|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра цифровой трансформации (ЦТ)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 2**

по дисциплине «Разработка баз данных»

|  |  |  |  |
| --- | --- | --- | --- |
| Студент группы ИНБО-01-17 | *ИНБО-12-23. Албахтин И.В.* | (подпись) | |
| Ассистент | *Брайловский А.В.* | (подпись) | |
|  |  | |  | |

Москва 2025 г.

ПРАКТИЧЕСКАЯ РАБОТА №2. МНОГОТАБЛИЧНЫЕ ЗАПРОСЫ И ТЕОРЕТИКО- МНОЖЕСТВЕННЫЕ ОПЕРАЦИИ В POSTGRES PRO

Цель:

• Научиться извлекать и комбинировать данные из нескольких связанных

таблиц с помощью соединений (JOIN) и теоретико-множественных операторов

(UNION, INTERSECT, EXCEPT), а также освоить продвинутые паттерны, такие как «само-соединение» и «анти-соединение».

• Сформировать глубокое концептуальное понимание и практические

навыки применения различных типов соединений таблиц (INNER, LEFT,

RIGHT, FULLJOIN) для извлечения связанных данных из нескольких

таблиц.

• Научиться применять теоретико-множественные операторы (UNION,

UNIONALL, INTERSECT, EXCEPT) для комбинирования и сравнения результатов нескольких независимых запросов, соблюдая правила их использования.

• Развить аналитические навыки для декомпозиции сложных бизнес-

вопросов в последовательность логических шагов, реализуемых с помощью SQL-запросов.

• Понять принцип работы специфических паттернов SQL, таких как «анти-соединение» (anti-join) для поиска несоответствий и «само-соединение»

(self-join) для работы с иерархическими данными в рамках одной таблицы.

**Постановка задачи**:

**Задание 1: демонстрация различных типов соединений.**

На основе индивидуальной схемы данных, составить и выполнить пять

аналитических запросов, демонстрирующих различные типы соединений.

Каждый запрос должен решать осмысленную задачу в рамках вашей предметной области.

1. В начале отчёта должны быть приложены скриншоты всех используемых таблиц индивидуальной схемы данных.

2. Запрос с INNER JOIN: подсчитайте количество связанных записей между таблицами (например, «сколько лекарств у каждого производителя?»)

3. Запрос с LEFT JOIN: проанализируйте наличие или отсутствие связей

(например, «сколько лекарств у каждого производителя, включая тех, у кого лекарств нет?»)

4. Запрос с RIGHT JOIN и WHERE... IS NULL (паттерн «анти- соединение»): найдите и подсчитайте записи без связей (например, «сколько лекарств не имеют производителя в базе?»)

5. Запрос с FULL JOIN: получите общую статистику – сколько всего связанных записей, и сколько записей без связей.

6. Запрос с CROSS JOIN: сформировать декартово произведение всех записей одной таблицы со всеми записями другой, создав тем самым все возможные комбинации строк между ними.

**Задание 2: применение теоретико-множественных операторов.**

На основе индивидуальной схемы данных составить и выполнить три запроса, демонстрирующих практическое применение операторов UNION,

INTERSECT и EXCEPT.

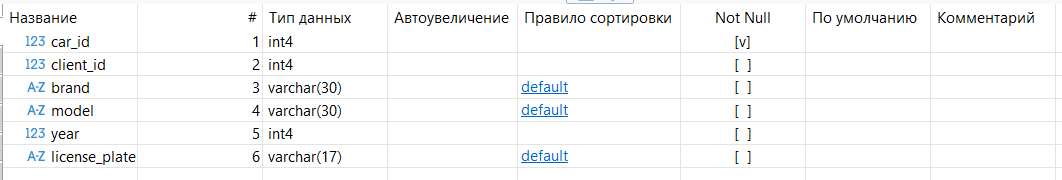
1. UNION: составить единый список из данных двух разных таблиц (столбцы должны быть совместимы по типу).

2. INTERSECT: найти общие записи, которые удовлетворяют двум разным условиям или находятся в двух разных наборах данных.

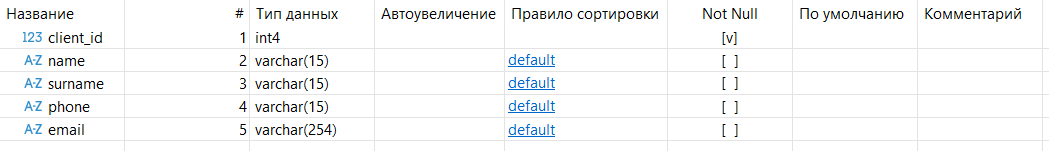
3. EXCEPT: найти записи, которые присутствуют в одном наборе данных, но отсутствуют в другом.

**ВЫПОЛНЕНИЕ ПРАКТИЧЕСКОЙ РАБОТЫ**

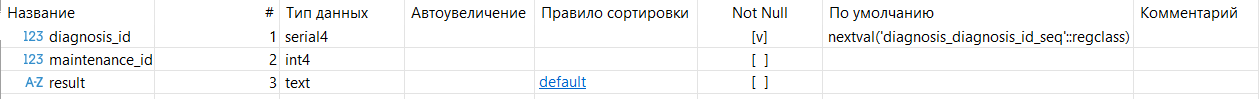
*Таблица 1. Таблица car (автомобиль)*



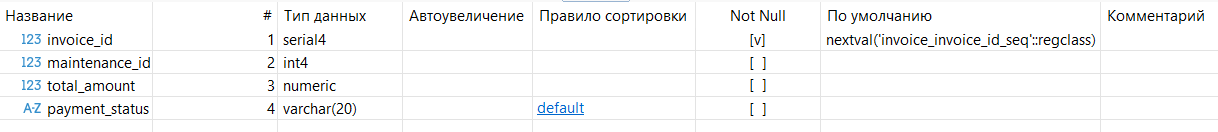
*Таблица 2. Таблица client (клиент)*



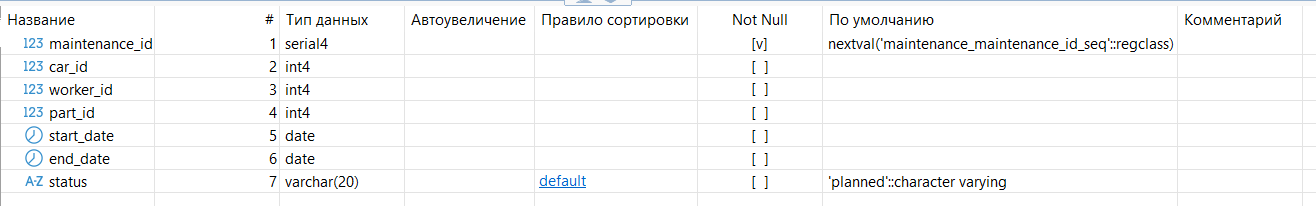
*Таблица 3. Таблица diagnosis (диагностика)*



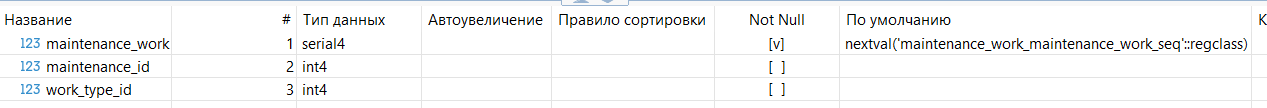
*Таблица 4. Таблица invoice (счёт за работы)*



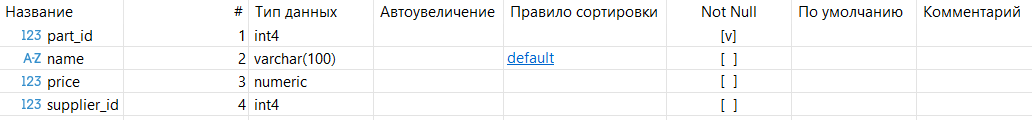
*Таблица 5. Таблица maintenance (обслуживание)*



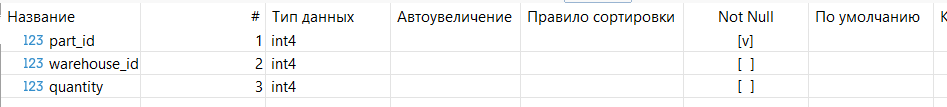
*Таблица 6. Таблица maintenance\_work (соеденительная таблица между обслуживанием и типом работы)*



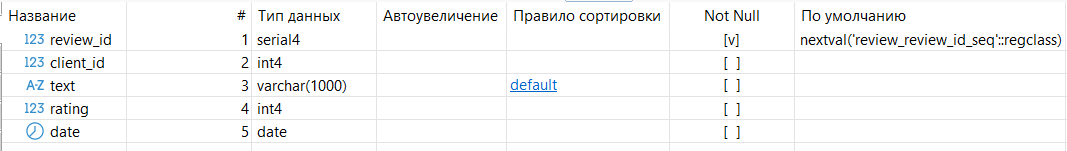
*Таблица 7. Таблица part (запчасти)*



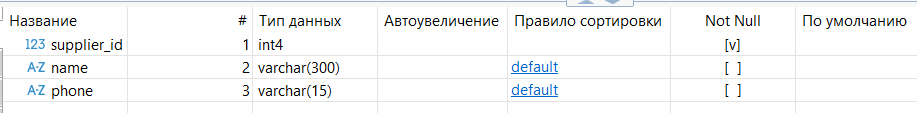
*Таблица 8. Таблица part\_warehouse (соеденительная таблица между складом и запчастями)*



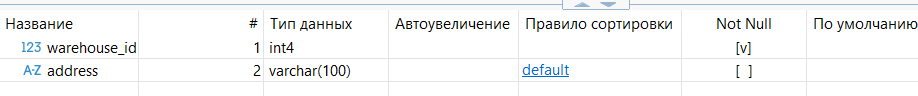
*Таблица 9. Таблица review (отзывы)*



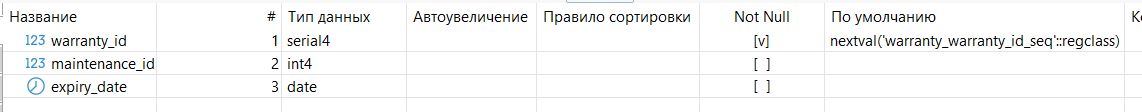
*Таблица 10. Таблица supplier (поставщики)*



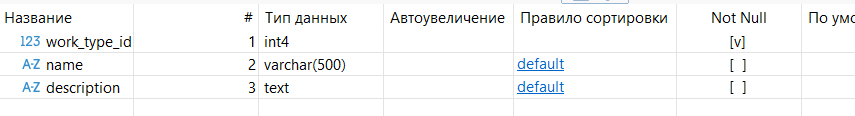
*Таблица 11. Таблица warehouse (склад)*



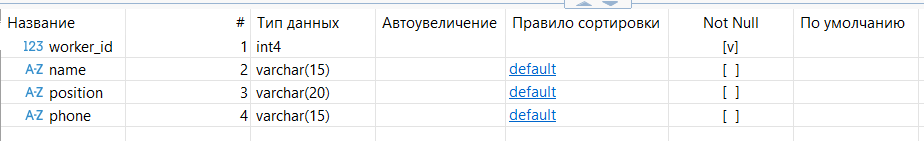
*Таблица 12. Таблица warranty (гарантия)*

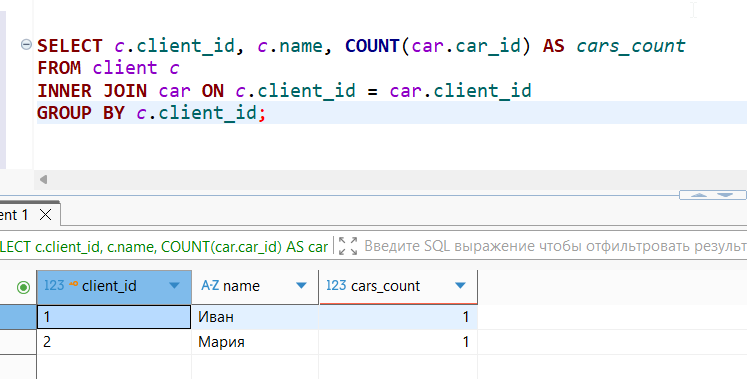


*Таблица 13. Таблица work\_type (тип работ)*

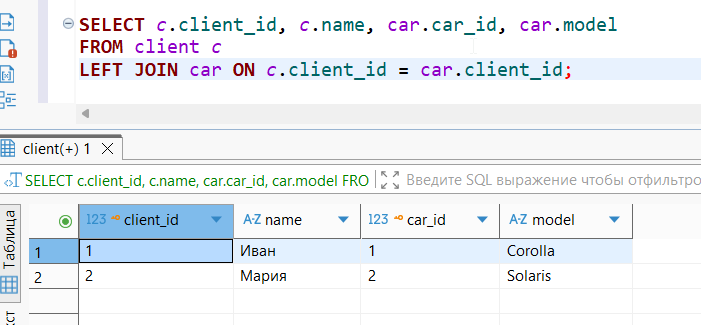


*Таблица 14. Таблица worker (сотрудник)*

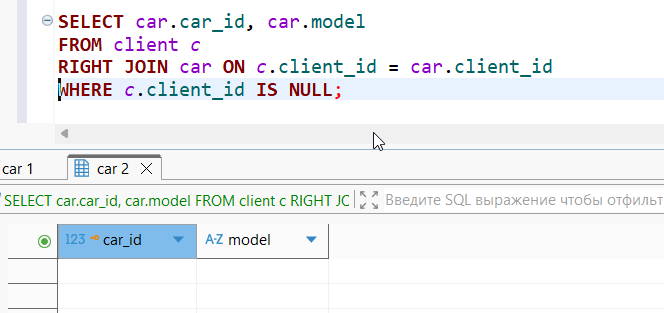


**Основы соединения таблиц (JOIN)** 

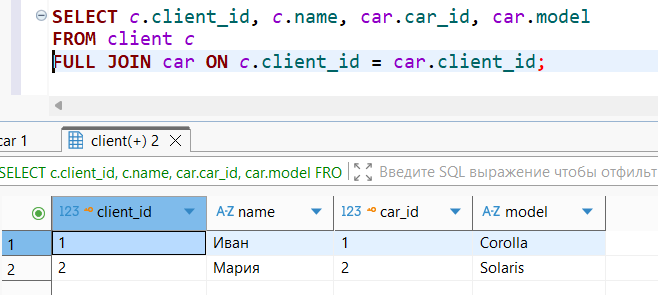
**Рисунок 1 - Найти количество машин, закреплённых за каждым клиентом (INNER JOIN)**



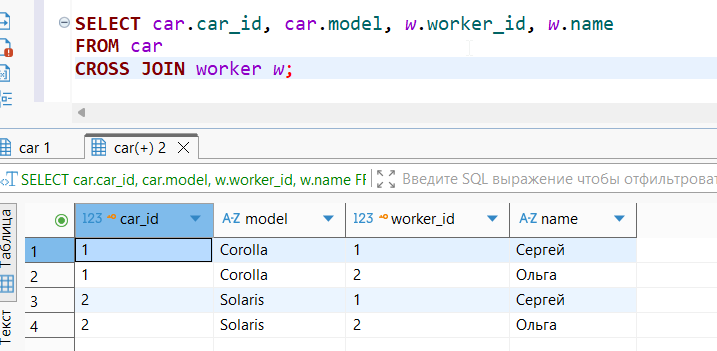
**Рисунок 2 - Показать всех клиентов и их автомобили, даже если машины у клиента отсутствуют (LEFT JOIN)**



**Рисунок 3 - Найти автомобили, у которых нет владельцев (клиентов,** **RIGHT JOIN + IS NULL)**

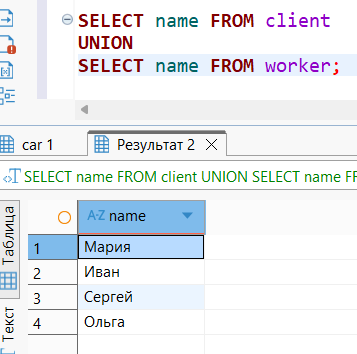


**Рисунок 4 - Получить список клиентов и машин, включая тех клиентов без машин и машины без клиентов (FULL JOIN)**



**Рисунок 5 - Создать все возможные комбинации “машина – работник сервиса” (CROSS JOIN)**

**Продвинутые техники и паттерны соединений**

****

**Рисунок 6 - Составить общий список всех имён клиентов и работников (UNION)**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Рисунок 7 - Найти имена, которые встречаются и среди клиентов, и среди работников (INTERSECT)**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Рисунок 8 - Найти клиентов, которые не являются работниками сервиса (EXCEPT)**

**ВЫВОД**

В ходе выполнения работы были изучены и реализованы различные виды соединений таблиц (INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN, CROSS JOIN), а также операторы работы с множествами (UNION, INTERSECT, EXCEPT).

Было показано, что:

* **INNER JOIN** используется для выборки только связанных данных (например, клиенты, у которых есть машины).
* **LEFT JOIN** позволяет выявить объекты без связей (например, клиентов без автомобилей).
* **RIGHT JOIN + IS NULL** удобно применять как анти-соединение для поиска «осиротевших» данных (например, автомобилей без владельцев).
* **FULL JOIN** даёт полную картину, объединяя все записи обеих таблиц, даже если связь отсутствует.
* **CROSS JOIN** формирует все возможные комбинации строк и полезен для генерации тестовых данных.

Операторы **UNION, INTERSECT и EXCEPT** позволили объединять, сравнивать и различать наборы данных, что важно при аналитической обработке информации.