



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-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)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
<a href="#">123</a> car_id	1	int4			[v]		
<a href="#">123</a> client_id	2	int4			[ ]		
<a href="#">A-Z</a> brand	3	varchar(30)		<a href="#">default</a>	[ ]		
<a href="#">A-Z</a> model	4	varchar(30)		<a href="#">default</a>	[ ]		
<a href="#">123</a> year	5	int4			[ ]		
<a href="#">A-Z</a> license_plate	6	varchar(17)		<a href="#">default</a>	[ ]		

Таблица 2. Таблица client (клиент)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
<a href="#">123</a> client_id	1	int4			[v]		
<a href="#">A-Z</a> name	2	varchar(15)		<a href="#">default</a>	[ ]		
<a href="#">A-Z</a> surname	3	varchar(15)		<a href="#">default</a>	[ ]		
<a href="#">A-Z</a> phone	4	varchar(15)		<a href="#">default</a>	[ ]		
<a href="#">A-Z</a> email	5	varchar(254)		<a href="#">default</a>	[ ]		

Таблица 3. Таблица diagnosis (диагностика)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
<a href="#">123</a> diagnosis_id	1	serial4			[v]	nextval('diagnosis_diagnosis_id_seq':regclass)	
<a href="#">123</a> maintenance_id	2	int4			[ ]		
<a href="#">A-Z</a> result	3	text		<a href="#">default</a>	[ ]		

Таблица 4. Таблица invoice (счёт за работы)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
<a href="#">123</a> invoice_id	1	serial4			[v]	nextval('invoice_invoice_id_seq':regclass)	
<a href="#">123</a> maintenance_id	2	int4			[ ]		
<a href="#">123</a> total_amount	3	numeric			[ ]		
<a href="#">A-Z</a> payment_status	4	varchar(20)		<a href="#">default</a>	[ ]		

Таблица 5. Таблица maintenance (обслуживание)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
<a href="#">123</a> maintenance_id	1	serial4			[v]	nextval('maintenance_maintenance_id_seq':regclass)	
<a href="#">123</a> car_id	2	int4			[ ]		
<a href="#">123</a> worker_id	3	int4			[ ]		
<a href="#">123</a> part_id	4	int4			[ ]		
<a href="#">🕒</a> start_date	5	date			[ ]		
<a href="#">🕒</a> end_date	6	date			[ ]		
<a href="#">A-Z</a> status	7	varchar(20)		<a href="#">default</a>	[ ]	'planned':character varying	

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

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
<a href="#">123</a> maintenance_work	1	serial4			[v]	nextval('maintenance_work_maintenance_work_seq':regclass)	
<a href="#">123</a> maintenance_id	2	int4			[ ]		
<a href="#">123</a> work_type_id	3	int4			[ ]		

Таблица 7. Таблица part (запчасти)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
<a href="#">123</a> part_id	1	int4			[v]		
<a href="#">A-Z</a> name	2	varchar(100)		<a href="#">default</a>	[ ]		
<a href="#">123</a> price	3	numeric			[ ]		
<a href="#">123</a> supplier_id	4	int4			[ ]		

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

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	K
<a href="#">123</a> part_id	1	int4			[v]		
<a href="#">123</a> warehouse_id	2	int4			[ ]		
<a href="#">123</a> quantity	3	int4			[ ]		

Таблица 9. Таблица review (отзывы)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	K
<a href="#">123</a> review_id	1	serial4			[v]	nextval('review_review_id_seq'::regclass)	
<a href="#">123</a> client_id	2	int4			[ ]		
<a href="#">A-Z</a> text	3	varchar(1000)		<a href="#">default</a>	[ ]		
<a href="#">123</a> rating	4	int4			[ ]		
<a href="#">🕒</a> date	5	date			[ ]		

Таблица 10. Таблица supplier (поставщики)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	K
<a href="#">123</a> supplier_id	1	int4			[v]		
<a href="#">A-Z</a> name	2	varchar(300)		<a href="#">default</a>	[ ]		
<a href="#">A-Z</a> phone	3	varchar(15)		<a href="#">default</a>	[ ]		

Таблица 11. Таблица warehouse (склад)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	K
<a href="#">123</a> warehouse_id	1	int4			[v]		
<a href="#">A-Z</a> address	2	varchar(100)		<a href="#">default</a>	[ ]		

Таблица 12. Таблица warranty (гарантия)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	K
<a href="#">123</a> warranty_id	1	serial4			[v]	nextval('warranty_warranty_id_seq'::regclass)	
<a href="#">123</a> maintenance_id	2	int4			[ ]		
<a href="#">🕒</a> expiry_date	3	date			[ ]		

Таблица 13. Таблица work\_type (тип работ)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	K
<a href="#">123</a> work_type_id	1	int4			[v]		
<a href="#">A-Z</a> name	2	varchar(500)		<a href="#">default</a>	[ ]		
<a href="#">A-Z</a> description	3	text		<a href="#">default</a>	[ ]		

Таблица 14. Таблица worker (сотрудник)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	K
<a href="#">123</a> worker_id	1	int4			[v]		
<a href="#">A-Z</a> name	2	varchar(15)		<a href="#">default</a>	[ ]		
<a href="#">A-Z</a> position	3	varchar(20)		<a href="#">default</a>	[ ]		
<a href="#">A-Z</a> phone	4	varchar(15)		<a href="#">default</a>	[ ]		

## Основы соединения таблиц (JOIN)

```
SELECT c.client_id, c.name, COUNT(car.car_id) AS cars_count
FROM client c
INNER JOIN car ON c.client_id = car.client_id
GROUP BY c.client_id;
```

client 1 X

SELECT c.client\_id, c.name, COUNT(car.car\_id) AS cars\_count | Введите SQL выражение чтобы отфильтровать результат

client_id	name	cars_count
1	Иван	1
2	Мария	1

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

```
SELECT c.client_id, c.name, car.car_id, car.model
FROM client c
LEFT JOIN car ON c.client_id = car.client_id;
```

client(+) 1 X

SELECT c.client\_id, c.name, car.car\_id, car.model FROM | Введите SQL выражение чтобы отфильтровать результат

client_id	name	car_id	model
1	Иван	1	Corolla
2	Мария	2	Solaris

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

```

SELECT car.car_id, car.model
FROM client c
RIGHT JOIN car ON c.client_id = car.client_id
WHERE c.client_id IS NULL;

```

car 1 car 2 X

SELECT car.car\_id, car.model FROM client c RIGHT JC | Введите SQL выражение чтобы отфильт

	123 car_id	A-Z model

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

```

SELECT c.client_id, c.name, car.car_id, car.model
FROM client c
FULL JOIN car ON c.client_id = car.client_id;

```

car 1 client(+) 2 X

SELECT c.client\_id, c.name, car.car\_id, car.model FRO | Введите SQL выражение чтобы отфильт

	123 client_id	A-Z name	123 car_id	A-Z model
1	1	Иван	1	Corolla
2	2	Мария	2	Solaris

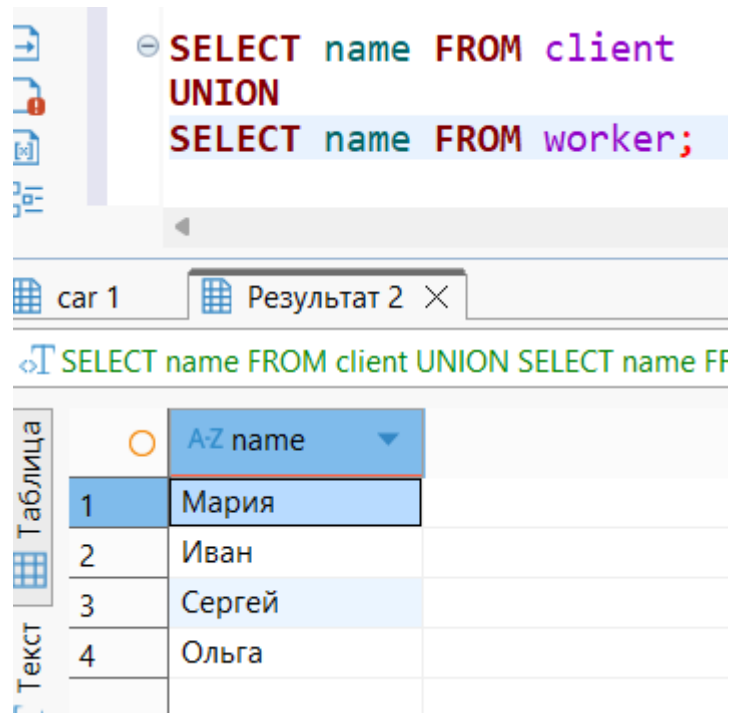
**Рисунок 4 - Получить список клиентов и машин, включая тех клиентов без машин и машины без клиентов (FULL JOIN)**

<pre> SELECT car.car_id, car.model, w.worker_id, w.name FROM car CROSS JOIN worker w; </pre>					
<div>car 1</div> <div>car(+) 2 ×</div>					
<div> <div>SELECT car.car_id, car.model, w.worker_id, w.name</div> <div>Введите SQL выражение чтобы отфильтровать</div> </div>					
Таблица		123 car_id	A-Z model	123 worker_id	A-Z name
	1	1	Corolla	1	Сергей
	2	1	Corolla	2	Ольга
	3	2	Solaris	1	Сергей
	4	2	Solaris	2	Ольга
Текст					

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



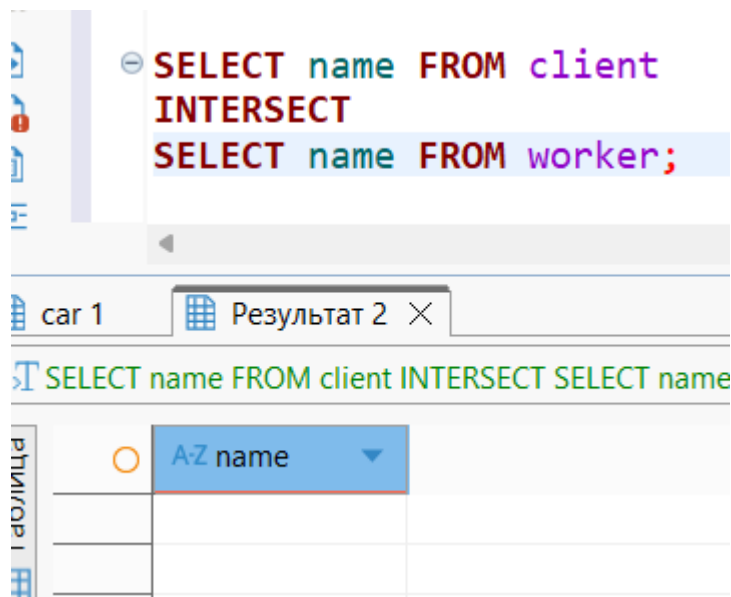
## Продвинутые техники и паттерны соединений



The screenshot shows a database query editor with a query window at the top containing the SQL statement: `SELECT name FROM client UNION SELECT name FROM worker;`. Below the query window, a tab labeled "Результат 2" is active, displaying the query text: `SELECT name FROM client UNION SELECT name FF`. The results are shown in a table with a dropdown menu set to "A-Z name". The table contains four rows of names: Мария, Иван, Сергей, and Ольга.

	A-Z name
1	Мария
2	Иван
3	Сергей
4	Ольга

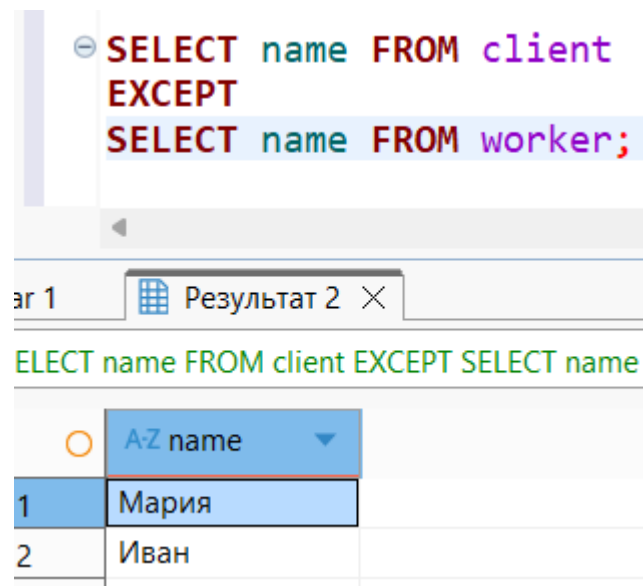
Рисунок 6 - Составить общий список всех имён клиентов и работников (UNION)



The screenshot shows a database query editor with a query window at the top containing the SQL statement: `SELECT name FROM client INTERSECT SELECT name FROM worker;`. Below the query window, a tab labeled "Результат 2" is active, displaying the query text: `SELECT name FROM client INTERSECT SELECT name`. The results are shown in a table with a dropdown menu set to "A-Z name". The table is currently empty.

	A-Z name
--	----------

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



```
SELECT name FROM client  
EXCEPT  
SELECT name FROM worker;
```

ar 1    Результат 2    X

SELECT name FROM client EXCEPT SELECT name

	A-Z name ▼
1	Мария
2	Иван

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.