT** **INTO** SALES (Sales\_ID, Product\_ID, Sales\_Total, Sales\_Date) **VALUES**

(1, 1, 150.00, **'2024-05-15'**),

(2, 3, 450.00, **'2024-05-16'**),

(3, 2, 100.00, **'2024-05-17'**),

(4, 4, 300.00, **'2024-05-18'**),

(5, 5, 600.00, **'2024-05-19'**);

Диаграмма базы данных

На рисунке 2 представлена диаграмма БД.

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

Формирование запросов

1) Показывать ассортимент предлагаемой мебели и цену товара.

--Показывать ассортимент предлагаемой мебели и цену товара.

**CREATE** **OR** REPLACE **FUNCTION** ShowProductInfo()

**RETURNS** **TABLE** (Product\_Name **VARCHAR** (200), Product\_Price **numeric**(8,4)) **AS** $$

**BEGIN**

**RETURN** QUERY

**SELECT** Product.product\_name, Product.product\_price

**FROM** Product ;

**END**;

$$ **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Вывод ассортимента мебели и цены товара |

2) Показывать количество проданной мебели выбранного образца за отчетный период времени.

--Показывать количество проданной мебели выбранного образца за отчетный период времени.

**CREATE** **OR** REPLACE **FUNCTION** show\_sales\_count(

**IN** start\_date\_param **VARCHAR**,

**IN** end\_date\_param **VARCHAR**

)

**RETURNS** **TABLE** (Product\_Name **VARCHAR**, Sales\_Count **INT**) **AS** $$

**BEGIN**

**RETURN** QUERY

**SELECT** p.Product\_Name, **COUNT**(s.Sales\_ID)::**INTEGER** **AS** Sales\_Count

**FROM** Product p

**JOIN** Sales s **ON** p.Product\_ID = s.Product\_ID

**WHERE** s.Sales\_Date **BETWEEN** start\_date\_param **AND** end\_date\_param

**GROUP** **BY** p.Product\_Name;

**END**;

$$ **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Вывод информации о проданной мебели за промежуток времени |

3) Показывать список сделанных заказов за отчетный период времени.

--Показывать список сделанных заказов за отчетный период времени.

**CREATE** **OR** **REPLACE** **FUNCTION** show\_order\_count(

start\_date **VARCHAR**(200),

end\_date **VARCHAR**(200)

)

**RETURNS** **TABLE** (

order\_id **INT**,

a\_id **INT**,

buyers\_id **INT**,

product\_id **INT**,

order\_date **VARCHAR**(200),

Status **VARCHAR**(20),

completion\_date **VARCHAR**(200)

) **AS** **$$**

**BEGIN**

**RETURN** QUERY

**SELECT** Orders.order\_id,Orders.a\_id, orders.buyers\_id, orders.product\_id, orders.status, orders.order\_date, orders.completion\_date

**from** Orders

**WHERE** orders.order\_date **BETWEEN** start\_date **AND** end\_date;

**END**;

**$$** **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Вывод списка сделанных заказов за период времени |

4) Рассчитывать стоимость заказа клиента по индивидуальному проекту.

--Рассчет наценки.

**CREATE** **OR** REPLACE **FUNCTION** calculate\_markup()

**RETURNS** **TABLE** (

Product\_Name **VARCHAR**(200),

Price\_Before\_Markup **numeric** (8,4),

Price\_After\_Markup **numeric** (8,4)

) **AS** $$

**BEGIN**

**RETURN** QUERY

**SELECT**

product.product\_name,

product.product\_price,

product.product\_price \* (1 + markup / 100.0) **AS** Price\_After\_Markup

**FROM**

PRODUCT;

**END**;

$$ **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Расчет стоимости |

Формирование процедур

1. Процедура добавления данных в таблицу Product:

-- Создание процедуры для вставки данных в таблицу PRODUCT

**CREATE** **OR** **REPLACE** **PROCEDURE** insert\_product(

p\_product\_id **INT**,

p\_product\_name **VARCHAR**,

p\_product\_price **numeric**,

p\_markup **INT**

)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** PRODUCT (Product\_ID, Product\_Name, Product\_Price, Markup)

**VALUES** (p\_product\_id, p\_product\_name, p\_product\_price, p\_markup);

**END**;

**$$**;

**call** insert\_product(6, **'Стул деревянный'**, 2599, 5)

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Product |

1. Процедура добавления данных в таблицу Buyers:

-- Создание процедуры для вставки данных в таблицу BUYERS

**CREATE** **OR** **REPLACE** **PROCEDURE** insert\_buyer(

p\_buyers\_id **INT**,

p\_buyers\_name **VARCHAR**,

p\_buyers\_phone\_number **VARCHAR**

)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** BUYERS (Buyers\_ID, Buyers\_Name, Buyers\_PhoneNumber)

**VALUES** (p\_buyers\_id, p\_buyers\_name, p\_buyers\_phone\_number);

**END**;

**$$**;

**CALL** insert\_buyer(6, **'Дмитриев Дмитрий'**, **'125-456-780'**);

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Buyers |

1. Процедура добавления данных в таблицу Availability:

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_availability(

p\_A\_ID **INT**,

p\_Product\_ID **INT**,

p\_Quantity **INT**

)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** Availability (A\_ID, Product\_ID, Quantity)

**VALUES** (p\_A\_ID, p\_Product\_ID, p\_Quantity);

**END**;

**$$**;

**CALL** add\_availability(6, 6, 56);

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Availability |

1. Процедура добавления данных в таблицу Order:

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_order(

p\_Order\_ID **INT**,

p\_A\_ID **INT**,

p\_Buyers\_ID **INT**,

p\_Product\_ID **INT**,

p\_Order\_Date **Varchar**(200),

p\_Total **numeric** (8,4),

p\_Status **VARCHAR**(20),

p\_Completion\_date **Varchar**(200)

)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** ORDERS (Order\_ID, A\_ID, Buyers\_ID, Product\_ID, Order\_Date, Total, Status, Completion\_date)

**VALUES** (p\_Order\_ID, p\_A\_ID, p\_Buyers\_ID, p\_Product\_ID, p\_Order\_Date, p\_Total, p\_Status, p\_Completion\_date);

**END**;

**$$**;

**CALL** add\_order(6, 6, 2, 6, **'2024-05-24'**, 2728.95, **'Готов'**, **'2024-06-01'**);

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Order |

1. Процедура добавления данных в таблицу Sales:

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_sale(

p\_Sales\_ID **INT**,

p\_Product\_ID **INT**,

p\_Sales\_Total **NUMERIC** (8,4),

p\_Sales\_Date **VARCHAR**(200),

p\_A\_ID **INT**,

p\_Sales\_Quantity **INT**

)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** SALES (Sales\_ID, Product\_ID, Sales\_Total, Sales\_Date, A\_ID, Sales\_Quantity)

**VALUES** (p\_Sales\_ID, p\_Product\_ID, p\_Sales\_Total, p\_Sales\_Date, p\_A\_ID, p\_Sales\_Quantity);

**END**;

**$$**;

**CALL** add\_sale(6, 2, 2835.80, **'2024-05-24'**, 2, 4);

**CALL** add\_sale(8, 1, 1918.8, **'2024-05-24'**, 1, 5);

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Sales |

Формирование триггера

Ниже представлен триггер, который позволяет обновлять кол-во товара после новой продажи

--Функция обновления наличия после продажи

**CREATE** **OR** **REPLACE** **FUNCTION** update\_availability\_after\_sale()

**RETURNS** **TRIGGER** **AS** **$$**

**BEGIN**

**UPDATE** Availability

**SET** Quantity = Quantity - **NEW**.Sales\_Quantity

**WHERE** Product\_ID = **NEW**.Product\_ID;

**RETURN** **NEW**;

**END**;

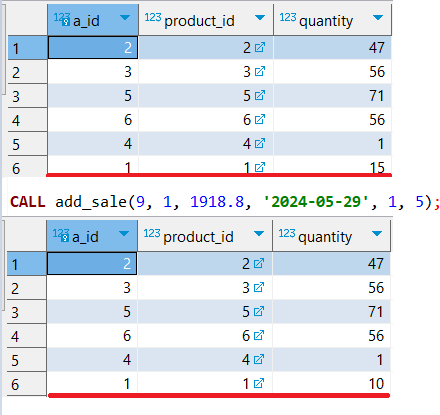
**$$** **LANGUAGE** plpgsql;

**CREATE** **TRIGGER** update\_availability\_after\_sale\_trigger

**AFTER** **INSERT** **ON** SALES

**FOR** **EACH** **ROW**

**EXECUTE** **FUNCTION** update\_availability\_after\_sale();



|  |
| --- |
|  |
| 1. – Наличие товара после новой продажи |

Вывод

В результате создания курсового проекта была спроектирована, создана и заполнена база данных для информационного обслуживания сотрудников магазина.

Были изучены операторы SQL на примере СУБД Postgres Pro, необходимые для фильтрации набора данных; операторы, необходимые для работы с данными, находящимися в разных таблицах, получен навык создания запросов на соединение данных из множества таблиц.