

ФГБОУ ВО

«Заполярный государственный университет им. Н.М. Федоровского»

Кафедра \_\_ИСиТ\_\_

Специальность \_\_ИЭ-21\_\_

ОТЧЕТ

о выполнении лабораторной работы №4

Выполнил:

Быков В. В.

Дата:

«17 » апреля 2023 г.

## Лабораторная работа №4

## Тема: SQL. Запросы.

**Цель работы:** Получение практических навыков работы с СУБД и языком SQL (оператор SELECT).

**Задание:**

1. разработать запросы к базе данных, созданной и заполненной на предыдущих лабораторных работах, следующих видов:

- запрос с условием на числовые данные ( $>$ ,  $<$ ,  $=$ , between);
- запрос с условием на текстовые данные (LIKE, IN);
- запрос с вычисляемым полем;
- запрос к нескольким таблицам (без явного указания JOIN);
- запрос с агрегирующей функцией (AVG, SUM, COUNT, MIN, MAX);
- запрос с группировкой (GROUP BY);
- запрос с сортировкой (ORDER BY);
- запрос с вложенным подзапросом (не менее 3 видов);
- запрос с оператором UNION;
- запрос с оператором INTERSECT;
- запрос с оператором EXCEPT;
- запрос с выражением CASE;
- запрос с оператором JOIN (пять видов);
- иерархический запрос.

2. Для каждого запроса подписать, что именно он возвращает с учетом предметной области (запросы со смыслом, а не только синтаксически правильные операторы).

**Коды запросов:**

**a.** SELECT \* FROM payments WHERE amount > 4000; - запрос вернет все строки из таблицы payments, где плата за поездку больше 4000

SELECT \* FROM payments WHERE amount < 300; - запрос вернет все строки из таблицы payments, где плата за поездку меньше 300

SELECT \* FROM feedback WHERE stars = 2; - запрос вернет все строки из таблицы feedback, где поставили 2 звезды за поездку

SELECT \* FROM payments WHERE amount BETWEEN 50 AND 190; запрос вернет все строки из таблицы payments, где плата за поездку была между 50 и 190

1

2

3

4

5

SELECT \* FROM payments WHERE amount > 4000;

SELECT \* FROM payments WHERE amount < 300;

Data Output

Сообщения

Notifications

	paymentId [PK] integer	payment_date timestamp without time zone	amount integer	client_id integer	trip_id integer
1	2	2023-03-10 20:30:00	200	2	2
2	3	2023-03-11 20:40:50	150	3	3

1SELECT \* FROM payments WHERE amount BETWEEN 50 AND 190;

Data OutputСообщенияNotifications

payment\_id

[PK] integer

payment\_date

timestamp without time zone

amount

integer

client\_id

integer

trip\_id

integer

1

3

2023-03-11 20:40:50

150

3









3














1 **SELECT** \* **FROM** feedback **WHERE** stars = 2;

Data OutputСообщенияNotifications

	feedback_id [PK] integer	messages text	client_id integer	payment_id integer	stars integer
1	2	У водителя нет прав	2	2	2

SELECT \* FROM feedback WHERE stars IN (5); - запрос вернет все строки из таблицы feedback, где поставили 5 звезд за поездку

Data Output	Сообщения	Notifications	
       			
	<b>client_id</b> [PK] integer	<b>first_name</b> character varying (30)	<b>last_name</b> character varying (30)
1	1	Black	Nigger
2	3	White	Nigger

	Data Output	Сообщения	Notifications		
	       				
	feedback_id [PK] integer 	messages text 	client_id integer 	payment_id integer 	stars integer 
1	1	The best driver ever	1	1	5

```
1 SELECT SUM(amount) AS "Выручка за день" FROM payments;
```

Выручка за день	
1	5350

```
1 SELECT * FROM clients, feedback WHERE feedback.client_id = clients.client_id;
```

<div><div><div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div></div><div></div></div></div>										
client_id	first_name	last_name	phone	car_id	feedback_id	messages	client_id	payment_id	stars	
integer	character varying (30)	character varying (30)	character varying (20)	integer	integer	text	integer	integer	integer	
	1	Black	Nigger		1	The best driver ever	1	1	5	
	2	Rocky	Stone		2	У водителя нет прав	2	2	2	
	3	White	Niqqer		3	Навел суеу и устроил ДТП	3	3		

SELECT SUM(amount) FROM payments; - вернет единственное значение, представляющее сумму всех значений столбца "amount".


SELECT MIN(amount) FROM payments; - вернет минимальное значение столбца "amount" в таблице "payments".

```
1 SELECT AVG(stars) FROM feedback;
```

	avg	
	numeric	
1	2.6666666666666667	

	avg numeric
1	2.6666666666666667

	avg numeric		min integer		count bigint
1	2.6666666666666667	1	150	1	3

Data Output		Сообщения	Notifications
	sum bigint 		
1	5350		

Data Output		Сообщения	Notifications
	sum bigint		
1	5350		

## UPDATE drivers

```
SET age = 50 WHERE driver_id = '3';
```

Группировка водителей старше 30 лет:  
SELECT first\_name, last\_name  
FROM drivers  
WHERE age > 30  
GROUP BY first\_name, last\_name; - будут возвращены только уникальные комбинации имени и фамилии водителей, которые соответствуют условию возраста.

1 SELECT first\_name, last\_name

2 FROM drivers

3 WHERE age > 30

4 GROUP BY first\_name, last\_name;

Data OutputСообщенияNotifications

	first_name character varying (30)	last_name character varying (30)
1	Abdu	Rozik
2	Justin	Bieber

g. Сортировка полученной суммы по возрастанию  
SELECT \* FROM payments ORDER BY amount ASC; - вернет все строки (записи) из таблицы "payments", отсортированные в порядке возрастания (ASC) значения столбца "amount".

1 SELECT \* FROM payments ORDER BY amount ASC;

Data OutputСообщенияNotifications

	payment_id [PK] integer	payment_date timestamp without time zone	amount integer	client_id integer	trip_id integer
1	3	2023-03-11 20:40:50	150	3	3
2	2	2023-03-10 20:30:00	200	2	2
3	1	2023-03-09 20:00:00	5000	1	1

h. Создание дополнительной таблицы:  
CREATE TABLE business\_drivers  
(  
    driver\_id int PRIMARY KEY,  
    first\_name character varying(30)NOT NULL,  
    last\_name character varying(30)NOT NULL,  
    phone character varying(20) NOT NULL,  
    CONSTRAINT business\_driver\_phone UNIQUE (phone)  
);  
ALTER TABLE business\_drivers RENAME COLUMN driver\_id TO business\_driver\_id;  
ALTER TABLE cars ADD COLUMN business\_driver\_id int;  
ALTER TABLE cars ADD FOREIGN KEY (business\_driver\_id) REFERENCES business\_drivers(business\_driver\_id);  
INSERT INTO business\_drivers  
VALUES  
(4,'Bob','Willson','+19132221301'),  
(5,'Tony','Montana','+19132221302'),  
(6,'Vito','Scaletta','+19132221303');  
INSERT INTO cars (car\_id, model, business\_driver\_id)  
VALUES  
(4,'Bentley Continental',4),  
(5,'Rolls-Royce Cullinan',5),  
(6,'Mercedes-Benz Maybach GLS 600',4),  
(7,'Велосипед четырехколесный SPORT',4),  
(8,'AUDI S5',6),  
(9,'Porsche 911',6);

SELECT first\_name,last\_name, (  
    SELECT COUNT(\*) FROM trips WHERE client\_id = clients.client\_id  
)  
AS "Кол-во поездок" FROM clients;- вернет список всех клиентов в таблице "clients" с дополнительным столбцом "Кол-во поездок", который показывает, сколько поездок каждый клиент совершил.

SELECT first\_name, last\_name, age  
FROM drivers  
WHERE age > (

```
SELECT AVG(age) FROM drivers
```

); - вернет список всех водителей, у которых возраст выше среднего возраста всех водителей в таблице "drivers".

```
SELECT * FROM clients
WHERE client_id IN (
    SELECT payment_id FROM feedback WHERE stars < 5
);
```

- вернет список всех клиентов в таблице "clients", у которых есть отзыв (запись в таблице "feedback"), связанный с платежом (записью в таблице "payments"), и в этом отзыве была оценка меньше 5 звезд.

```
1 SELECT * FROM clients
2 WHERE client_id IN (
3     SELECT payment_id FROM feedback WHERE stars < 5
4 );
```

	client_id [PK] integer	first_name character varying (30)	last_name character varying (30)	phone character varying (20)	car_id integer
1	2	Rocky	Stone	+79132221302	2
2	3	White	Nigger	+79132221303	3

```
1 SELECT first_name, last_name, (
2     SELECT COUNT(*) FROM trips WHERE client_id = clien
3 )
4 AS "Кол-во поездок" FROM clients;
```

	first_name character varying (30)	last_name character varying (30)	Кол-во поездок bigint
1	Black	Nigger	1
2	Rocky	Stone	1
3	White	Nigger	1

```
1 SELECT first_name, last_name, age
2 FROM drivers
3 WHERE age > (
4     SELECT AVG(age) FROM drivers
5 );
```

	first_name character varying (30)	last_name character varying (30)	age integer
1	Justin	Bieber	50

**i.** Создание дополнительного поля age к таблицам business\_drivers и clients:

```
ALTER TABLE clients ADD COLUMN age int;
UPDATE clients
SET age = 29 WHERE client_id = '1';
UPDATE clients
SET age = 72 WHERE client_id = '2';
UPDATE clients
SET age = 19 WHERE client_id = '3';
ALTER TABLE business_drivers ADD COLUMN age int;
UPDATE business_drivers
SET age = 31 WHERE driver_id = '4';
UPDATE business_drivers
SET age = 65 WHERE driver_id = '5';
UPDATE business_drivers
SET age = 23 WHERE driver_id = '6';
```

```
SELECT first_name, last_name, age
FROM drivers
WHERE age > 40
UNION
SELECT first_name, last_name, age
FROM business_drivers
WHERE age > 40 - вернет список всех водителей из таблиц "drivers" и "business_drivers", у которых возраст больше 40 лет.
```

```
1 SELECT first_name, last_name, age
2 FROM drivers
3 WHERE age > 40
4 UNION
5 SELECT first_name, last_name, age
6 FROM business_drivers
7 WHERE age > 40
8
```

	first_name character varying (30)	last_name character varying (30)	age integer
1	Tony	Montana	65
2	Justin	Bieber	50

**j.** SELECT age  
FROM drivers  
INTERSECT  
SELECT age  
FROM business\_drivers - вернет список возрастов, которые присутствуют и в таблице "drivers", и в таблице "business\_drivers".

```
1 SELECT age
2 FROM drivers
3 INTERSECT
4 SELECT age
5 FROM business_drivers
```

Data Output			Сообщения	Notifications
	age	integer		
1		31		
2		23		

**k.** SELECT first\_name, last\_name, age  
FROM drivers  
EXCEPT  
SELECT first\_name, last\_name, age  
FROM business\_drivers - вернет список водителей из таблицы "drivers", чьи данные (имя, фамилия, возраст) не встречаются в таблице "business\_drivers".

```
1 SELECT first_name, last_name, age
2 FROM drivers
3 EXCEPT
4 SELECT first_name, last_name, age
5 FROM business_drivers
```

Data Output				Сообщения	Notifications
	first_name	last_name	age		
	character varying (30)	character varying (30)	integer		
1	Abdu	Rozik	31		
2	Hasbulla	Magomedov	23		
3	Justin	Bieber	50		

**l.** SELECT first\_name, last\_name, age,  
CASE  
    WHEN age >= 60 THEN 'пожилой водитель'  
    WHEN age >= 35 AND age <= 59 THEN 'водитель среднего возраста'  
    WHEN age >= 18 AND age <= 34 THEN 'молодой водитель'  
END AS "status"  
FROM drivers; - вернет список водителей из таблицы "drivers" с добавлением столбца "status", который будет содержать информацию о возрасте водителя в виде текстового описания ("молодой водитель", "водитель среднего возраста", "пожилой водитель").

```
1 SELECT first_name, last_name, age,
2 CASE
3     WHEN age >= 60 THEN 'пожилой водитель'
4     WHEN age >= 35 AND age <= 59 THEN 'водитель среднего возраста'
5     WHEN age >= 18 AND age <= 34 THEN 'молодой водитель'
6 END AS "status"
7 FROM drivers;
```

Data Output					Сообщения	Notifications
	first_name	last_name	age	status		
	character varying (30)	character varying (30)	integer	text		
1	Hasbulla	Magomedov	23	молодой водитель		
2	Abdu	Rozik	31	молодой водитель		
3	Justin	Bieber	50	водитель среднего возраста		

**m. SELECT \***  
FROM drivers  
INNER JOIN business\_drivers ON drivers.age = business\_drivers.age; - вернет только те строки, в которых возраст водителя из таблицы "drivers" соответствует возрасту водителя из таблицы "business\_drivers".

SELECT \*  
FROM drivers  
LEFT JOIN business\_drivers ON drivers.age = business\_drivers.age; - вернет все строки из таблицы "drivers", а также соответствующие строки из таблицы "business\_drivers", при этом если нет соответствующей строки в таблице "business\_drivers", то значения возвращаются как NULL.

SELECT \*  
FROM drivers  
RIGHT JOIN business\_drivers ON drivers.age = business\_drivers.age; - вернет все строки из таблицы "business\_drivers", а также соответствующие строки из таблицы "drivers", при этом если нет соответствующей строки в таблице "drivers", то значения возвращаются как NULL.

SELECT \*  
FROM drivers  
FULL OUTER JOIN business\_drivers ON drivers.age = business\_drivers.age; - вернет все строки из таблицы "drivers" и все строки из таблицы "business\_drivers", при этом если нет соответствующей строки в одной из таблиц, то значения возвращаются как NULL.

SELECT \*  
FROM drivers  
CROSS JOIN clients; - вернет все возможные комбинации строк из таблиц "drivers" и "clients", т.е. каждая строка из таблицы "drivers" будет объединена со всеми строками из таблицы "clients".

1 SELECT \*

2 FROM drivers

3 INNER JOIN business\_drivers ON drivers.age = business\_drivers.age;

Data Output

Сообщения

Notifications

	phone	age	driver_id	first_name	last_name	phone	age	
	character varying (20)	integer	integer	character varying (30)	character varying (30)	character varying (20)	integer	
1	lov	+79998887766	23	6	Vito	Scaletta	+19132221303	23
2		+79132221202	31	4	Bob	Willson	+19132221301	31

1 SELECT \*

2 FROM drivers

3 LEFT JOIN business\_drivers ON drivers.age = business\_drivers.age;

Data Output

Сообщения

Notifications

	arying (30)	phone character varying (20)	age integer	driver_id integer	first_name character varying (30)	last_name character varying (30)	phone character varying (20)	age integer
1	ov	+79998887766	23	6	Vito	Scaletta	+19132221303	23
2		+79132221202	31	4	Bob	Willson	+19132221301	31
3		+79132221203	50	[null]	[null]	[null]	[null]	[null]

1 SELECT \*

2 FROM drivers

3 RIGHT JOIN business\_drivers ON drivers.age = business\_drivers.age;

Data Output

Сообщения

Notifications

	phone	age	driver_id	first_name	last_name	phone	age	
	character varying (20)	integer	integer	character varying (30)	character varying (30)	character varying (20)	integer	
1		+79132221202	31	4	Bob	Willson	+19132221301	31
2		[null]	[null]	5	Tony	Montana	+19132221302	65
3	ov	+79998887766	23	6	Vito	Scaletta	+19132221303	23

Запрос

История запросов

1 SELECT \*

2 FROM drivers

3 FULL OUTER JOIN business\_drivers ON drivers.age = business\_drivers.age;

Data Output

Сообщения

Notifications

	phone	age	driver_id	first_name	last_name	phone	age	
	character varying (20)	integer	integer	character varying (30)	character varying (30)	character varying (20)	integer	
1	ov	+79998887766	23	6	Vito	Scaletta	+19132221303	23
2		+79132221202	31	4	Bob	Willson	+19132221301	31
3		+79132221203	50	[null]	[null]	[null]	[null]	[null]
4		[null]	[null]	5	Tony	Montana	+19132221302	65

1 SELECT \*

2 FROM drivers

3 CROSS JOIN clients;

Data Output

Сообщения

Notifications

	phone character varying (20)	age integer	client_id integer	first_name character varying (30)	last_name character varying (30)	phone character varying (20)	car_id integer	age integer
1	+79998887766	23	1	Black	Nigger	+79132221301	1	29
2	+79998887766	23	2	Rocky	Stone	+79132221302	2	72
3	+79998887766	23	3	White	Nigger	+79132221303	3	19
4	+79132221202	31	1	Black	Nigger	+79132221301	1	29
5	+79132221202	31	2	Rocky	Stone	+79132221302	2	72
6	+79132221202	31	3	White	Nigger	+79132221303	3	19
7	+79132221203	50	1	Black	Nigger	+79132221301	1	29
8	+79132221203	50	2	Rocky	Stone	+79132221302	2	72
9	+79132221203	50	3	White	Nigger	+79132221303	3	19

п. Иерархический запрос для поиска всех начальников конкретного сотрудника:

```
CREATE TABLE taxi_structure
(
    structure_id int PRIMARY KEY,
    employee_name character varying(50) NOT NULL,
    manager int REFERENCES taxi_structure
);
INSERT INTO taxi_structure
VALUES
(1, 'Steve Jobs', null),
(2, 'Bob', 1),
(3, 'John', 1),
(4, 'Ryan Gosling', 2),
(5, 'Connor McGregor', 4),
(6, 'Will Smith', 7),
(7, 'Big Bob', 4),
(8, 'Lil Bob', 7),
(9, 'Chief Keef', 6),
(10, 'Vodila', 2);

WITH RECURSIVE managers AS (
    SELECT structure_id, employee_name, manager
    FROM taxi_structure
    WHERE employee_name = 'Chief Keef'
    UNION ALL
    SELECT i.structure_id, i.employee_name, i.manager
    FROM taxi_structure AS i
    INNER JOIN managers AS e ON i.structure_id = e.manager
)
SELECT structure_id, employee_name, manager
FROM managers;
```

- запрос используется для построения иерархии руководства, начиная с сотрудника “Chief Keef”. запрос вернет столбцы "structure\_id", "employee\_name" и "manager" для всех руководителей в иерархии данного сотрудника.

1 WITH RECURSIVE managers AS (

2 SELECT structure\_id, employee\_name, manager

3 FROM taxi\_structure

4 WHERE employee\_name = 'Chief Keef'

5 UNION ALL

6 SELECT i.structure\_id, i.employee\_name, i.manager

7 FROM taxi\_structure AS i

8 INNER JOIN managers AS e ON i.structure\_id = e.manager

9 )

10 SELECT structure\_id, employee\_name, manager

11 FROM managers;

Data Output

	structure_id integer	employee_name character varying (50)	manager integer
1	9	Chief Keef	6
2	6	Will Smith	7
3	7	Big Bob	4
4	4	Ryan Gosling	2
5	2	Bob	1
6	1	Steve Jobs	[null]