

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Базы данных»
Тема: Реализация базы данных в СУБД PostgreSQL

Студент гр. 3384

Козьмин Н.В.

Преподаватель

Михайлова С.В.

Санкт-Петербург

2025

Цель работы.

Необходимо развернуть PostgreSQL локально:

- Написать запросы для создания таблиц.
- Заполнить тестовыми данными: 5-10 строк на каждую таблицу, обязательно наличие связи между ними, данные приближены к реальности.
- Написать запросы к БД, отвечающие на вопросы.

Задание.

Вариант 8. Пусть требуется создать программную систему, предназначенную для директора продовольственного магазина. Такая система должна обеспечивать хранение сведений о магазине, об имеющихся в нем товарах, о торговых базах и товарах, хранящихся на этих базах. Магазин осуществляет закупку товаров на разных базах, предпочитая при этом закупать одни виды товара на одних базах, а другие на других. Магазин характеризуется классом, номером и имеет несколько отделов. Каждый товар в каждом магазине продается, по крайней мере, в одном отделе. Каждый отдел имеет заведующего. Товары, имеющиеся в магазине и хранящиеся на базах, характеризуются ценой, сортом и количеством. Розничные цены в магазине зависят от класса магазина. Директор магазина должен иметь возможность изменить цену товара по своему усмотрению, осуществить закупку недостающего товара на базе. Он может также закрыть один из отделов или открыть новый, при этом товары могут перемещаться из отдела в отдел. Директору могут потребоваться следующие сведения:

- Какие товары имеются в магазине (на базе)?
- Какие отсутствующие товары может заказать магазин на базе?
- Какие товары, и в каком количестве имеются в отделе магазина?
- Список заведующих отделами магазина?
- Суммарная стоимость товара в каждом отделе?
- На каких базах, и в каких количествах есть товар нужного наименования?

Выполнение.

Продemonстрируем работу с СУБД PostgreSQL в pgAdmin 4. Укажем на рис. 1 и 2 подключение к “серверу” на локальном хосте.

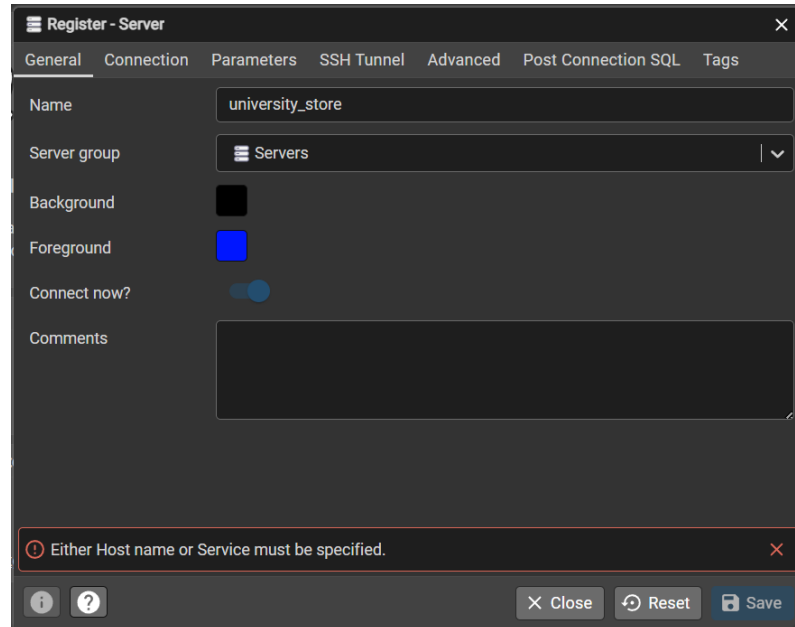


Рис. 1

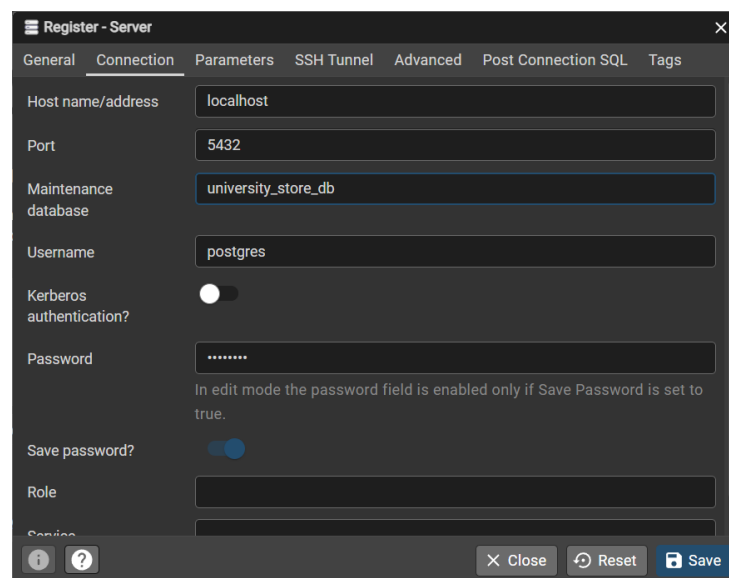


Рис. 2

Теперь приведем работу в Query Tool для нашего сервера на рис. 3.

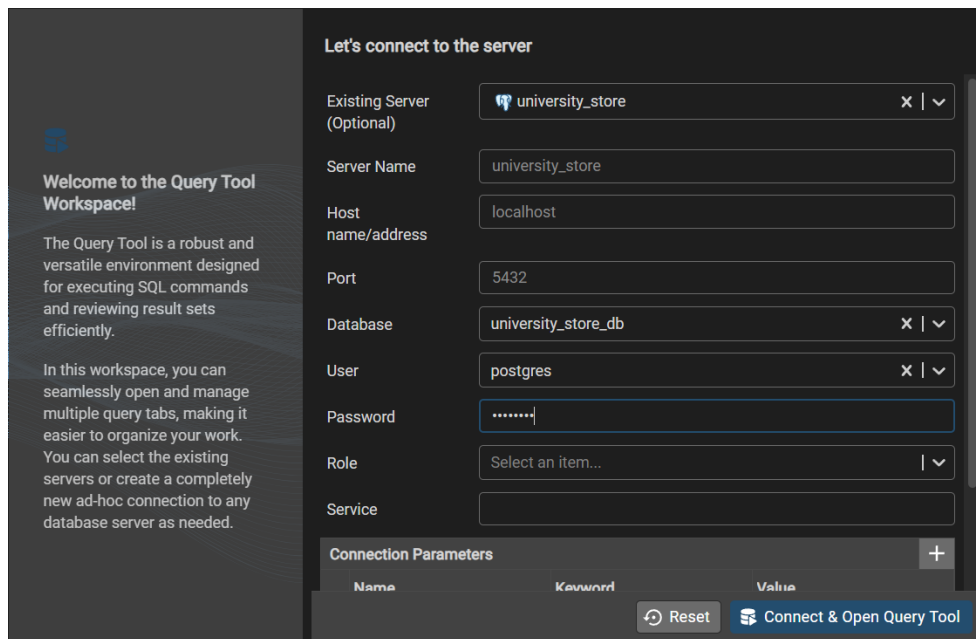


Рис. 3

Затем покажем скриншоты на каждый запуск запросов. На рис. 4 укажем создание таблиц по реляционной модели из предыдущей лаб. работы и отдельно запустим наполнение таблиц данными.

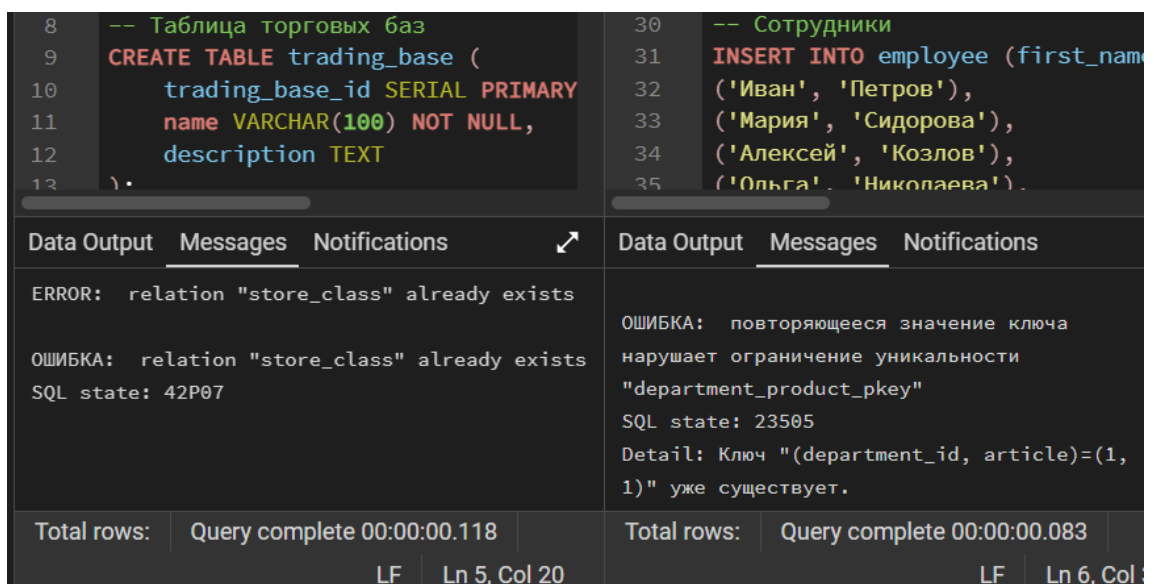


Рис. 4

На этих двух вкладках выводятся ошибки, так как запросы запускаются повторно и данные уже есть в базе данных. Далее на рис. 5-7 приведем запросы, получающие ответы на вопросы по заданию.

| store_name | department_name | article | product_name |
|----------------------|-----------------|---------|--------------|
| Супермаркет "Восток" | Молочный отдел | 1 | Моло |
| Супермаркет "Восток" | Напитки | 1 | Моло |
| Супермаркет "Восток" | Хлебный отдел | 2 | Хлеб |

| trading_base_name | article | product_name |
|--------------------------------|---------|--------------------|
| Южный распределительный цен... | 6 | Кофе молотый |
| Южный распределительный цен... | 4 | Курица охлажденная |
| Южный распределительный цен... | 1 | Молоко 2,5% |
| Южный распределительный цен... | 5 | Рис круглый |

Рис. 5

| store_name | available_quantity | base_price | priority |
|---------------------|--------------------|------------|----------|
| за "Северная" | 200 | 85.50 | 1 |
| еделительный цен... | 180 | 82.00 | 2 |
| за "Северная" | 150 | 45.00 | 2 |

| department_name | product_name | quantity |
|-----------------|--------------------|----------|
| Молочный отдел | Йогурт натуральный | 30 |
| Молочный отдел | Молоко 2,5% | 50 |
| Молочный отдел | Сыр Российский | 25 |

Рис. 6

| store_name | department_name | manager_name |
|----------------------|-----------------|-----------------|
| Супермаркет "Восток" | Молочный отдел | Мария Сидоро... |
| Супермаркет "Восток" | Напитки | Дмитрий Орлов |
| Супермаркет "Восток" | Хлебный отдел | Алексей Козлов |

| store_name | department_name | total_value |
|------------------------|--------------------|-------------|
| Специмагазин "Техника" | Бытовая техника | 224000.00 |
| Премиум маркет "Лю... | Кондитерский отдел | 7500.00 |
| Магазин "Эконом" | Бакалея | 5250.00 |
| Супермаркет "Восток" | Молочный отдел | 4750.00 |

| product_name | trading_base | base_desc |
|--------------|--------------------------------|--------------|
| Молоко 2,5% | Центральная база "Северная" | Крупнейши... |
| Молоко 2,5% | Южный распределительный цен... | Современн... |

Рис.7

Запросы успешно выполняются и отвечают на поставленные вопросы.

Приведем ссылки на результаты работы в приложении А и исходный код в приложении Б.

Выводы.

В ходе выполнения лабораторной работы создана нормализованная реляционная модель из 10 таблиц, реализованы связи между сущностями через внешние ключи и учтены требования для торговой сети. При наполнении базы данных создано по 5-15 записей в каждой таблице. Данные реалистичны и взаимосвязаны. Учтены различные сценарии работы сети. Разработанные SQL-запросы реализуют базовую аналитику и отвечают на поставленные вопросы. Добавлено выполненное задание на защиту в приложении В.

ПРИЛОЖЕНИЕ А

ССЫЛКИ НА РЕЗУЛЬТАТЫ

Ссылка на www.db-fiddle.com:

<https://www.db-fiddle.com/f/gdKQpYkuz93Xx4hwa2Wr5P/1>

Ссылка на pr:

<https://github.com/moevm/sql-2025-3384/pull/26>

ПРИЛОЖЕНИЕ Б

ИСХОДНЫЙ КОД

Код create_university_store.sql:

```
-- Таблица классов магазинов
CREATE TABLE store_class (
    store_class_id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    description TEXT
);

-- Таблица торговых баз
CREATE TABLE trading_base (
    trading_base_id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    description TEXT
);

-- Таблица сотрудников
CREATE TABLE employee (
    employee_id SERIAL PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL
);

-- Таблица магазинов
CREATE TABLE store (
    store_id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    description TEXT,
    store_class_id INTEGER REFERENCES store_class(store_class_id),
    director_id INTEGER REFERENCES employee(employee_id)
);

-- Таблица отделов
CREATE TABLE department (
    department_id SERIAL PRIMARY KEY,
    store_id INTEGER REFERENCES store(store_id),
    manager_id INTEGER REFERENCES employee(employee_id),
    name VARCHAR(100) NOT NULL
);

-- Таблица товаров
CREATE TABLE product (
    article SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    sort VARCHAR(50)
);

-- Таблица товаров в отделах
CREATE TABLE department_product (
    department_id INTEGER REFERENCES department(department_id),
    article INTEGER REFERENCES product(article),
    count INTEGER NOT NULL DEFAULT 0,
    PRIMARY KEY (department_id, article)
);

-- Таблица товаров на складах баз
CREATE TABLE warehouse_product (
```



```

        trading_base_id INTEGER REFERENCES trading_base(trading_base_id),
        article INTEGER REFERENCES product(article),
        count INTEGER NOT NULL DEFAULT 0,
        price DECIMAL(10,2) NOT NULL,
        PRIMARY KEY (trading_base_id, article)
    );

-- Таблица цен товаров по классам магазинов
CREATE TABLE product_price (
    store_class_id INTEGER REFERENCES store_class(store_class_id),
    article INTEGER REFERENCES product(article),
    price DECIMAL(10,2) NOT NULL,
    PRIMARY KEY (store_class_id, article)
);

-- Таблица приоритетов поставок
CREATE TABLE warehouse_priority (
    article INTEGER REFERENCES product(article),
    store_id INTEGER REFERENCES store(store_id),
    trading_base_id INTEGER REFERENCES trading_base(trading_base_id),
    priority INTEGER NOT NULL DEFAULT 5,
    PRIMARY KEY (article, store_id, trading_base_id)
);

```

Код university_store_values.sql:

```

-- Классы магазинов
INSERT INTO store_class (name, description) VALUES
('Элитный', 'Магазины премиум-класса с широким ассортиментом'),
('Стандарт', 'Магазины среднего ценового сегмента'),
('Эконом', 'Бюджетные магазины'),
('Специализированный', 'Магазины узкой направленности'),
('Гипермаркет', 'Крупные торговые центры');

-- Торговые базы
INSERT INTO trading_base (name, description) VALUES
('Центральная база "Северная"', 'Крупнейший распределительный центр в северном регионе'),
('Южный распределительный центр', 'Современный логистический комплекс южного направления'),
('Западный складской комплекс', 'Складские помещения с системой климат-контроля'),
('Восточная база снабжения', 'База снабжения для розничных сетей восточного региона'),
('Центральный логистический центр', 'Основной хаб для федеральных сетей');

-- Сотрудники
INSERT INTO employee (first_name, last_name) VALUES
('Иван', 'Петров'),
('Мария', 'Сидорова'),
('Алексей', 'Козлов'),
('Ольга', 'Николаева'),
('Сергей', 'Васильев'),
('Елена', 'Федорова'),
('Дмитрий', 'Орлов'),
('Анна', 'Морозова'),
('Павел', 'Семенов'),
('Ирина', 'Волкова');

-- Магазины
INSERT INTO store (name, description, store_class_id, director_id) VALUES
('Супермаркет "Восток"', 'Крупный супермаркет в центре города', 2, 1),

```

```

('Гипермаркет "Мега"', 'Торговый центр с полным ассортиментом', 5, 4),
('Магазин "Эконом"', 'Бюджетный магазин для ежедневных покупок', 3, 6),
('Премиум маркет "Люкс"', 'Магазин премиум-класса', 1, 9),
('Спецмагазин "Техника"', 'Специализированный магазин электроники', 4, 4);

-- Отделы
INSERT INTO department (store_id, manager_id, name) VALUES
(1, 2, 'Молочный отдел'),
(1, 3, 'Хлебный отдел'),
(2, 5, 'Овощной отдел'),
(2, 7, 'Мясной отдел'),
(3, 8, 'Бакалея'),
(4, 10, 'Гастрономия'),
(4, 2, 'Кондитерский отдел'),
(5, 3, 'Бытовая техника'),
(5, 5, 'Электроника'),
(1, 7, 'Напитки');

-- Товары
INSERT INTO product (name, sort) VALUES
('Молоко 2,5%', 'Пастеризованное'),
('Хлеб Бородинский', 'Ржаной'),
('Картофель', 'Отборный'),
('Курица охлажденная', 'Бройлер'),
('Рис круглый', 'Высший сорт'),
('Кофе молотый', 'Арабика'),
('Шоколад горький', 'Премиум'),
('Чай черный', 'Цейлон'),
('Сок яблочный', 'Осветленный'),
('Телевизор LED', 'Smart TV'),
('Смартфон', 'Флагман'),
('Йогурт натуральный', 'Без добавок'),
('Сыр Российский', 'Полутвердый'),
('Колбаса докторская', 'Вареная'),
('Печенье овсяное', 'С шоколадом');

-- Товары в отделах
INSERT INTO department_product (department_id, article, count) VALUES
(1, 1, 50), (1, 12, 30), (1, 13, 25),
(2, 2, 40), (2, 15, 45),
(3, 3, 100), (3, 5, 60),
(4, 4, 35), (4, 14, 28),
(5, 5, 55), (5, 6, 20), (5, 7, 35),
(6, 7, 15), (6, 8, 25), (6, 13, 18),
(7, 7, 30), (7, 15, 40),
(8, 10, 8), (9, 11, 12),
(10, 1, 25), (10, 9, 35);

-- Товары на складах баз
INSERT INTO warehouse_product (trading_base_id, article, count, price)
VALUES
(1, 1, 200, 85.50), (1, 2, 150, 45.00), (1, 3, 500, 35.00),
(2, 4, 180, 320.00), (2, 1, 100, 90.00), (2, 5, 300, 95.00), (2, 6, 120,
450.00),
(3, 7, 90, 180.00), (3, 8, 200, 120.00), (3, 9, 150, 110.00),
(4, 10, 25, 25000.00), (4, 11, 40, 45000.00),
(5, 12, 100, 65.00), (5, 13, 80, 580.00), (5, 14, 60, 420.00),
(1, 15, 120, 85.00), (2, 1, 180, 82.00);

-- Цены товаров по классам магазинов
INSERT INTO product_price (store_class_id, article, price) VALUES
(1, 1, 120.00), (1, 2, 65.00), (1, 3, 50.00),
(2, 1, 95.00), (2, 2, 48.00), (2, 3, 38.00),

```

```

(3, 1, 80.00), (3, 2, 40.00), (3, 3, 30.00),
(1, 7, 250.00), (2, 7, 200.00), (3, 7, 150.00),
(4, 10, 28000.00), (5, 10, 27000.00);

-- Приоритеты поставок (с учетом trading_base_id)
INSERT INTO warehouse_priority (article, store_id, trading_base_id,
priority) VALUES
(1, 1, 1, 1), (1, 2, 1, 2), (1, 3, 1, 1),
(2, 1, 1, 2), (2, 2, 1, 3), (2, 3, 1, 2),
(3, 1, 1, 1), (3, 2, 1, 1), (3, 3, 1, 1),
(10, 4, 4, 1), (10, 5, 4, 1), (11, 5, 4, 1),
(7, 4, 3, 2), (7, 1, 3, 3),
(1, 1, 2, 2), (2, 1, 2, 3);

```

Код university_store_requests.sql:

```

-- 1.1 Какие товары имеются в магазине?
SELECT
    s.name as store_name,
    d.name as department_name,
    p.article as article,
    p.name as product_name,
    p.sort as product_sort,
    dp.count as quantity,
    pp.price as current_price,
    (dp.count * pp.price) as total_value
FROM store s
JOIN department d ON s.store_id = d.store_id
JOIN department_product dp ON d.department_id = dp.department_id
JOIN product p ON dp.article = p.article
JOIN product_price pp ON p.article = pp.article AND s.store_class_id =
pp.store_class_id
WHERE s.store_id = 1
ORDER BY d.name, p.name;

-- 1.2 Какие товары имеются на базе?
SELECT
    tb.name as trading_base_name,
    p.article as article,
    p.name as product_name,
    p.sort as product_sort,
    wp.count as available_quantity,
    wp.price as base_price,
    (wp.count * wp.price) as total_value
FROM trading_base tb
JOIN warehouse_product wp ON tb.trading_base_id = wp.trading_base_id
JOIN product p ON wp.article = p.article
WHERE tb.trading_base_id = 2
    AND wp.count > 0
ORDER BY p.name;

-- 2. Какие отсутствующие товары может заказать магазин на базе?
SELECT --DISTINCT
    s.name as store_name,
    p.name as product_name,
    tb.name as trading_base,
    wp.count as available_quantity,
    wp.price as base_price,
    w_priority.priority
FROM product p
JOIN warehouse_product wp ON p.article = wp.article
JOIN trading_base tb ON wp.trading_base_id = tb.trading_base_id

```

```

JOIN store s ON s.store_id = 1
JOIN warehouse_priority w_priority ON w_priority.store_id = s.store_id AND
w_priority.article = p.article AND w_priority.trading_base_id =
tb.trading_base_id
WHERE p.article IN (
    SELECT dp.article
    FROM department_product dp
    JOIN department d ON dp.department_id = d.department_id
    WHERE d.store_id = s.store_id AND dp.count = 0
)
AND wp.count >= 1
ORDER BY p.name, w_priority.priority;

-- 3. Какие товары, и в каком количестве имеются в отделе магазина?
SELECT
    s.name as store_name,
    d.name as department_name,
    p.name as product_name,
    dp.count as quantity
FROM store s
JOIN department d ON s.store_id = d.store_id
JOIN department_product dp ON d.department_id = dp.department_id
JOIN product p ON dp.article = p.article
WHERE d.department_id = 1
ORDER BY p.name;

-- 4. Список заведующих отделами магазина?
SELECT
    s.name as store_name,
    d.name as department_name,
    e.first_name || ' ' || e.last_name as manager_name
FROM store s
JOIN department d ON s.store_id = d.store_id
JOIN employee e ON d.manager_id = e.employee_id
WHERE s.store_id = 1
ORDER BY d.name;

-- 5. Суммарная стоимость товара в каждом отделе?
SELECT
    s.name as store_name,
    d.name as department_name,
    SUM(dp.count * pp.price) as total_value
FROM store s
JOIN department d ON s.store_id = d.store_id
JOIN department_product dp ON d.department_id = dp.department_id
JOIN product_price pp ON dp.article = pp.article AND s.store_class_id =
pp.store_class_id
GROUP BY s.name, d.name
ORDER BY total_value DESC;

-- 6. На каких базах, и в каких количествах есть товар нужного наименования?
SELECT
    p.name as product_name,
    tb.name as trading_base,
    tb.description as base_description,
    wp.count as available_quantity,
    wp.price as price_per_unit
FROM product p
JOIN warehouse_product wp ON p.article = wp.article
JOIN trading_base tb ON wp.trading_base_id = tb.trading_base_id
WHERE p.name ILIKE '%молоко%' OR p.article = 1
ORDER BY wp.count DESC;

```

ПРИЛОЖЕНИЕ В

ЗАДАНИЕ НА ЗАЩИТУ

Новая часть кода в university_store_requests.sql с описанием постановки задачи в комментариях:

```
-- 2.доп Какие отсутствующие товары может заказать магазин на базе (в том
числе, которых не было, но для которых есть приоритет)?
SELECT --DISTINCT
    s.name as store_name,
    p.name as product_name,
    tb.name as trading_base,
    wp.count as available_quantity,
    wp.price as base_price,
    w_priority.priority
FROM product p
JOIN warehouse_product wp ON p.article = wp.article
JOIN trading_base tb ON wp.trading_base_id = tb.trading_base_id
JOIN store s ON s.store_id = 1
JOIN warehouse_priority w_priority ON w_priority.store_id = s.store_id AND
w_priority.article = p.article AND w_priority.trading_base_id =
tb.trading_base_id
WHERE
(
    p.article IN (
        SELECT dp.article
        FROM department_product dp
        JOIN department d ON dp.department_id = d.department_id
        WHERE d.store_id = s.store_id AND dp.count = 0
    )
    OR p.article NOT IN (
        SELECT dp.article
        FROM department_product dp
        JOIN department d ON dp.department_id = d.department_id
        WHERE d.store_id = s.store_id
    )
)
AND wp.count >= 0
ORDER BY p.name, w_priority.priority;
```