

Домашнее задание 3 (vo_HW)

Тема: Группировка данных и оконные функции

Ильиных Александр Александрович

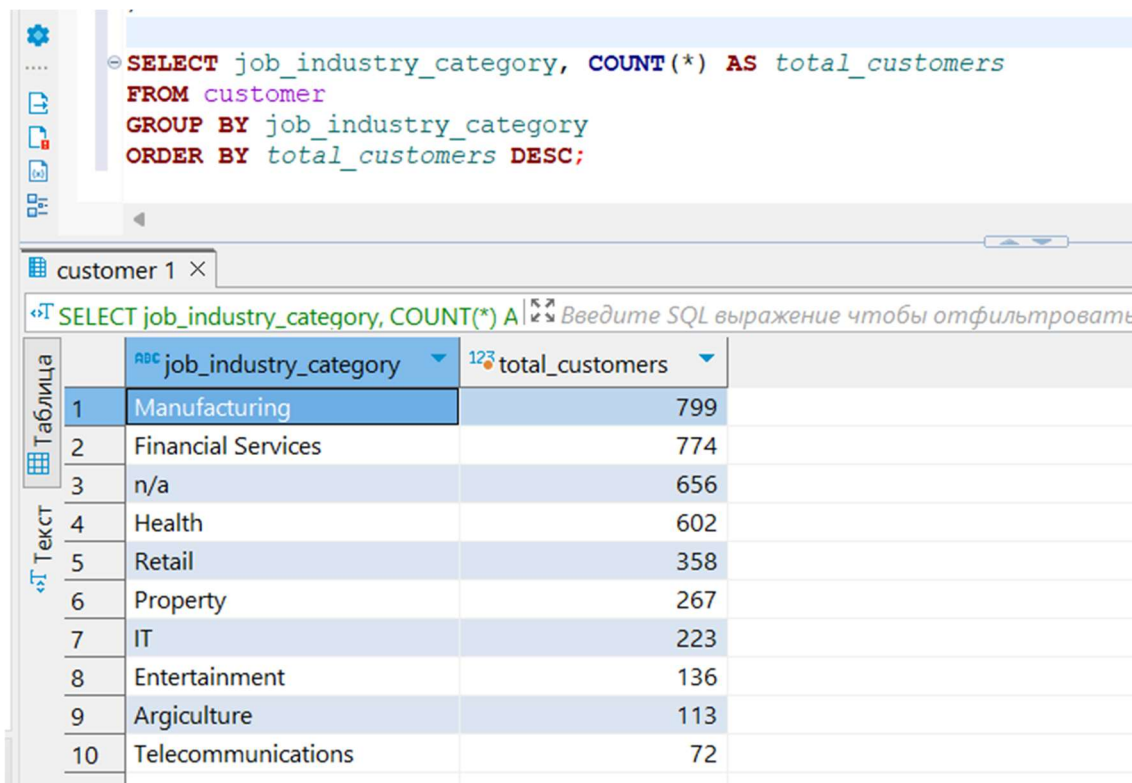
1. Создаем таблицы с заданными структурами и загружаем данные из csv-файлов.

```
CREATE TABLE customer (  
    customer_id int4  
    ,first_name varchar(50)  
    ,last_name varchar(50)  
    ,gender varchar(30)  
    ,DOB varchar(50)  
    ,job_title varchar(50)  
    ,job_industry_category varchar(50)  
    ,wealth_segment varchar(50)  
    ,deceased_indicator varchar(50)  
    ,owns_car varchar(30)  
    ,address varchar(50)  
    ,postcode varchar(30)  
    ,state varchar(30)  
    ,country varchar(30)  
    ,property_valuation int4  
    ,UNIQUE("customer_id")  
)  
  
CREATE TABLE transaction (  
    transaction_id int4  
    ,product_id int4  
    ,customer_id int4  
    ,transaction_date varchar(30)  
    ,online_order varchar(30)  
    ,order_status varchar(30)  
    ,brand varchar(30)  
    ,product_line varchar(30)  
    ,product_class varchar(30)  
    ,product_size varchar(30)  
    ,list_price float4  
    ,standard_cost float4  
    ,UNIQUE("transaction_id")  
    ,FOREIGN KEY ("customer_id") REFERENCES customer("customer_id")  
)
```

2. Выполнить следующие запросы:

2.1 Вывести распределение (количество) клиентов по сферам деятельности, отсортировав результат по убыванию количества. — (1 балл)

```
SELECT job_industry_category, COUNT(*) AS total_customers
FROM customer
GROUP BY job_industry_category
ORDER BY total_customers DESC;
```



The screenshot shows a database query editor with the following SQL query:

```
SELECT job_industry_category, COUNT(*) AS total_customers
FROM customer
GROUP BY job_industry_category
ORDER BY total_customers DESC;
```

Below the query editor, a table view displays the results of the query. The table has two columns: 'job_industry_category' and 'total_customers'. The results are sorted in descending order of 'total_customers'.

	job_industry_category	total_customers
1	Manufacturing	799
2	Financial Services	774
3	n/a	656
4	Health	602
5	Retail	358
6	Property	267
7	IT	223
8	Entertainment	136
9	Argiculture	113
10	Telecommunications	72

2.2 Найти сумму транзакций за каждый месяц по сферам деятельности, отсортировав по месяцам и по сфере деятельности. — (1 балл)

```
SELECT EXTRACT(MONTH FROM CAST(t.transaction_date AS TIMESTAMP)) AS month,
       c.job_industry_category,
       SUM(t.list_price) AS total_transaction_sum
FROM transaction t
INNER JOIN customer c ON t.customer_id = c.customer_id
GROUP BY EXTRACT(MONTH FROM CAST(t.transaction_date AS TIMESTAMP)),
         c.job_industry_category
ORDER BY month, c.job_industry_category;
```

customer 1 x

SELECT EXTRACT(MONTH FROM CAST(t.transaction_date AS TIMESTAMP)) AS month, c.job_industry_category, SUM(t.list_price) AS total_transaction_sum FROM transaction t INNER JOIN customer c ON t.customer_id = c.customer_id GROUP BY EXTRACT(MONTH FROM CAST(t.transaction_date AS TIMESTAMP)), c.job_industry_category ORDER BY month, c.job_industry_category;

customer 1 x

SELECT EXTRACT(MONTH FROM CAST(t.transaction_date AS TIMESTAMP)) AS month, c.job_industry_category, SUM(t.list_price) AS total_transaction_sum FROM transaction t INNER JOIN customer c ON t.customer_id = c.customer_id GROUP BY EXTRACT(MONTH FROM CAST(t.transaction_date AS TIMESTAMP)), c.job_industry_category ORDER BY month, c.job_industry_category;

	month	job_industry_category	total_transaction_sum
1	1	Argiculture	43 513,812
2	1	Entertainment	64 089,934
3	1	Financial Services	366 383,78
4	1	Health	286 860,44
5	1	IT	107 783,414
6	1	Manufacturing	365 232,38
7	1	n/a	316 819,72
8	1	Property	100 686,97
9	1	Retail	182 375,72
10	1	Telecommunications	31 210,2
11	2	Argiculture	60 016,81
12	2	Entertainment	63 965,99
13	2	Financial Services	375 961,8
14	2	Health	268 525,75

Обновить

Save

Cancel

Экспорт данных ...

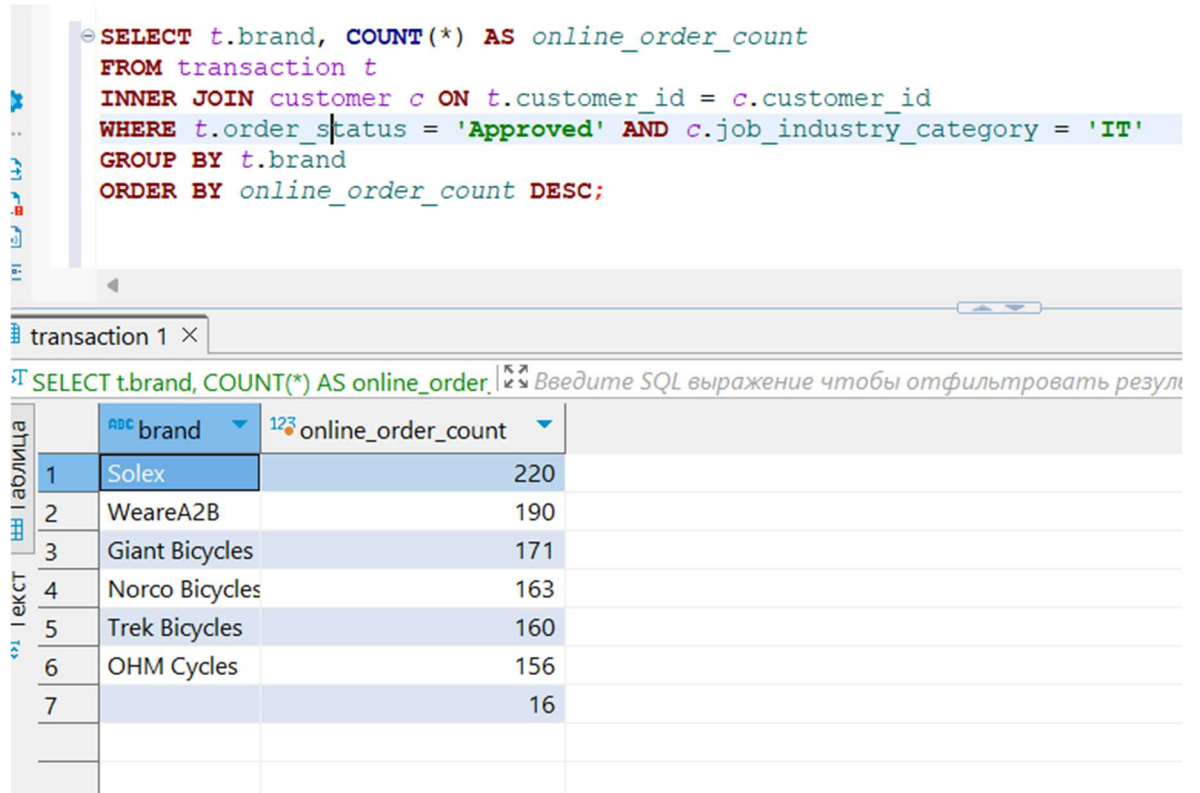
200

120

120 строк получено - 0,027s (0,001s получ.), 2024-02-20 в 22:48:54

2.3 Вывести количество онлайн-заказов для всех брендов в рамках подтвержденных заказов клиентов из сферы IT. — (1 балл)

```
SELECT t.brand, COUNT(*) AS online_order_count
FROM transaction t
INNER JOIN customer c ON t.customer_id = c.customer_id
WHERE t.order_status = 'Approved' AND c.job_industry_category = 'IT'
GROUP BY t.brand
ORDER BY online_order_count DESC;
```



The screenshot shows a SQL IDE interface. The top pane displays the SQL query. The bottom pane shows the results of the query in a table. The table has two columns: 'brand' and 'online_order_count'. The results are sorted in descending order of 'online_order_count'.

	brand	online_order_count
1	Solex	220
2	WeareA2B	190
3	Giant Bicycles	171
4	Norco Bicycles	163
5	Trek Bicycles	160
6	OHM Cycles	156
7		16

2.4 Найти по всем клиентам сумму всех транзакций (list_price), максимум, минимум и количество транзакций, отсортировав результат по убыванию суммы транзакций и количества клиентов. Выполните двумя способами: используя только group by и используя только оконные функции. Сравните результат. — (2 балла)

используя только оконные функции group by:

```
SELECT
  c.customer_id,
  SUM(t.list_price) AS total_transaction,
  MAX(t.list_price) AS max_transaction,
  MIN(t.list_price) AS min_transaction,
  COUNT(t.transaction_id) AS transaction_count
FROM customer c
INNER JOIN transaction t ON c.customer_id = t.customer_id
GROUP BY c.customer_id
ORDER BY total_transaction DESC, transaction_count DESC;
```

SELECT

```
c.customer_id,
SUM(t.list_price) AS total_transaction,
MAX(t.list_price) AS max_transaction,
MIN(t.list_price) AS min_transaction,
COUNT(t.transaction_id) AS transaction_count
FROM customer c
INNER JOIN transaction t ON c.customer_id = t.customer_id
GROUP BY c.customer_id
ORDER BY total_transaction DESC, transaction_count DESC;
```

	customer_id	total_transaction	max_transaction	min_transaction	transaction_count
1	2 183	19 071,322	2 005,66	230,91	14
2	1 129	18 349,27	1 992,93	290,62	13
3	1 597	18 052,68	2 091,47	360,4	12
4	941	17 898,459	2 091,47	1 057,51	10
5	2 788	17 258,94	2 083,94	183,86	11
6	936	17 160,24	2 005,66	183,86	12
7	1 887	17 133,932	2 091,47	688,63	11
8	1 302	17 035,83	1 977,36	71,16	13
9	1 140	16 199,24	2 083,94	183,86	13
10	2 309	16 122,341	2 091,47	290,62	12
11	729	15 825,999	2 091,47	586,45	10
12	1 103	15 447,92	1 977,36	230,91	12
13	1 317	15 370,81	2 091,47	569,56	9
14	2 874	15 001,01	2 005,66	544,05	11

Обновить Save Cancel Экспорт данных ... 200 3 493

3493 строк получено - 0,035s (0,010s получ.), 2024-02-20 в 23:05:25

таблица Инт. вставка 71 : 1 : 2003 Sel: 0 | 0

```

SELECT
  c.customer_id,
  SUM(t.list_price) OVER (PARTITION BY c.customer_id) AS total_transaction,
  MAX(t.list_price) OVER (PARTITION BY c.customer_id) AS max_transaction,
  MIN(t.list_price) OVER (PARTITION BY c.customer_id) AS min_transaction,
  COUNT(*) OVER (PARTITION BY c.customer_id) AS transaction_count
FROM
  customer c
  JOIN transaction t ON c.customer_id = t.customer_id
ORDER BY total_transaction DESC, transaction_count DESC;

```

customer 1 X

SELECT c.customer_id, SUM(t.list_price) OVER (PARTITION BY c.customer_id) AS total_transaction, MAX(t.list_price) OVER (PARTITION BY c.customer_id) AS max_transaction, MIN(t.list_price) OVER (PARTITION BY c.customer_id) AS min_transaction, COUNT(*) OVER (PARTITION BY c.customer_id) AS transaction_count

	customer_id	total_transaction	max_transaction	min_transaction	transaction_count
1	2 183	19 071,318	2 005,66	230,91	14
2	2 183	19 071,318	2 005,66	230,91	14
3	2 183	19 071,318	2 005,66	230,91	14
4	2 183	19 071,318	2 005,66	230,91	14
5	2 183	19 071,318	2 005,66	230,91	14
6	2 183	19 071,318	2 005,66	230,91	14
7	2 183	19 071,318	2 005,66	230,91	14
8	2 183	19 071,318	2 005,66	230,91	14
9	2 183	19 071,318	2 005,66	230,91	14
10	2 183	19 071,318	2 005,66	230,91	14
11	2 183	19 071,318	2 005,66	230,91	14
12	2 183	19 071,318	2 005,66	230,91	14
13	2 183	19 071,318	2 005,66	230,91	14
14	2 183	19 071,318	2 005,66	230,91	14

Обновить Save Cancel Экспорт данных ... 200 19 997

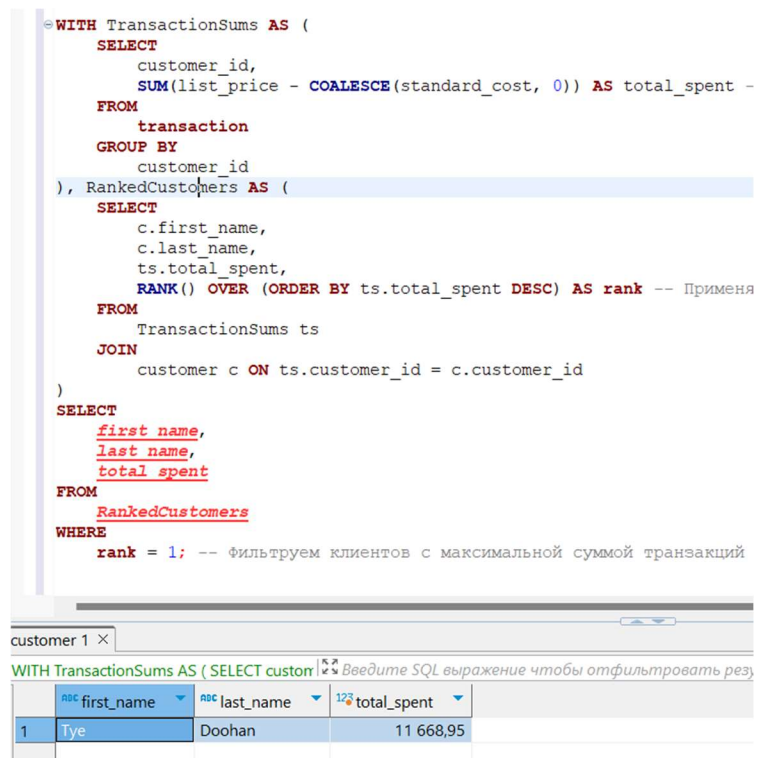
... 19997 строк получено - 0,056s (0,009s получ.), 2024-02-20 в 23:15:57

Используя оператор GROUP BY, результат будет содержать только уникальные значения customer_id. В таблице transaction есть несколько строк для одного customer_id, они сгруппированы вместе и будет одна строка результата для каждого customer_id. Получилось **3493 записей**.

С другой стороны, при использовании оконных функций, которые не группируют данные, все строки будут участвовать в вычислениях оконных функций, что привело к большому количеству строк в итоговом результате. Получилось **19997 записей**.

2.5 Найти имена и фамилии клиентов с минимальной/максимальной суммой транзакций за весь период (сумма транзакций не может быть null). Напишите отдельные запросы для минимальной и максимальной суммы. — (2 балла)

```
WITH TransactionSums AS (
    SELECT
        customer_id,
        SUM(list_price - COALESCE(standard_cost, 0)) AS total_spent --
        Используем COALESCE для исключения NULL значений в стоимости
    FROM
        transaction
    GROUP BY
        customer_id
), RankedCustomers AS (
    SELECT
        c.first_name,
        c.last_name,
        ts.total_spent,
        RANK() OVER (ORDER BY ts.total_spent DESC) AS rank -- Применяем
        оконную функцию для ранжирования по сумме транзакций
    FROM
        TransactionSums ts
    JOIN
        customer c ON ts.customer_id = c.customer_id
)
SELECT
    first_name,
    last_name,
    total_spent
FROM
    RankedCustomers
WHERE
    rank = 1; -- Фильтруем клиентов с максимальной суммой транзакций
```



The screenshot shows a SQL IDE with the following query in the editor:

```
WITH TransactionSums AS (
    SELECT
        customer_id,
        SUM(list_price - COALESCE(standard_cost, 0)) AS total_spent -
    FROM
        transaction
    GROUP BY
        customer_id
), RankedCustomers AS (
    SELECT
        c.first_name,
        c.last_name,
        ts.total_spent,
        RANK() OVER (ORDER BY ts.total_spent DESC) AS rank -- Применя
    FROM
        TransactionSums ts
    JOIN
        customer c ON ts.customer_id = c.customer_id
)
SELECT
    first_name,
    last_name,
    total_spent
FROM
    RankedCustomers
WHERE
    rank = 1; -- Фильтруем клиентов с максимальной суммой транзакций
```

Below the editor, the results are displayed in a table. The first row shows the customer with the highest total spent.

	first_name	last_name	total_spent
1	Tye	Doohan	11 668,95

```

WITH TransactionSums AS (
    SELECT
        customer_id,
        SUM(list_price - COALESCE(standard_cost, 0)) AS total_spent --
Исключаем NULL значения в стоимости
    FROM
        transaction
    GROUP BY
        customer_id
), RankedCustomers AS (
    SELECT
        c.first_name,
        c.last_name,
        ts.total_spent,
        RANK() OVER (ORDER BY ts.total_spent ASC) AS rank -- Ранжируем по
возрастанию суммы транзакций
    FROM
        TransactionSums ts
    JOIN
        customer c ON ts.customer_id = c.customer_id
)
SELECT
    first_name,
    last_name,
    total_spent
FROM
    RankedCustomers
WHERE
    rank = 1; -- Фильтруем клиентов с минимальной суммой транзакций

```

```

WITH TransactionSums AS (
    SELECT
        customer_id,
        SUM(list_price - COALESCE(standard_cost, 0)) AS total_spent --
    FROM
        transaction
    GROUP BY
        customer_id
), RankedCustomers AS (
    SELECT
        c.first_name,
        c.last_name,
        ts.total_spent,
        RANK() OVER (ORDER BY ts.total_spent ASC) AS rank -- Ранжируем
    FROM
        TransactionSums ts
    JOIN
        customer c ON ts.customer_id = c.customer_id
)
SELECT
    first_name,
    last_name,
    total_spent
FROM
    RankedCustomers
WHERE
    rank = 1; -- Фильтруем клиентов с минимальной суммой транзакций

```

customer 1 ×

WITH TransactionSums AS (SELECT custom

Введите SQL выражение чтобы отфильтровать резул

Таблица	first_name	last_name	total_spent
1	Hamlen	Slograve	15,080002

2.6 Вывести только самые первые транзакции клиентов. Решить с помощью оконных функций. — (1 балл)

```
WITH RankedTransactions AS (  
    SELECT  
        t.*,  
        ROW_NUMBER() OVER (PARTITION BY t.customer_id ORDER BY  
t.transaction_date ASC) AS rn  
    -- Применяем оконную функцию, разделяя данные по customer_id и сортируя  
    их по дате транзакции в порядке возрастания  
    FROM  
        transaction t  
)  
SELECT  
    *  
FROM  
    RankedTransactions  
WHERE  
    rn = 1; -- Фильтруем, чтобы получить только первые транзакции  
каждого клиента
```

The screenshot shows a SQL IDE interface. The top pane displays the SQL query used to filter transactions. The bottom pane shows the results of the query in a table format.

SQL Query:

```
WITH RankedTransactions AS (  
    SELECT  
        t.*,  
        ROW_NUMBER() OVER (PARTITION BY t.customer_id ORDER BY t.transaction_date ASC) AS rn  
    -- Применяем оконную функцию, разделяя данные по customer_id и сортируя их по дате транзакции в  
    FROM  
        transaction t  
)  
SELECT  
    *  
FROM  
    RankedTransactions  
WHERE  
    rn = 1; -- Фильтруем, чтобы получить только первые транзакции каждого клиента
```

Table Results:

	transaction_id	product_id	customer_id	transaction_date	online_order	
3483	3 525	50	3 490	2017-01-01 00:00:00.000	True	A
3484	1 282	97	3 491	2017-02-02 00:00:00.000	False	A
3485	18 129	80	3 492	2017-06-10 00:00:00.000	True	A
3486	619	12	3 493	2017-03-16 00:00:00.000	False	A
3487	18 470	38	3 494	2017-04-08 00:00:00.000	False	A
3488	18 035	0	3 495	2017-01-13 00:00:00.000	True	A
3489	9 769	5	3 496	2017-03-07 00:00:00.000	False	A
3490	8 276	18	3 497	2017-09-01 00:00:00.000	True	A
3491	13 469	12	3 498	2017-02-03 00:00:00.000	True	A
3492	2 794	62	3 499	2017-01-12 00:00:00.000	False	A
3493	6 309	69	3 500	2017-01-09 00:00:00.000	True	A

The bottom status bar shows the number of rows selected: 200 / 3 493.

2.7 Вывести имена, фамилии и профессии клиентов, между соседними транзакциями которых был максимальный интервал (интервал вычисляется в днях) — (2 балла).

```
WITH TransactionIntervals AS (  
    SELECT  
        customer_id,  
        transaction_date,  
        LEAD(transaction_date) OVER (PARTITION BY customer_id ORDER BY  
transaction_date) - transaction_date AS interval_days  
    FROM  
        transaction  
) , MaxIntervals AS (  
    SELECT  
        customer_id,  
        MAX(interval_days) AS max_interval  
    FROM  
        TransactionIntervals  
    GROUP BY  
        customer_id  
) , MaxInterval AS (  
    SELECT  
        MAX(max_interval) AS max_interval  
    FROM  
        MaxIntervals  
)  
SELECT  
    c.first_name,  
    c.last_name,  
    c.job_title  
FROM  
    customer c  
JOIN  
    MaxIntervals mi ON c.customer_id = mi.customer_id  
JOIN  
    MaxInterval mi2 ON mi.max_interval = mi2.max_interval;
```

```

WITH TransactionIntervals AS (
    SELECT
        customer_id,
        transaction_date,
        LEAD(transaction_date) OVER (PARTITION BY customer_id
            ORDER BY transaction_date) - transaction_date AS interval_days
    FROM
        transaction
), MaxIntervals AS (
    SELECT
        customer_id,
        MAX(interval_days) AS max_interval
    FROM
        TransactionIntervals
    GROUP BY
        customer_id
), MaxInterval AS (
    SELECT
        MAX(max_interval) AS max_interval
    FROM
        MaxIntervals
)
SELECT
    c.first_name,
    c.last_name,
    c.job_title
FROM
    customer c
JOIN
    MaxIntervals mi ON c.customer_id = mi.customer_id
JOIN
    MaxInterval mi2 ON mi.max_interval = mi2.max_interval;

```

customer 1 ×

WITH TransactionIntervals AS (SELECT cust

	first_name	last_name	job_title
1	Susanetta		Legal Assistant