

Отчет БД Практика Лабораторная 20

Часть 1.1

```
-- 1. Выборка из одной таблицы.  
-- 1.1 Выбрать из произвольной таблицы данные и  
-- отсортировать их по двум произвольным  
-- имеющимся в таблице признакам (разные направления сортировки)
```

```
Select  
    A_ID,  
    balance,  
    debtor_status  
FROM accounts  
ORDER BY DEBTOR_STATUS DESC, A_ID ASC
```

1.



	a_id [PK] bigint	balance numeric (15,2)	debtor_status boolean
1	2	-44.00	true
2	1	422.50	false
3	3	237.00	false
4	4	85.00	false
5	5	4107.50	false
6	6	2075.00	false
7	7	19.00	false
8	8	322.00	false
9	9	297.50	false
10	10	6507.50	false



-- 1.2 Выбрать из произвольной таблицы те записи, которые удовлетворяют
-- условию отбора (where). Привести 2-3 запроса.

```
SELECT
    A_ID,
    BALANCE
FROM ACCOUNTS
WHERE DEBTOR_STATUS = FALSE
```

```
SELECT
    A_ID,
    BALANCE
FROM ACCOUNTS
WHERE (A_ID % 3 = 0) AND BALANCE > 200
```

2.

	a_id [PK] bigint 	balance numeric (15,2) 
1	1	422.50
2	3	237.00
3	4	85.00
4	5	4107.50
5	6	2075.00
6	7	19.00
7	8	322.00
8	9	297.50
9	10	6507.50

	a_id [PK] bigint 	balance numeric (15,2) 
1	3	237.00
2	6	2075.00
3	9	297.50

```
-- 1.3 Привести примеры 2-3 запросов с использованием агрегатных функций
-- (count, max, sum и др.) с группировкой и без группировки.
```




```
-- С группировкой
```

```
SELECT
    SUM(BALANCE),
    DEBTOR_STATUS
FROM ACCOUNTS
GROUP BY DEBTOR_STATUS
```

```
--Без группировки
```

```
SELECT
    COUNT(*)
FROM ACCOUNTS
WHERE BALANCE > 1000
```

3.

	sum numeric 	debtor_status boolean 		count bigint 
1	14073.00	false		
2	-44.00	true	1	3

```
-- 1.4 Привести примеры подведения подытога с использованием
-- GROUP BY [ALL] [ CUBE | ROLLUP](2-3 запроса).
-- В ROLLUP и CUBE использовать не менее 2-х столбцов.
```

```
SELECT
CASE WHEN GROUPING(legal_entity_name) = 1 THEN
    'ALL_TYPES' ELSE
        legal_entity_name END
    as entity_type,
COUNT(*) as user_count,
AVG(LENGTH(fio)) as avg_name_length
FROM users
GROUP BY ALL (legal_entity_name)
ORDER BY entity_type;

SELECT
COALESCE(transaction_type, 'ALL_TYPES') as type,
COALESCE(TO_CHAR(transaction_date, 'YYYY-MM'), 'ALL_MONTHS') as month,
COUNT(*) as count,
SUM(amount) as total
FROM transactions
GROUP BY CUBE (transaction_type, TO_CHAR(transaction_date, 'YYYY-MM'))
ORDER BY type, month;
```

4.

	entity_type text	user_count bigint	avg_name_length numeric
1	АО "БизнесКонсалт"	1	18.0000000000000000
2	ИП Семенов А.В.	1	15.0000000000000000
3	ООО "СтальПроект"	1	17.0000000000000000
4	ООО "ТелекомСерв...	1	19.0000000000000000
5	[null]	6	22.6666666666666667

	type text	month text	count bigint	total numeric
1	ALL_TYPES	2024-01	30	25471.00
2	ALL_TYPES	ALL_MONTHS	30	25471.00
3	expense	2024-01	20	5721.00
4	expense	ALL_MONTHS	20	5721.00
5	income	2024-01	10	19750.00
6	income	ALL_MONTHS	10	19750.00

```
-- 1.5 Выбрать из таблиц информацию об объектах,
-- в названиях которых нет заданной последовательности букв (LIKE).

SELECT
    U_ID,
    FIO
FROM USERS
WHERE FIO LIKE '%ОВ%'
```

5.

	u_id [PK] bigint	fio text
1	1	Иванов Иван Иванович
2	2	Петрова Анна Сергеевна
3	3	Сидоров Михаил Петрович
4	4	Козлова Елена Викторовна
5	7	Николаев Дмитрий Олегов...
6	8	ИП Семенов А.В.
7	9	Волкова Ольга Игоревна

Часть 1.2

```
-- 2.1 Вывести информацию подчиненной (дочерней) таблицы, заменяя коды
-- (значения внешних ключей) соответствующими символьными значениями из
-- родительских таблиц. Привести 2-3 запроса с использованием классического
-- подхода соединения таблиц (where).

SELECT
    c.call_id,
    u.fio      as user_name,
    ct.city_id,
    c.call_date,
    c.duration
FROM
    calls c,
    users u,
    city ct
WHERE
    c.user_id = u.U_ID
AND
    c.city_id = ct.CITY_ID
ORDER BY
    c.call_date ASC
LIMIT 5;

SELECT
    t.T_ID as transaction_id,
    u.fio  as user_name,
    t.amount,
    t.transaction_date
FROM
    transactions t,
    accounts a,
    users u
WHERE
    t.A_ID = a.A_ID
AND
    a.A_ID = u.U_ID
ORDER BY
    t.transaction_date DESC
LIMIT 5;
```

1.

	call_id bigint	user_name text	city_id bigint	call_date timestamp without time zone	duration interval
1	1	Иванов Иван Иванович	1	2024-01-10 09:05:00	00:15:00
2	3	Петрова Анна Сергеевна	3	2024-01-10 11:30:00	00:25:00
3	2	Иванов Иван Иванович	2	2024-01-10 14:20:00	00:08:00
4	5	Сидоров Михаил Петров...	5	2024-01-11 10:15:00	00:30:00
5	4	Петрова Анна Сергеевна	4	2024-01-11 16:45:00	00:12:00

	transaction_id bigint	user_name text	amount numeric (15,2)	transaction_date timestamp without time zone
1	30	ООО "СтальПроект"	292.50	2024-01-19 16:30:00
2	28	Волкова Ольга Игорев...	22.50	2024-01-18 21:30:00
3	29	ООО "СтальПроект"	1200.00	2024-01-18 12:30:00
4	27	Волкова Ольга Игорев...	80.00	2024-01-17 19:32:00
5	26	ИП Семенов А.В.	98.00	2024-01-17 12:43:00

2.

-- 2.2. Реализовать запросы пункта 2.1 через внутреннее соединение inner join.

```
SELECT
    c.call_id,
    u.fio as user_name,
    ct.city_id,
    c.call_date,
    c.duration
FROM calls c
INNER JOIN users u ON c.user_id = u.U_ID
INNER JOIN city ct ON c.city_id = ct.CITY_ID
ORDER BY c.call_date ASC
LIMIT 5;

SELECT
    t.T_ID as transaction_id,
    u.fio as user_name,
    t.amount,
    t.transaction_date
FROM transactions t
INNER JOIN accounts a ON t.A_ID = a.A_ID
INNER JOIN users u ON a.A_ID = u.U_ID
ORDER BY t.transaction_date DESC
LIMIT 5;
```

Результат тот же, скрин не прилагается

3.

-- 2.3. Левое внешнее соединение left join. Привести 2-3 запроса.

```
SELECT
    ct.city_id,
    ct.n_cost,
    ct.d_cost,
    c.call_id,
    c.call_date
FROM city ct
LEFT JOIN calls c ON ct.city_id = c.city_id
ORDER BY ct.city_id
LIMIT 5;
```

-- Опять же, запрос показывает все счета, включая те,
-- по которым не было транзакций (У меня таких нет)

```
SELECT
    a.A_ID,
    a.balance,
    t.transaction_type,
    t.amount
FROM accounts a
LEFT JOIN transactions t ON a.A_ID = t.A_ID
ORDER BY a.A_ID
LIMIT 5;
```

	city_id bigint	n_cost numeric (10,2)	d_cost numeric (10,2)	call_id bigint	call_date timestamp without time zone
1	1	2.50	2.00	1	2024-01-10 09:05:00
2	1	2.50	2.00	17	2024-01-17 19:00:00
3	1	2.50	2.00	8	2024-01-13 17:25:00
4	2	5.00	4.00	13	2024-01-15 18:10:00
5	2	5.00	4.00	2	2024-01-10 14:20:00

	a_id bigint	balance numeric (15,2)	transaction_type text	amount numeric (15,2)
1	1	422.50	expense	40.00
2	1	422.50	expense	37.50
3	1	422.50	income	500.00
4	2	-44.00	income	300.00
5	2	-44.00	expense	144.00

4.

```
-- 2.4. Правое внешнее соединение right join. Привести 2-3 запроса
-- Почти аналогичные запросы, фантазии у меня маловато
```

```
SELECT
    t.T_ID,
    t.amount,
    t.transaction_date,
    u.fio,
    u.inn
FROM transactions t
RIGHT JOIN accounts a ON t.A_ID = a.A_ID
RIGHT JOIN users u ON a.A_ID = u.U_ID
LIMIT 5;
```

```
SELECT
    c.call_id,
    c.call_date,
    c.duration,
    ct.city_id,
    ct.n_cost,
    ct.d_cost
FROM calls c
RIGHT JOIN city ct ON c.city_id = ct.city_id
ORDER BY ct.city_id
LIMIT 5;
```

	t_id bigint 🔒	amount numeric (15,2) 🔒	transaction_date timestamp without time zone 🔒	fio text 🔒	inn text 🔒
1	1	500.00	2024-01-01 10:00:00	Иванов Иван Иванович	12345678...
2	11	37.50	2024-01-10 09:20:00	Иванов Иван Иванович	12345678...
3	12	40.00	2024-01-10 14:28:00	Иванов Иван Иванович	12345678...
4	2	300.00	2024-01-01 11:00:00	Петрова Анна Сергеев...	23456789...
5	13	200.00	2024-01-10 11:55:00	Петрова Анна Сергеев...	23456789...

	call_id bigint 🔒	call_date timestamp without time zone 🔒	duration interval 🔒	city_id bigint 🔒	n_cost numeric (10,2) 🔒	d_cost numeric (10,2) 🔒
1	1	2024-01-10 09:05:00	00:15:00	1	2.50	2.00
2	17	2024-01-17 19:00:00	00:32:00	1	2.50	2.00
3	8	2024-01-13 17:25:00	00:22:00	1	2.50	2.00
4	13	2024-01-15 18:10:00	00:05:00	2	5.00	4.00
5	2	2024-01-10 14:20:00	00:08:00	2	5.00	4.00

```
-- 2.5. Привести примеры 2-3 запросов с использованием агрегатных функций
-- и группировки.
```

```
SELECT
    a.debtor_status,
    COUNT
        (t.T_ID) as transaction_count,
    SUM(
        CASE WHEN t.transaction_type = 'income' THEN
            t.amount ELSE
                0
        END) as total_income,
    SUM(CASE WHEN t.transaction_type = 'expense' THEN
        t.amount ELSE 0 END) as total_expense,
    SUM
        (t.amount) as Трафик,
    COUNT(DISTINCT t.A_ID) as users_
FROM accounts a
INNER JOIN transactions t ON a.A_ID = t.A_ID
GROUP BY a.debtor_status
ORDER BY debtor_status;

SELECT
    ct.city_id,
    COUNT(c.call_id) as total_calls,
    SUM(EXTRACT(EPOCH FROM c.duration)/60) as total_minutes,
    AVG(EXTRACT(EPOCH FROM c.duration)/60) as avg_minutes,
    MAX(EXTRACT(EPOCH FROM c.duration)/60) as max_minutes
FROM city ct
INNER JOIN calls c ON ct.city_id = c.city_id
GROUP BY ct.city_id
ORDER BY total_calls DESC
LIMIT 5;
```

5.

	debtor_status boolean	transaction_count bigint	total_income numeric	total_expense numeric	Трафик numeric	users_ bigint
1	false	27	19450.00	5377.00	24827.00	9
2	true	3	300.00	344.00	644.00	1

	city_id [PK] bigint	total_calls bigint	total_minutes numeric	avg_minutes numeric	max_minutes numeric
1	1	3	69.0000000000000000	23.0000000000000000	32.0000000000000000
2	2	2	13.0000000000000000	6.5000000000000000	8.0000000000000000
3	3	2	32.0000000000000000	16.0000000000000000	25.0000000000000000
4	5	2	65.0000000000000000	32.5000000000000000	35.0000000000000000
5	8	2	180.0000000000000000	90.0000000000000000	120.0000000000000000

```
-- 2.6. Привести примеры 2-3 запросов с использованием
-- группировки и условия отбора групп (Having).
```

```
SELECT
    u.fio,
    u.legal_entity_name,
    COUNT(t.T_ID) as total_transactions,
    SUM(t.amount) as total_amount,
    AVG(t.amount) as avg_transaction,
    MAX(t.amount) as max_transaction,
    MIN(t.amount) as min_transaction
FROM users u
INNER JOIN transactions t ON u.U_ID = t.A_ID
GROUP BY u.fio, u.legal_entity_name
HAVING COUNT(t.T_ID) > 0
ORDER BY total_amount DESC
LIMIT 5;
```

```
SELECT
    u.fio,
    COUNT(t.T_ID) as transaction_count,
    SUM(t.amount) as total_amount
FROM users u
INNER JOIN transactions t ON u.U_ID = t.A_ID
GROUP BY u.fio
HAVING COUNT(t.T_ID) > 1
    AND SUM(t.amount) > 4000
ORDER BY transaction_count DESC
LIMIT 5;
```

6.

	fio text	legal_entity_name text	total_transactions bigint	total_amount numeric	avg_transaction numeric	max_transaction numeric	min_transaction numeric
1	ООО "СтальПроект"	ООО "СтальПроект"	3	9492.50	3164.1666666666666667	8000.00	292.50
2	ООО "ТелекомСервис"	ООО "ТелекомСерв...	3	5892.50	1964.1666666666666667	5000.00	292.50
3	АО "БизнесКонсалт"	АО "БизнесКонсалт"	3	3925.00	1308.3333333333333333	3000.00	400.00
4	ИП Семенов А.В.	ИП Семенов А.В.	3	2678.00	892.6666666666666667	1500.00	98.00
5	Сидоров Михаил Петров...	[null]	3	1263.00	421.0000000000000000	750.00	63.00

	fio text	transaction_count bigint	total_amount numeric
1	ООО "СтальПроект"	3	9492.50
2	ООО "ТелекомСерв...	3	5892.50

-- 2.7. Привести примеры 3-4 вложенных
 -- (соотнесенных, с использованием IN, EXISTS) запросов.

```
SELECT
  u.fio,
  u.inn,
  u.adress
FROM users u
WHERE u.U_ID IN (
  SELECT DISTINCT c.user_id
  FROM calls c
  WHERE c.duration > INTERVAL '20 minutes'
);
```

```
SELECT
  u.fio,
  a.balance
FROM users u
INNER JOIN accounts a ON u.U_ID = a.A_ID
WHERE u.U_ID IN (
  SELECT A_ID
  FROM transactions
  GROUP BY A_ID
  HAVING SUM(amount) > 1000
);
```

7.

	fio text	inn text	adress text
1	Петрова Анна Сергеевна	23456789...	СПб, Невский пр-т, 25
2	Сидоров Михаил Петров...	34567890...	Казань, ул. Баумана, 15
3	Козлова Елена Викторов...	45678901...	Новосибирск, ул. Кирова, 8
4	ООО "ТелекомСервис"	56789012...	Москва, ул. Тверская, 10
5	АО "БизнесКонсалт"	67890123...	Екатеринбург, пр-т Ленина, 50
6	ИП Семенов А.В.	89012345...	Краснодар, ул. Красная, 100
7	Волкова Ольга Игоревна	90123456...	Сочи, ул. Курортная, 5
8	ООО "СтальПроект"	01234567...	Челябинск, ул. Metallургов, ...

	fio text	balance numeric (15,2)
1	Сидоров Михаил Петров...	237.00
2	ООО "ТелекомСервис"	4107.50
3	АО "БизнесКонсалт"	2075.00
4	ИП Семенов А.В.	322.00
5	ООО "СтальПроект"	6507.50

```

CREATE VIEW view_ AS(
  SELECT
    c.call_id,
    u.fio      as user_name,
    ct.city_id,
    c.call_date,
    c.duration
  FROM
    calls c,
    users u, |
    city ct
  WHERE
    c.user_id = u.U_ID
  AND
    c.city_id = ct.CITY_ID
  ORDER BY
    c.call_date ASC
  LIMIT 5
);

SELECT * FROM view_

CREATE view view_1 as (
  SELECT
    t.T_ID as transaction_id,
    u.fio  as user_name,
    t.amount,
    t.transaction_date
  FROM
    transactions t,
    accounts a,
    users u
  WHERE
    t.A_ID = a.A_ID
  AND
    a.A_ID = u.U_ID
  ORDER BY
    t.transaction_date DESC
  LIMIT 5);

SELECT * FROM view_1

```




1.3.1.



1.3.2.

--3.2 Привести примеры использования общетабличных выражений (CTE) (2-3 запроса)

```
WITH avg_call_duration AS (
    SELECT AVG(EXTRACT(EPOCH FROM duration)) as avg_seconds
    FROM calls
)
SELECT
    c.call_id,
    c.duration,
    (EXTRACT(EPOCH FROM c.duration)) as duration_seconds
FROM calls c, avg_call_duration acd
WHERE EXTRACT(EPOCH FROM c.duration) > acd.avg_seconds
LIMIT 5;

WITH debtors AS (
    SELECT u.fio, a.balance
    FROM users u
    INNER JOIN accounts a ON u.U_ID = a.A_ID
    WHERE a.debtor_status = true
)
SELECT *
FROM debtors
LIMIT 5;
```

	call_id [PK] bigint 	duration interval 	duration_seconds numeric 
1	7	00:40:00	2400.000000
2	9	00:45:00	2700.000000
3	10	01:00:00	3600.000000
4	12	00:35:00	2100.000000
5	15	01:30:00	5400.000000

	fio text 	balance numeric (15,2) 
1	Петрова Анна Сергеев...	-44.00

1.4.1.

```
-- 4.1 Привести примеры 3-4 запросов с использованием
-- ROW_NUMBER, RANK, DENSE_RANK (с PARTITION BY и без)

-- Самая простая нумерация
SELECT
    fio,
    inn,
    ROW_NUMBER() OVER (ORDER BY fio) as row_num
FROM users
LIMIT 10;

--Разница была бы видна, если бы были абоненты с равным значением баланса
SELECT
    u.fio,
    a.balance,
    RANK() OVER (ORDER BY a.balance DESC) as rank_balance,
    DENSE_RANK() OVER (ORDER BY a.balance DESC) as dense_rank_balance
FROM users u
INNER JOIN accounts a ON u.U_ID = a.A_ID
WHERE a.balance > 0
LIMIT 10;

-- Нумерация внутри каждого типа транзакции, сортировка по сумме
SELECT
    u.fio,
    t.transaction_type,
    t.amount,
    t.transaction_date,
    ROW_NUMBER() OVER (PARTITION BY t.transaction_type ORDER BY t.amount DESC) as type_rank
FROM transactions t
INNER JOIN users u ON t.A_ID = u.U_ID
LIMIT 15;
```

	fio text	inn text	row_num bigint
1	АО "БизнесКонсалт"	67890123...	1
2	Волкова Ольга Игоревна	90123456...	2
3	Иванов Иван Иванович	12345678...	3
4	ИП Семенов А.В.	89012345...	4
5	Козлова Елена Викторовна	45678901...	5
6	Николаев Дмитрий Олегов...	78901234...	6
7	ООО "СтальПроект"	01234567...	7
8	ООО "ТелекомСервис"	56789012...	8
9	Петрова Анна Сергеевна	23456789...	9
10	Сидоров Михаил Петрович	34567890...	10

	fio text	balance numeric (15,2)	rank_balance bigint	dense_rank_balance bigint
1	ООО "СтальПроект"	6507.50	1	1
2	ООО "ТелекомСервис"	4107.50	2	2
3	АО "БизнесКонсалт"	2075.00	3	3
4	Иванов Иван Иванович	422.50	4	4
5	ИП Семенов А.В.	322.00	5	5
6	Волкова Ольга Игоревна	297.50	6	6
7	Сидоров Михаил Петрович	237.00	7	7
8	Козлова Елена Викторовна	85.00	8	8
9	Николаев Дмитрий Олегов...	19.00	9	9

	fio text	transaction_type text	amount numeric (15,2)	transaction_date timestamp without time zone	type_rank bigint
1	ООО "СтальПроект"	expense	1200.00	2024-01-18 12:30:00	1
2	ИП Семенов А.В.	expense	1080.00	2024-01-16 10:15:00	2
3	ООО "ТелекомСервис"	expense	600.00	2024-01-14 12:30:00	3
4	АО "БизнесКонсалт"	expense	525.00	2024-01-15 16:55:00	4
5	Сидоров Михаил Петрович	expense	450.00	2024-01-11 10:45:00	5
6	АО "БизнесКонсалт"	expense	400.00	2024-01-14 14:20:00	6
7	ООО "ТелекомСервис"	expense	292.50	2024-01-13 09:45:00	7
8	ООО "СтальПроект"	expense	292.50	2024-01-19 16:30:00	8


```

-- 5.1 Привести примеры 3-4 запросов с использованием
-- UNION / UNION ALL, EXCEPT, INTERSECT.
-- Данные в одном из запросов отсортируйте по произвольному признаку.

-- Пользователи, которые совершали звонки и имеют транзакции
SELECT u.U_ID, u.fio
FROM users u
INNER JOIN calls c ON u.U_ID = c.user_id

INTERSECT

SELECT u.U_ID, u.fio
FROM users u
INNER JOIN transactions t ON u.U_ID = t.A_ID;

-- Пользователи с положительным балансом ИЛИ должники
SELECT u.fio, a.balance, a.debtor_status
FROM users u
INNER JOIN accounts a ON u.U_ID = a.A_ID
WHERE a.balance > 1000

UNION

SELECT u.fio, a.balance, a.debtor_status
FROM users u
INNER JOIN accounts a ON u.U_ID = a.A_ID
WHERE a.debtor_status = true
ORDER BY balance DESC;

-- Все транзакции больше 500 и все транзакции типа 'expense'
SELECT 'large_transaction' as type, T_ID, amount, transaction_type
FROM transactions
WHERE amount > 500

UNION ALL

SELECT 'expense_transaction' as type, T_ID, amount, transaction_type
FROM transactions
WHERE transaction_type = 'expense'
LIMIT 10;

```

1.5.1.

	u_id bigint	fio text
1	4	Козлова Елена Викторовна
2	9	Волкова Ольга Игоревна
3	5	ООО "ТелекомСервис"
4	1	Иванов Иван Иванович
5	2	Петрова Анна Сергеевна
6	8	ИП Семенов А.В.
7	7	Николаев Дмитрий Олегов...
8	6	АО "БизнесКонсалт"
9	10	ООО "СтальПроект"
10	3	Сидоров Михаил Петрович

	fio text	balance numeric (15,2)	debtor_status boolean
1	ООО "СтальПроект"	6507.50	false
2	ООО "ТелекомСервис"	4107.50	false
3	АО "БизнесКонсалт"	2075.00	false
4	Петрова Анна Сергеев...	-44.00	true

	type text	t_id bigint	amount numeric (15,2)	transaction_type text
1	large_transaction	3	750.00	income
2	large_transaction	5	5000.00	income
3	large_transaction	6	3000.00	income
4	large_transaction	8	1500.00	income
5	large_transaction	10	8000.00	income
6	large_transaction	20	600.00	expense
7	large_transaction	22	525.00	expense
8	large_transaction	25	1080.00	expense
9	large_transaction	29	1200.00	expense
10	expense_transacti...	11	37.50	expense

1.6.1.

```
--6.1 Привести примеры получения сводных (итоговых) таблиц с использованием CASE

-- Позволяет посмотреть сводку за нужную дату
SELECT
    u.fio,
    SUM(CASE WHEN t.transaction_type = 'income' THEN t.amount ELSE 0 END) as total_income,
    SUM(CASE WHEN t.transaction_type = 'expense' THEN t.amount ELSE 0 END) as total_expense,
    SUM(t.amount) as net_balance
FROM users u
INNER JOIN transactions t ON u.U_ID = t.A_ID
WHERE t.transaction_date >= CURRENT_DATE - INTERVAL '1000 days'
GROUP BY u.fio
LIMIT 10;

-- Анализ продолжительности звонков по абонентам
SELECT
    u.fio,
    COUNT(*) as total_calls,
    COUNT(CASE WHEN c.duration < INTERVAL '1 minute' THEN 1 END) as short_calls,
    COUNT(CASE WHEN c.duration BETWEEN INTERVAL '1 minute' AND INTERVAL '5 minutes' THEN 1 END) as medium_calls,
    COUNT(CASE WHEN c.duration > INTERVAL '5 minutes' THEN 1 END) as long_calls
FROM users u
INNER JOIN calls c ON u.U_ID = c.user_id
GROUP BY u.fio
LIMIT 10;
```

	fio text	totalIncome numeric	totalExpense numeric	net_balance numeric
1	Николаев Дмитрий Олегов...	100.00	81.00	181.00
2	АО "БизнесКонсалт"	3000.00	925.00	3925.00
3	Иванов Иван Иванович	500.00	77.50	577.50
4	ООО "СтальПроект"	8000.00	1492.50	9492.50
5	ИП Семенов А.В.	1500.00	1178.00	2678.00
6	Петрова Анна Сергеевна	300.00	344.00	644.00
7	ООО "ТелекомСервис"	5000.00	892.50	5892.50
8	Волкова Ольга Игоревна	400.00	102.50	502.50
9	Козлова Елена Викторовна	200.00	115.00	315.00
10	Сидоров Михаил Петрович	750.00	513.00	1263.00

	fio text	total_calls bigint	short_calls bigint	medium_calls bigint	long_calls bigint
1	Николаев Дмитрий Олегов...	2	0	1	1
2	АО "БизнесКонсалт"	2	0	0	2
3	Иванов Иван Иванович	2	0	0	2
4	ООО "СтальПроект"	2	0	0	2
5	ИП Семенов А.В.	2	0	0	2
6	Петрова Анна Сергеевна	2	0	0	2
7	ООО "ТелекомСервис"	2	0	0	2
8	Волкова Ольга Игоревна	2	0	0	2
9	Козлова Елена Викторовна	2	0	0	2
10	Сидоров Михаил Петрович	2	0	0	2

1.6.2.

```
-- 6.2 Привести примеры получения сводных (итоговых) таблиц с использованием PIVOT и UNPIVOT.

CREATE EXTENSION IF NOT EXISTS tablefunc; -- т.к. это Postgre

SELECT *
FROM crosstab(
    'SELECT
        EXTRACT(MONTH FROM transaction_date) as month,
        transaction_type,
        SUM(amount) as total_amount
    FROM transactions
    GROUP BY EXTRACT(MONTH FROM transaction_date), transaction_type
    ORDER BY 1, 2',
    'VALUES (''income''), (''expense'')'
) AS ct(month numeric, income numeric, expense numeric);

-- Если честно, то не уверен что точно так же можно сделать UNPIVOT в Postgre
SELECT city_id, 'night_cost' as cost_type, n_cost as cost_value FROM city
UNION ALL
SELECT city_id, 'day_cost' as cost_type, d_cost as cost_value FROM city
UNION ALL
SELECT city_id, 'avg_cost' as cost_type, (n_cost + d_cost)/2 as cost_value FROM city
ORDER BY city_id, cost_type
LIMIT 10;
```

	city_id bigint	cost_type text	cost_value numeric
1	1	avg_cost	2.2500000000000000
2	1	day_cost	2.00
3	1	night_cost	2.50
4	2	avg_cost	4.5000000000000000
5	2	day_cost	4.00
6	2	night_cost	5.00
7	3	avg_cost	7.2500000000000000
8	3	day_cost	6.50
9	3	night_cost	8.00
10	4	avg_cost	11.0000000000000000

	month numeric	income numeric	expense numeric
1	1	19750.00	5721.00