JensonUSA
www.JensonUSA.com

Jenson USA

SQL ANALYSIS

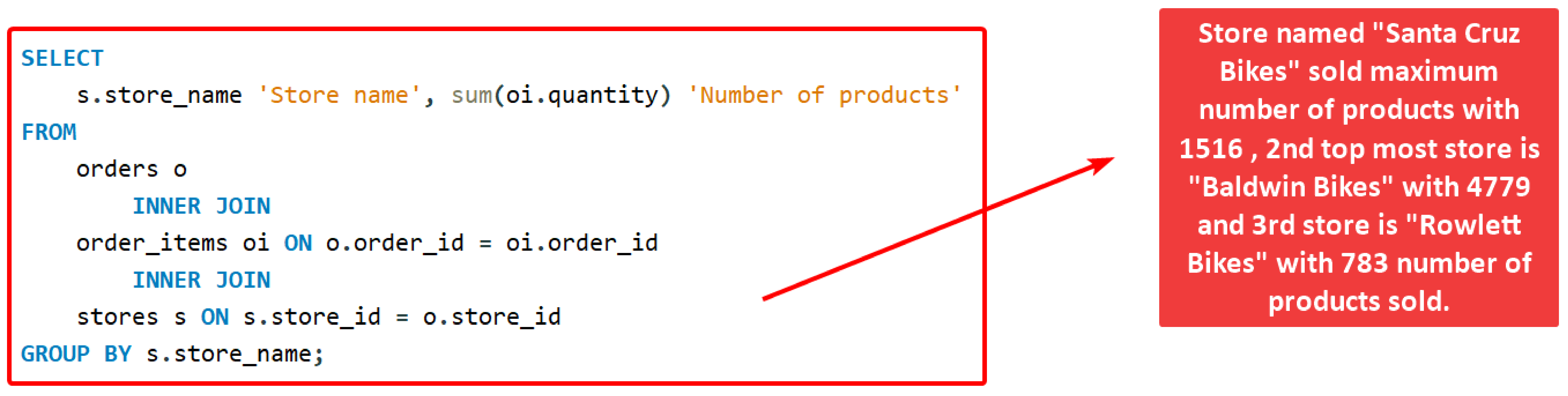
America's Bike & Adventure Shop

ABOUT COMPANY

Jenson USA is a leading American cycling retailer founded in 1994 and headquartered in Riverside, California. With over 30,000 products available online and at retail locations in Corona and Riverside, the company caters to enthusiasts of road, mountain, BMX, gravel, and commuter bikes. Jenson USA emphasizes a customer-centric approach, encapsulated in its core value: "Customers Are Everything.

1.Find the total number of products sold by each store along with the store name.

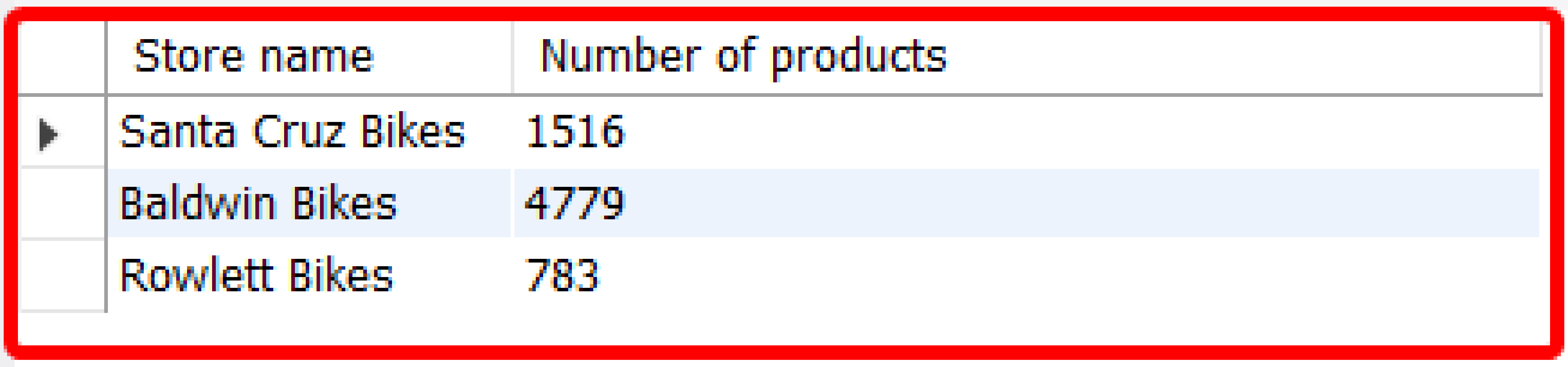
QUERY



```
SELECT
    s.store_name 'Store name', sum(oi.quantity) 'Number of products'
FROM
    orders o
        INNER JOIN
    order_items oi ON o.order_id = oi.order_id
        INNER JOIN
    stores s ON s.store_id = o.store_id
GROUP BY s.store_name;
```

Store named "Santa Cruz Bikes" sold maximum number of products with 1516 , 2nd top most store is "Baldwin Bikes" with 4779 and 3rd store is "Rowlett Bikes" with 783 number of products sold.

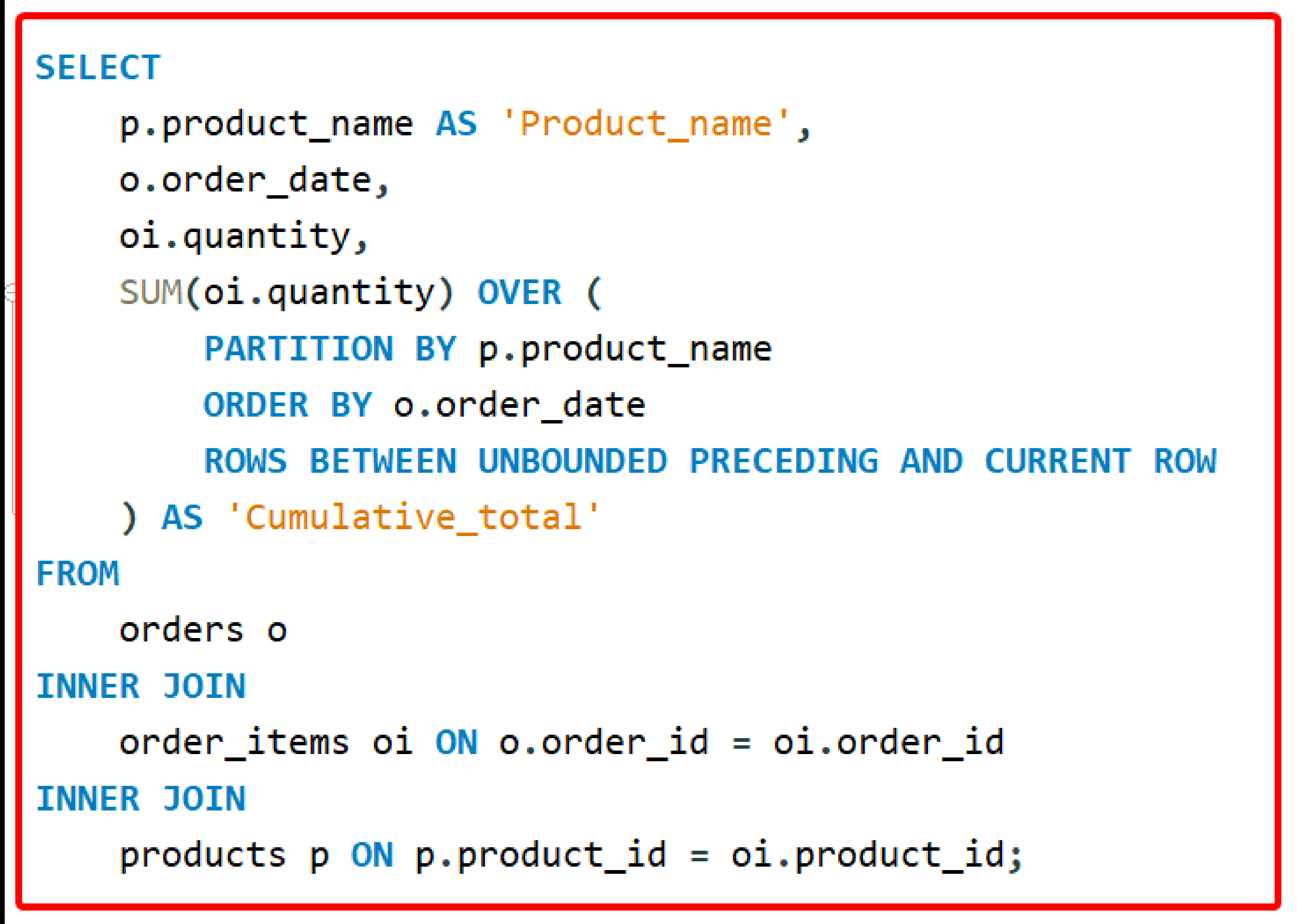
OUTPUT



Store name	Number of products
Santa Cruz Bikes	1516
Baldwin Bikes	4779
Rowlett Bikes	783

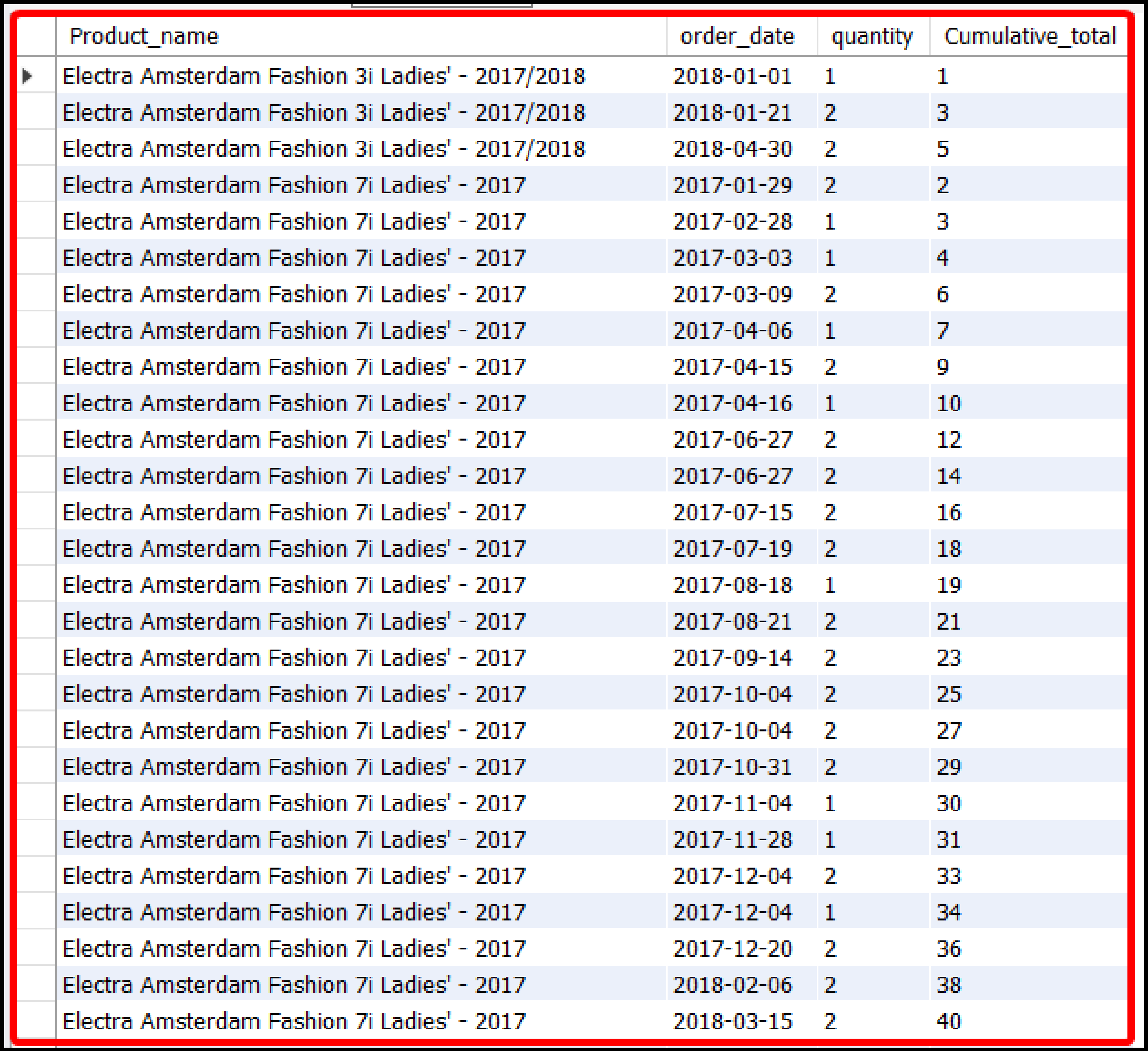
2.Calculate the cumulative sum of quantities sold for each product over time.

QUERY



```
SELECT
    p.product_name AS 'Product_name',
    o.order_date,
    oi.quantity,
    SUM(oi.quantity) OVER (
        PARTITION BY p.product_name
        ORDER BY o.order_date
        ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
    ) AS 'Cumulative_total'
FROM
    orders o
INNER JOIN
    order_items oi ON o.order_id = oi.order_id
INNER JOIN
    products p ON p.product_id = oi.product_id;
```

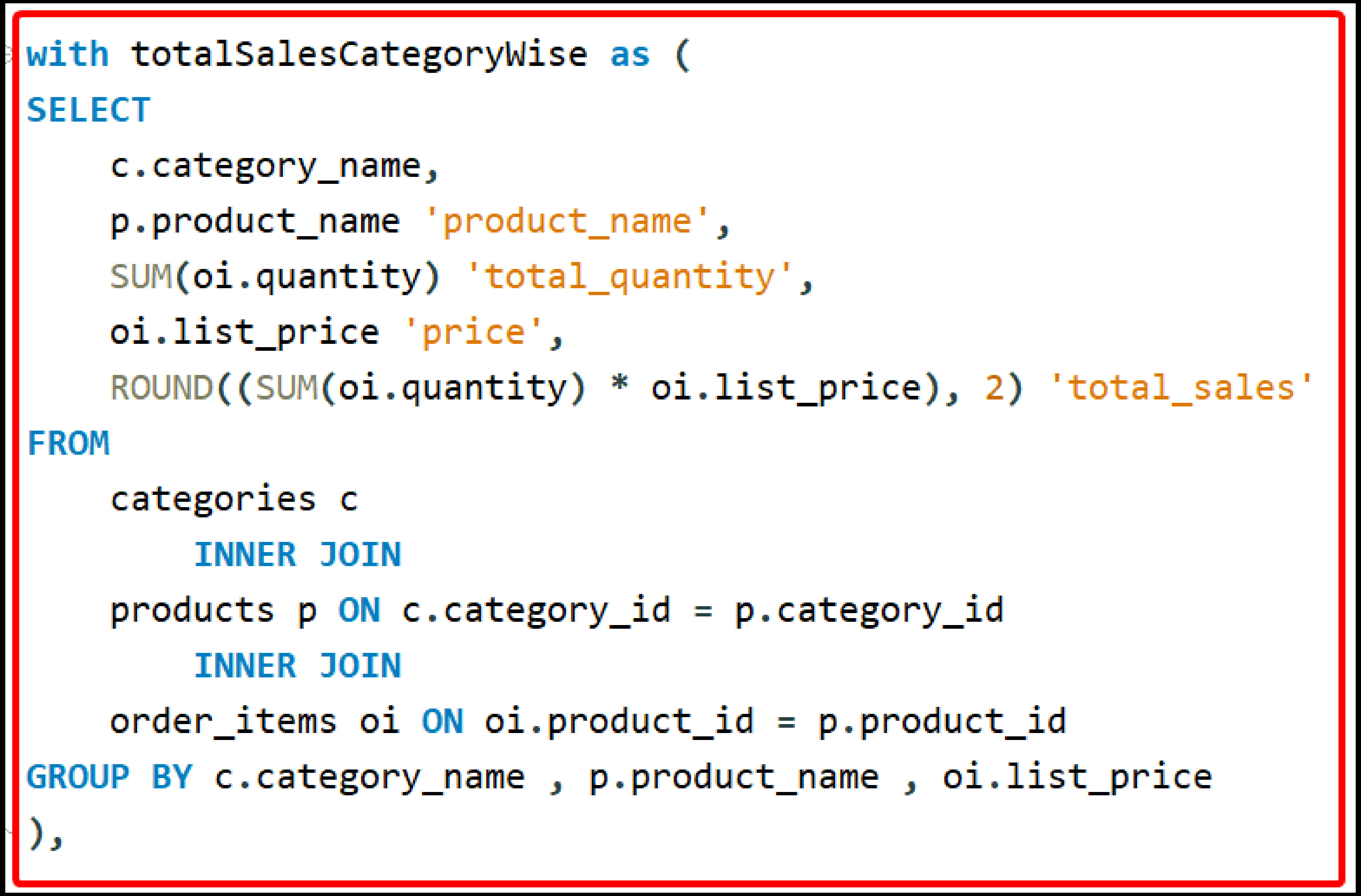
OUTPUT



Product_name	order_date	quantity	Cumulative_total
Electra Amsterdam Fashion 3i Ladies' - 2017/2018	2018-01-01	1	1
Electra Amsterdam Fashion 3i Ladies' - 2017/2018	2018-01-21	2	3
Electra Amsterdam Fashion 3i Ladies' - 2017/2018	2018-04-30	2	5
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-01-29	2	2
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-02-28	1	3
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-03-03	1	4
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-03-09	2	6
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-04-06	1	7
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-04-15	2	9
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-04-16	1	10
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-06-27	2	12
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-06-27	2	14
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-07-15	2	16
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-07-19	2	18
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-08-18	1	19
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-08-21	2	21
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-09-14	2	23
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-10-04	2	25
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-10-04	2	27
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-10-31	2	29
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-11-04	1	30
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-11-28	1	31
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-12-04	2	33
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-12-04	1	34
Electra Amsterdam Fashion 7i Ladies' - 2017	2017-12-20	2	36
Electra Amsterdam Fashion 7i Ladies' - 2017	2018-02-06	2	38
Electra Amsterdam Fashion 7i Ladies' - 2017	2018-03-15	2	40

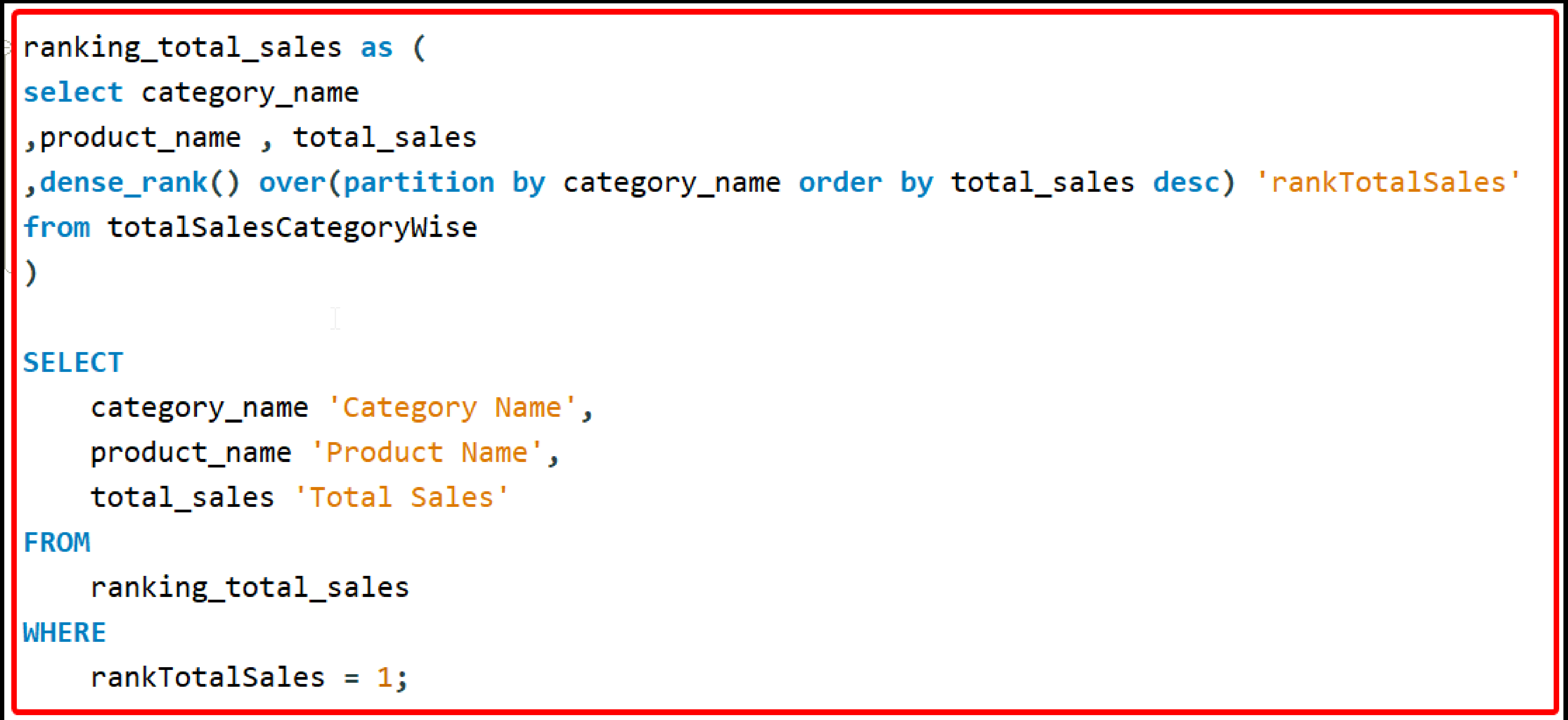
3.Find the product with the highest total sales (quantity * price) for each category.

QUERY



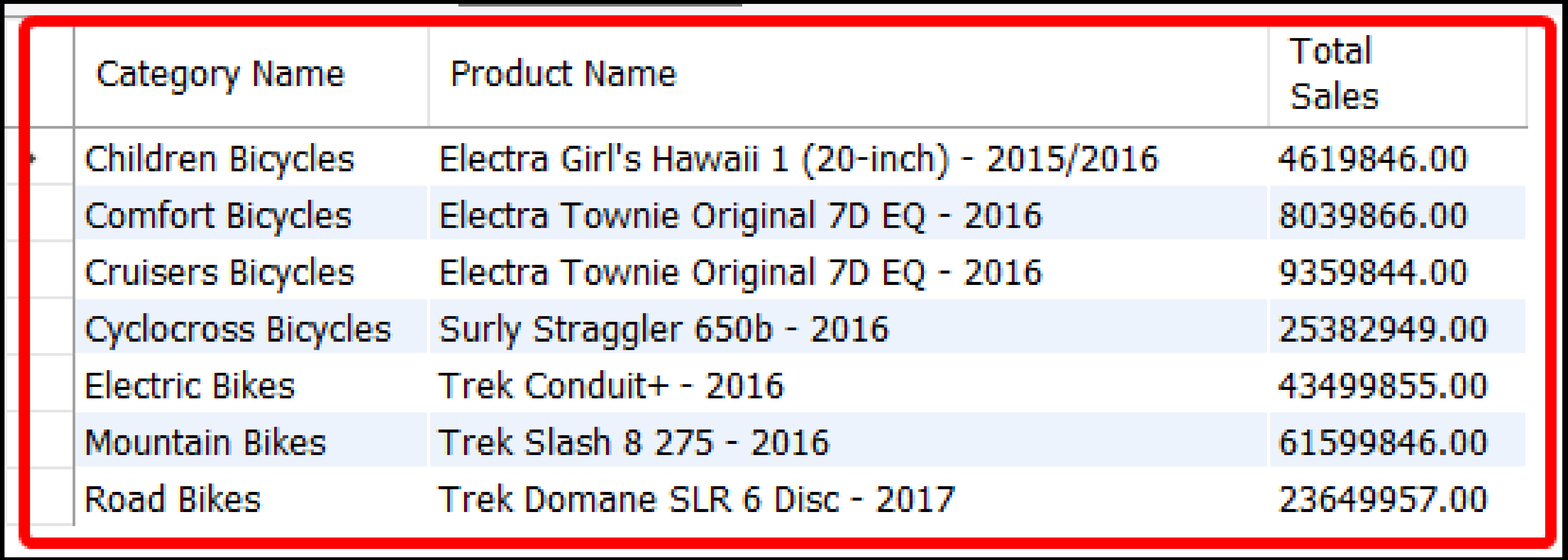
```
with totalSalesCategoryWise as (
SELECT
    c.category_name,
    p.product_name 'product_name',
    SUM(oi.quantity) 'total_quantity',
    oi.list_price 'price',
    ROUND((SUM(oi.quantity) * oi.list_price), 2) 'total_sales'
FROM
    categories c
        INNER JOIN
    products p ON c.category_id = p.category_id
        INNER JOIN
    order_items oi ON oi.product_id = p.product_id
GROUP BY c.category_name , p.product_name , oi.list_price
),
```

QUERY



```
ranking_total_sales as (
select category_name
,product_name , total_sales
,dense_rank() over(partition by category_name order by total_sales desc) 'rankTotalSales'
from totalSalesCategoryWise
)

SELECT
    category_name 'Category Name',
    product_name 'Product Name',
    total_sales 'Total Sales'
FROM
    ranking_total_sales
WHERE
    rankTotalSales = 1;
```

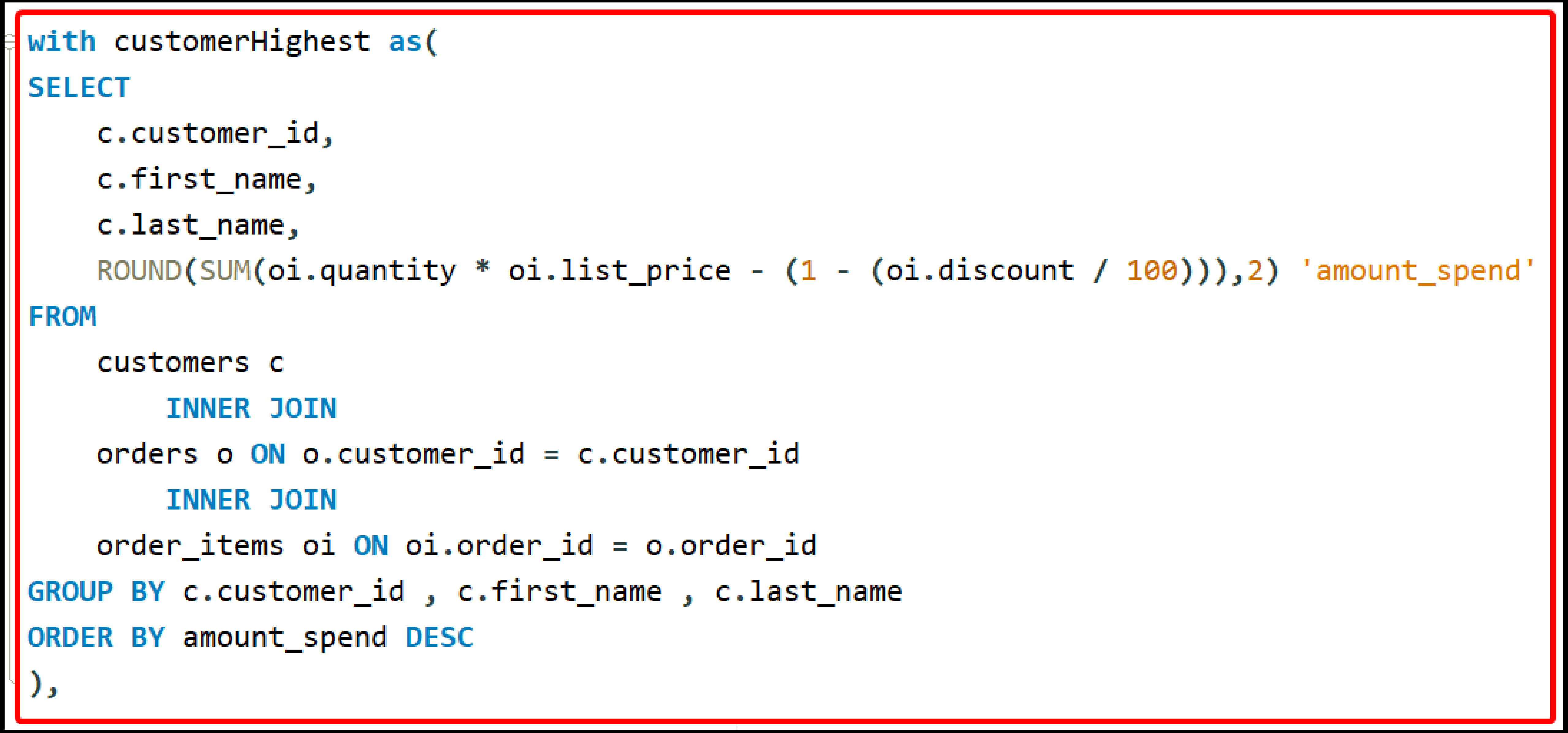
OUTPUT



Category Name	Product Name	Total Sales
Children Bicycles	Electra Girl's Hawaii 1 (20-inch) - 2015/2016	4619846.00
Comfort Bicycles	Electra Townie Original 7D EQ - 2016	8039866.00
Cruisers Bicycles	Electra Townie Original 7D EQ - 2016	9359844.00
Cyclocross Bicycles	Surly Straggler 650b - 2016	25382949.00
Electric Bikes	Trek Conduit+ - 2016	43499855.00
Mountain Bikes	Trek Slash 8 275 - 2016	61599846.00
Road Bikes	Trek Domane SLR 6 Disc - 2017	23649957.00

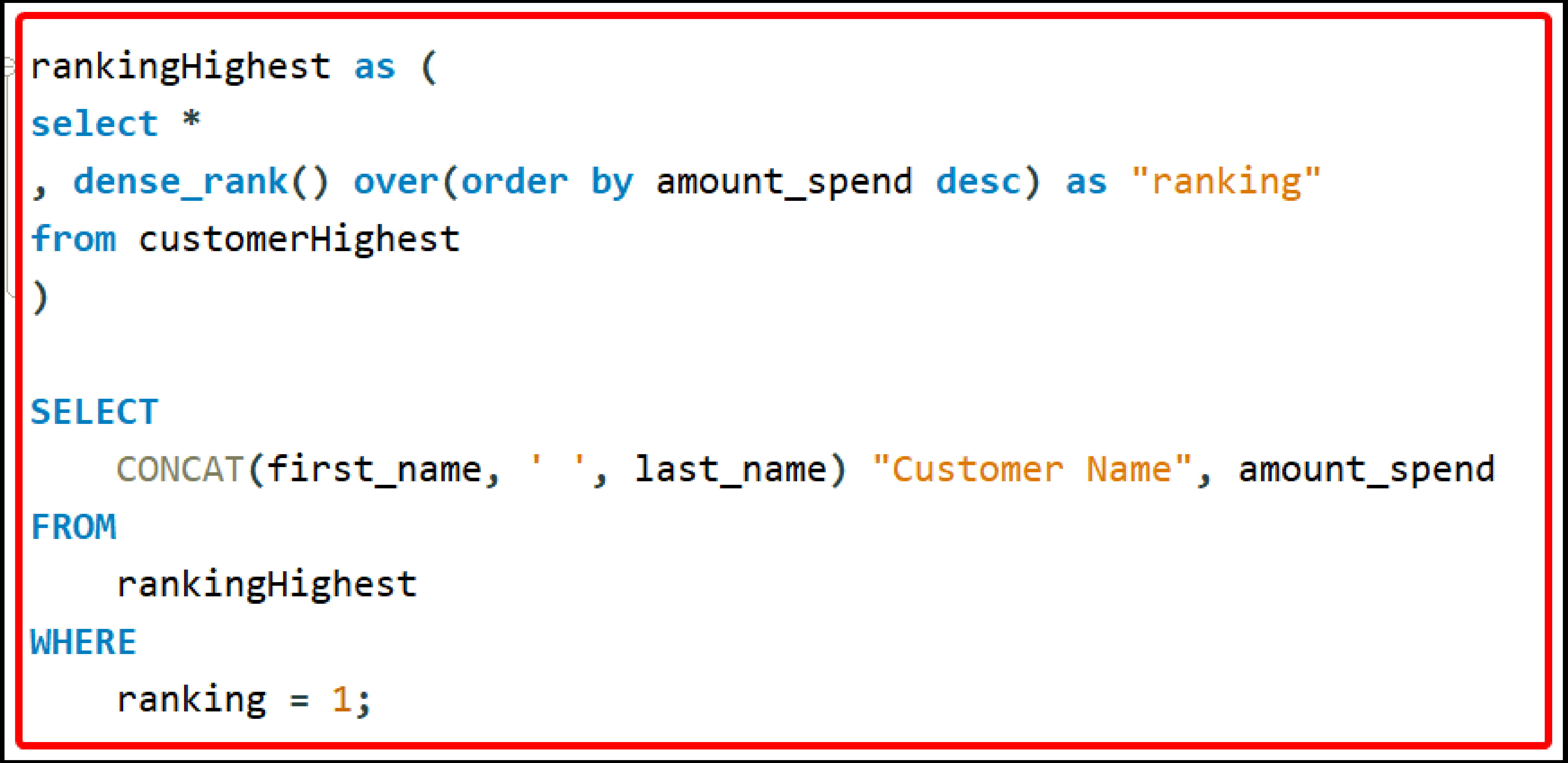
4.Find the customer who spent the most money on orders.

QUERY



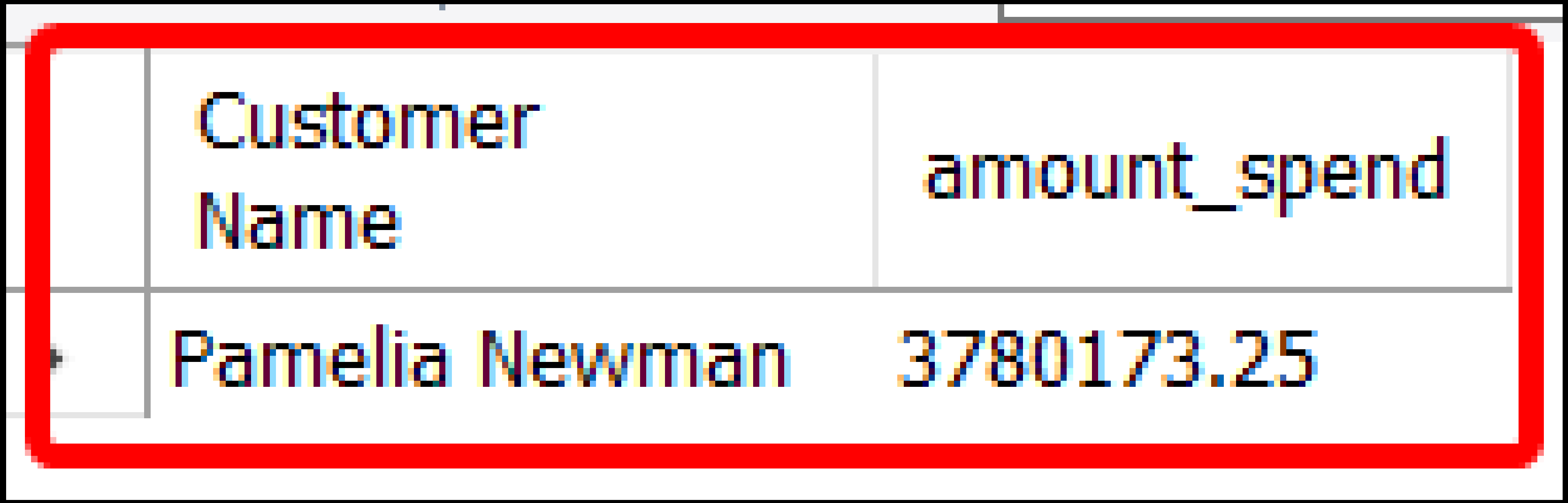
```
with customerHighest as(
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    ROUND(SUM(oi.quantity * oi.list_price - (1 - (oi.discount / 100))),2) 'amount_spend'
FROM
    customers c
        INNER JOIN
    orders o ON o.customer_id = c.customer_id
        INNER JOIN
    order_items oi ON oi.order_id = o.order_id
GROUP BY c.customer_id , c.first_name , c.last_name
ORDER BY amount_spend DESC
),
```

QUERY



```
rankingHighest as (
select *
, dense_rank() over(order by amount_spend desc) as "ranking"
from customerHighest
)

SELECT
    CONCAT(first_name, ' ', last_name) "Customer Name", amount_spend
FROM
    rankingHighest
WHERE
    ranking = 1;
```

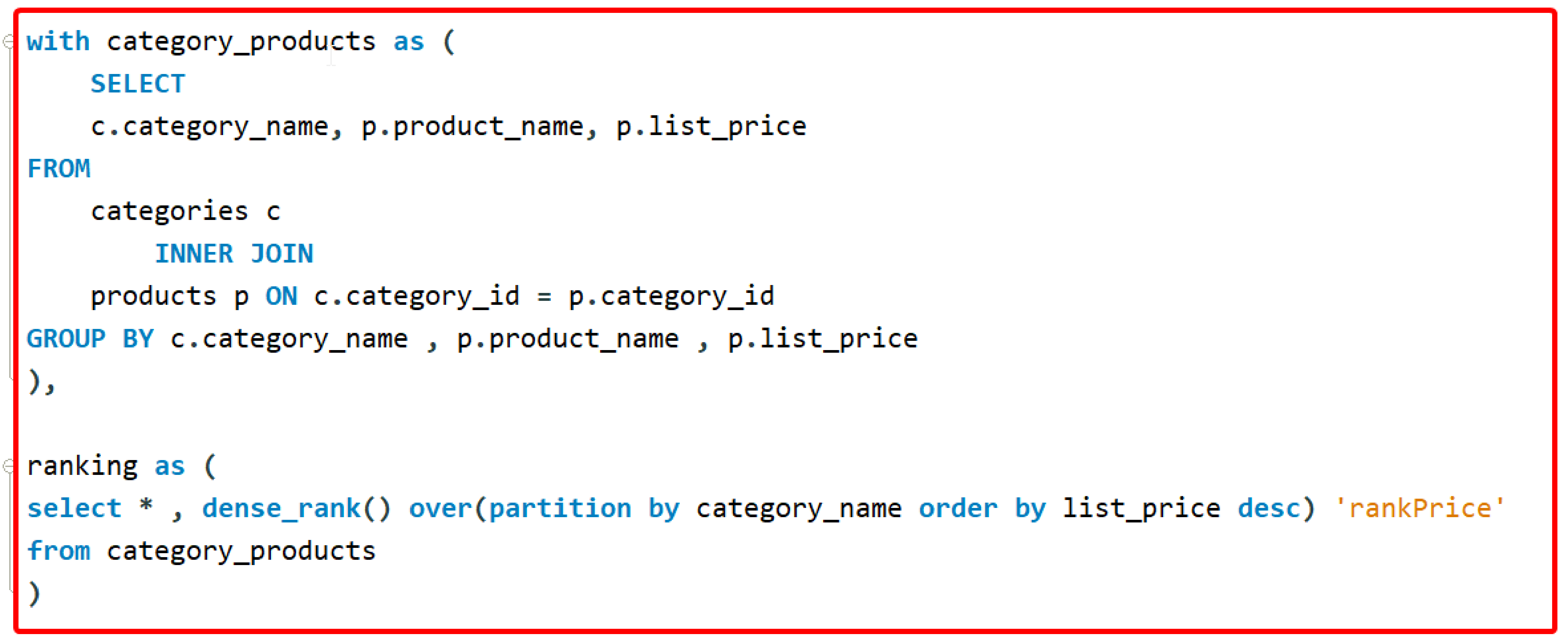
OUTPUT



Customer Name	amount_spend
Pamelia Newman	3780173.25

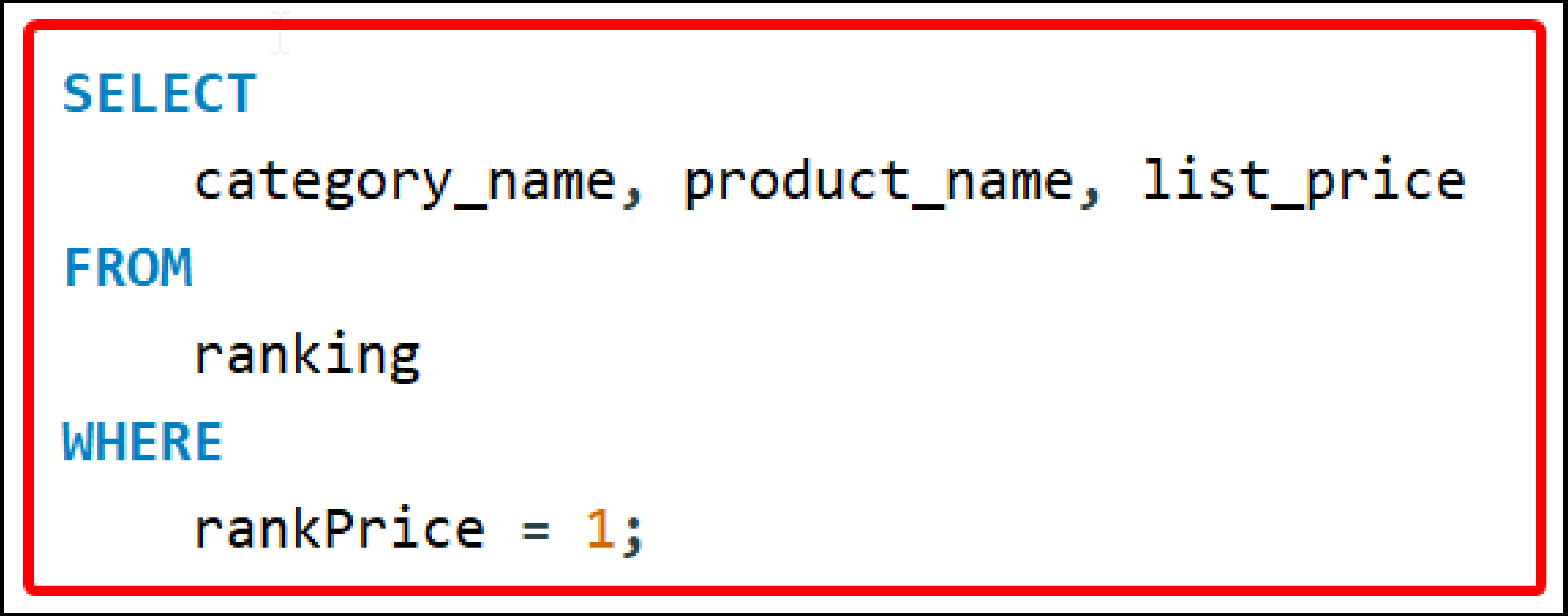
5.Find the highest-priced product for each category name.

QUERY



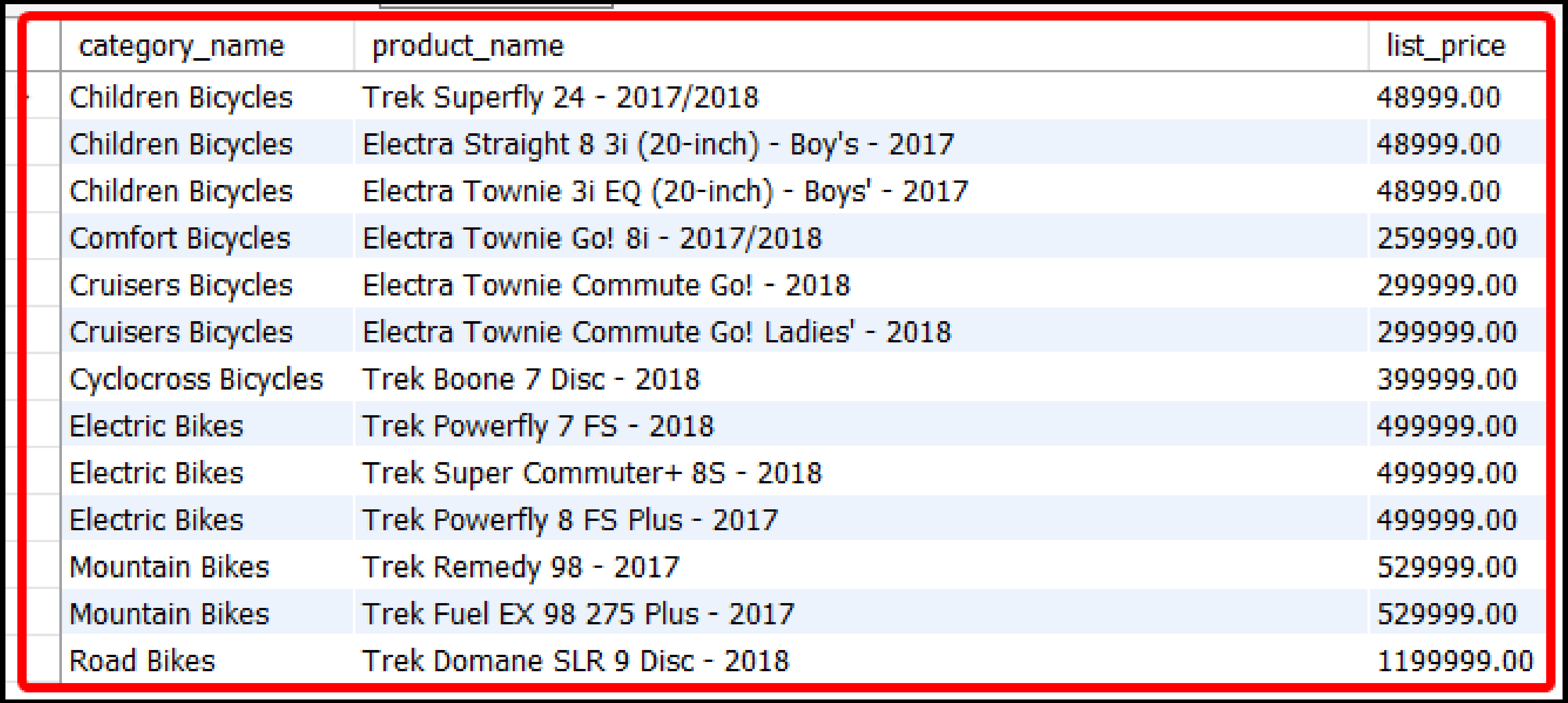
```
with category_products as (
    SELECT
    c.category_name, p.product_name, p.list_price
FROM
    categories c
        INNER JOIN
    products p ON c.category_id = p.category_id
GROUP BY c.category_name , p.product_name , p.list_price
),

ranking as (
select * , dense_rank() over(partition by category_name order by list_price desc) 'rankPrice'
from category_products
)
```

QUERY



```
SELECT
    category_name, product_name, list_price
FROM
    ranking
WHERE
    rankPrice = 1;
```

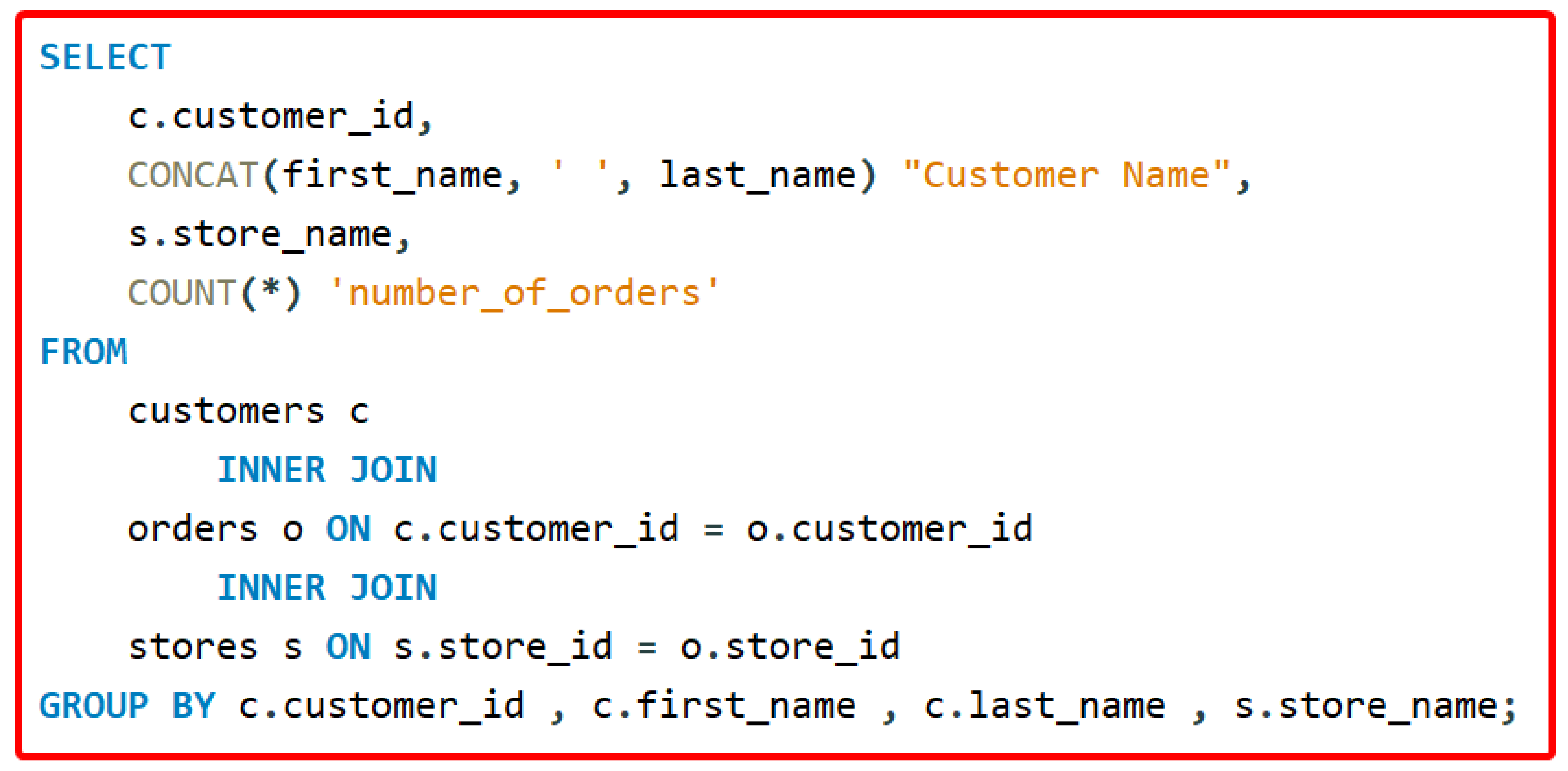
OUTPUT



category_name	product_name	list_price
Children Bicycles	Trek Superfly 24 - 2017/2018	48999.00
Children Bicycles	Electra Straight 8 3i (20-inch) - Boy's - 2017	48999.00
Children Bicycles	Electra Townie 3i EQ (20-inch) - Boys' - 2017	48999.00
Comfort Bicycles	Electra Townie Go! 8i - 2017/2018	259999.00
Cruisers Bicycles	Electra Townie Commute Go! - 2018	299999.00
Cruisers Bicycles	Electra Townie Commute Go! Ladies' - 2018	299999.00
Cyclocross Bicycles	Trek Boone 7 Disc - 2018	399999.00
Electric Bikes	Trek Powerfly 7 FS - 2018	499999.00
Electric Bikes	Trek Super Commuter+ 8S - 2018	499999.00
Electric Bikes	Trek Powerfly 8 FS Plus - 2017	499999.00
Mountain Bikes	Trek Remedy 98 - 2017	529999.00
Mountain Bikes	Trek Fuel EX 98 275 Plus - 2017	529999.00
Road Bikes	Trek Domane SLR 9 Disc - 2018	1199999.00

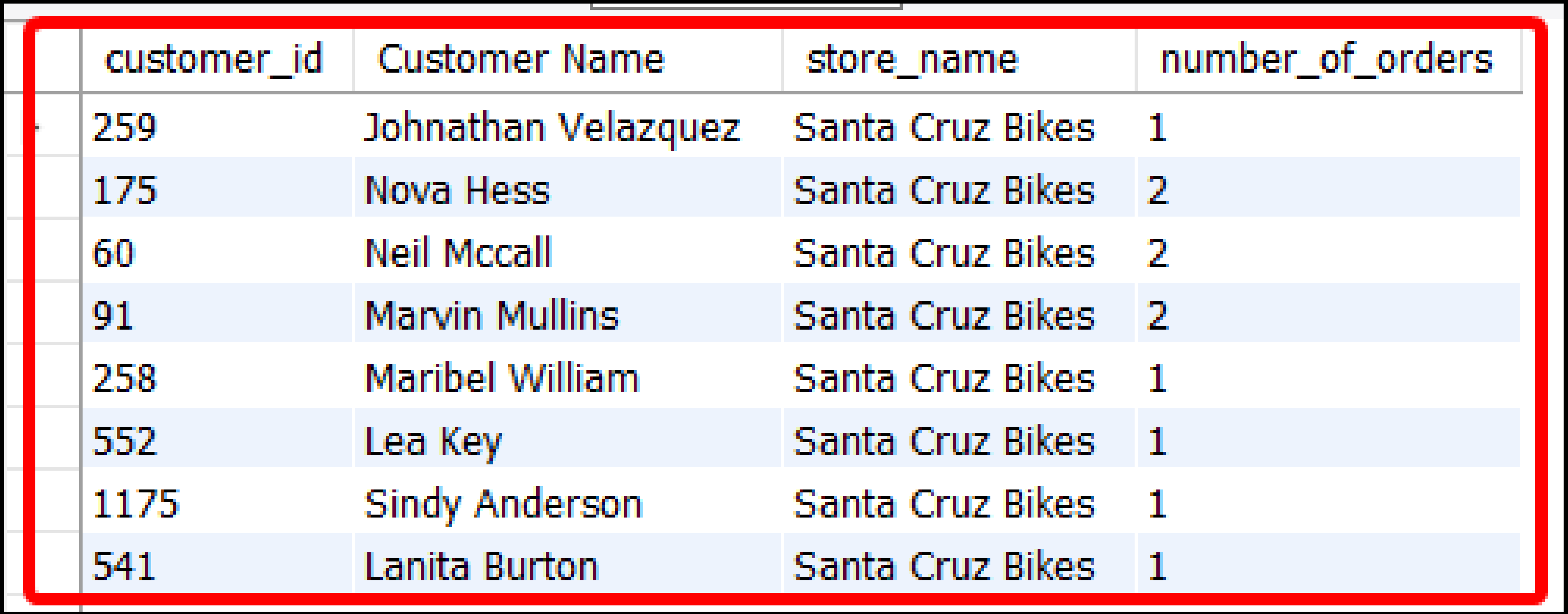
6.Find the total number of orders placed by each customer per store.

QUERY



```sql
SELECT
    c.customer_id,
    CONCAT(first_name, ' ', last_name) "Customer Name",
    s.store_name,
    COUNT(*) 'number_of_orders'
FROM
    customers c
        INNER JOIN
    orders o ON c.customer_id = o.customer_id
        INNER JOIN
    stores s ON s.store_id = o.store_id
GROUP BY c.customer_id , c.first_name , c.last_name , s.store_name;
```

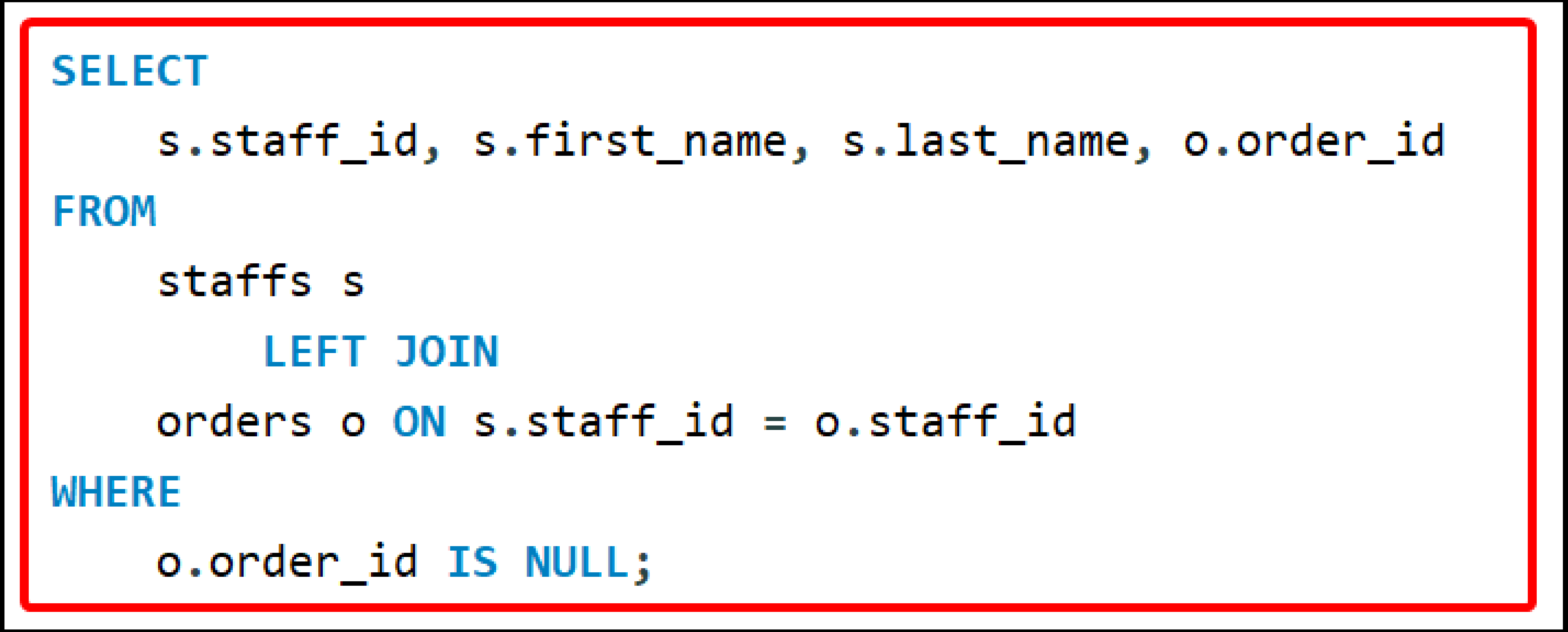
OUTPUT



customer_id	Customer Name	store_name	number_of_orders
259	Johnathan Velazquez	Santa Cruz Bikes	1
175	Nova Hess	Santa Cruz Bikes	2
60	Neil Mccall	Santa Cruz Bikes	2
91	Marvin Mullins	Santa Cruz Bikes	2
258	Maribel William	Santa Cruz Bikes	1
552	Lea Key	Santa Cruz Bikes	1
1175	Sindy Anderson	Santa Cruz Bikes	1
541	Lanita Burton	Santa Cruz Bikes	1

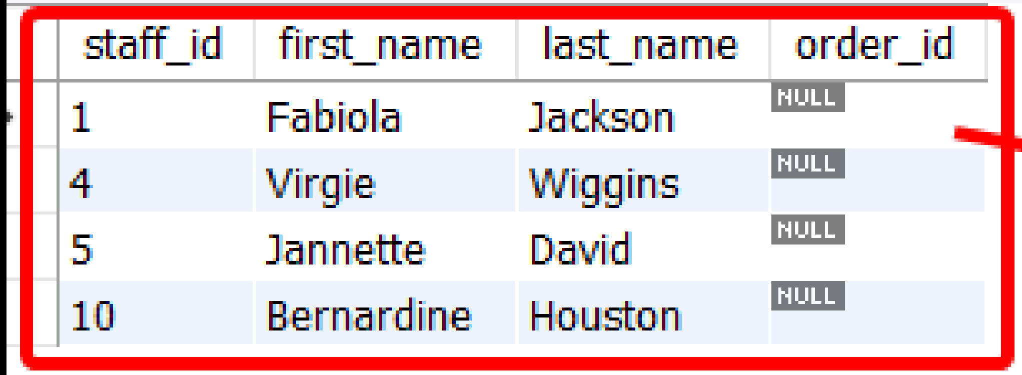
7.Find the names of staff members who have not made any sales.

QUERY



```
SELECT
    s.staff_id, s.first_name, s.last_name, o.order_id
FROM
    staffs s
        LEFT JOIN
    orders o ON s.staff_id = o.staff_id
WHERE
    o.order_id IS NULL;
```

OUTPUT

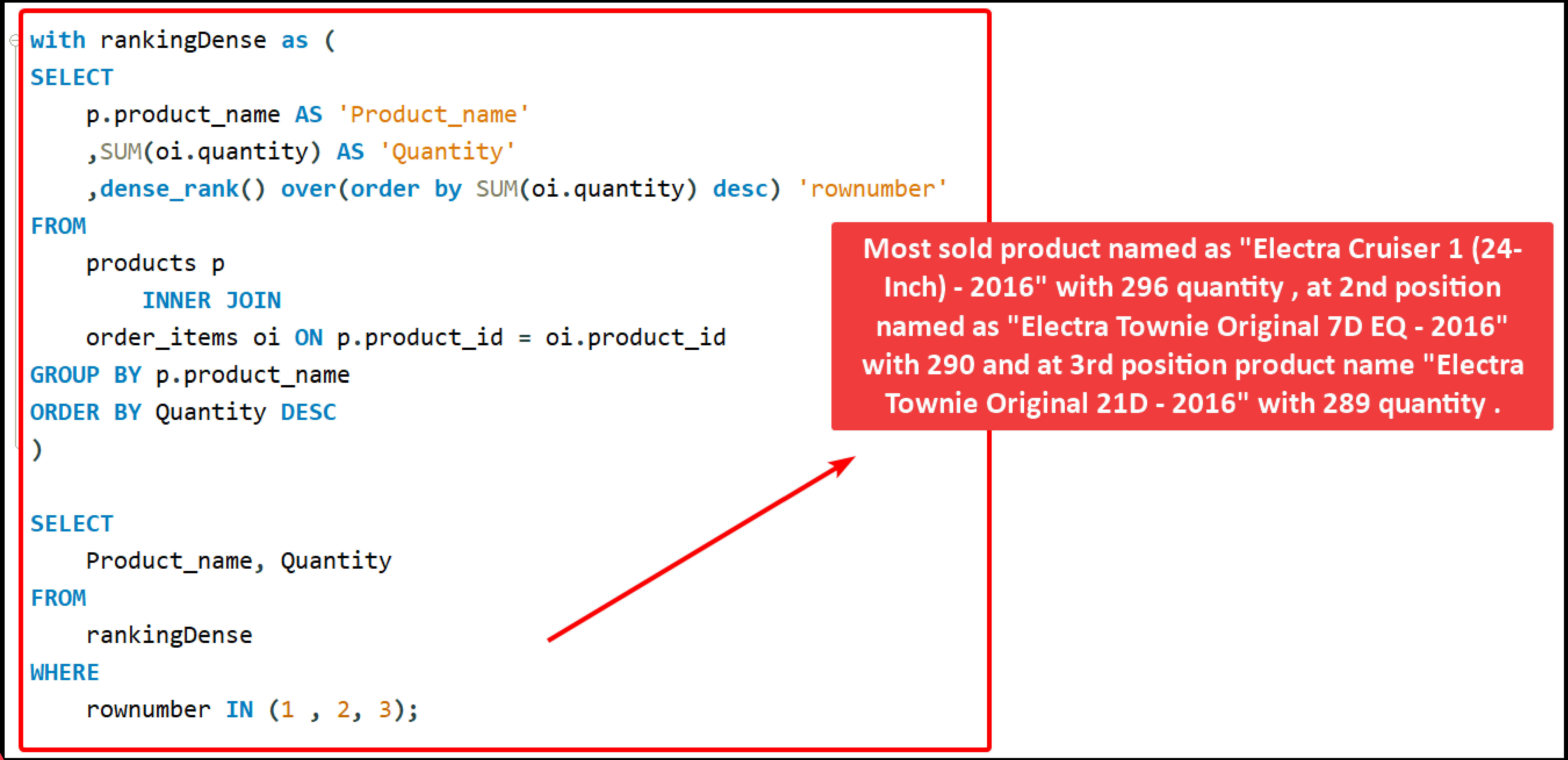


staff_id	first_name	last_name	order_id
1	Fabiola	Jackson	NULL
4	Virgie	Wiggins	NULL
5	Jannette	David	NULL
10	Bernardine	Houston	NULL

These are the employees with no sales yet , need to discuss issues they are facing while managing sales , whether needs to provide training or other kind of support to boost their morale .

8.Find the top 3 most sold products in terms of quantity.

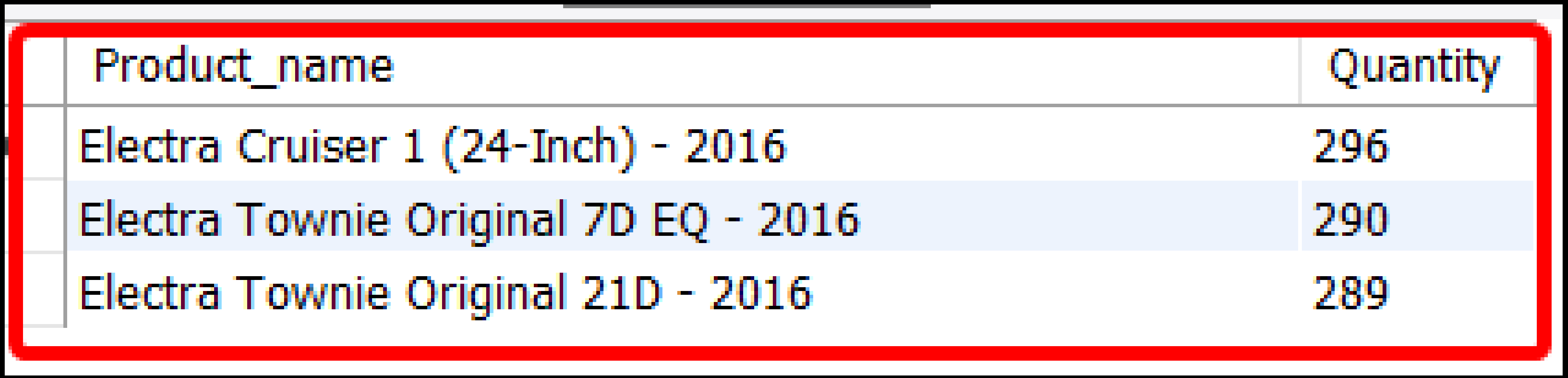
QUERY



```
with rankingDense as (
SELECT
    p.product_name AS 'Product_name'
    ,SUM(oi.quantity) AS 'Quantity'
    ,dense_rank() over(order by SUM(oi.quantity) desc) 'rownumber'
FROM
    products p
        INNER JOIN
    order_items oi ON p.product_id = oi.product_id
GROUP BY p.product_name
ORDER BY Quantity DESC
)

SELECT
    Product_name, Quantity
FROM
    rankingDense
WHERE
    rownumber IN (1 , 2, 3);
```

Most sold product named as "Electra Cruiser 1 (24-Inch) - 2016" with 296 quantity , at 2nd position named as "Electra Townie Original 7D EQ - 2016" with 290 and at 3rd position product name "Electra Townie Original 21D - 2016" with 289 quantity .

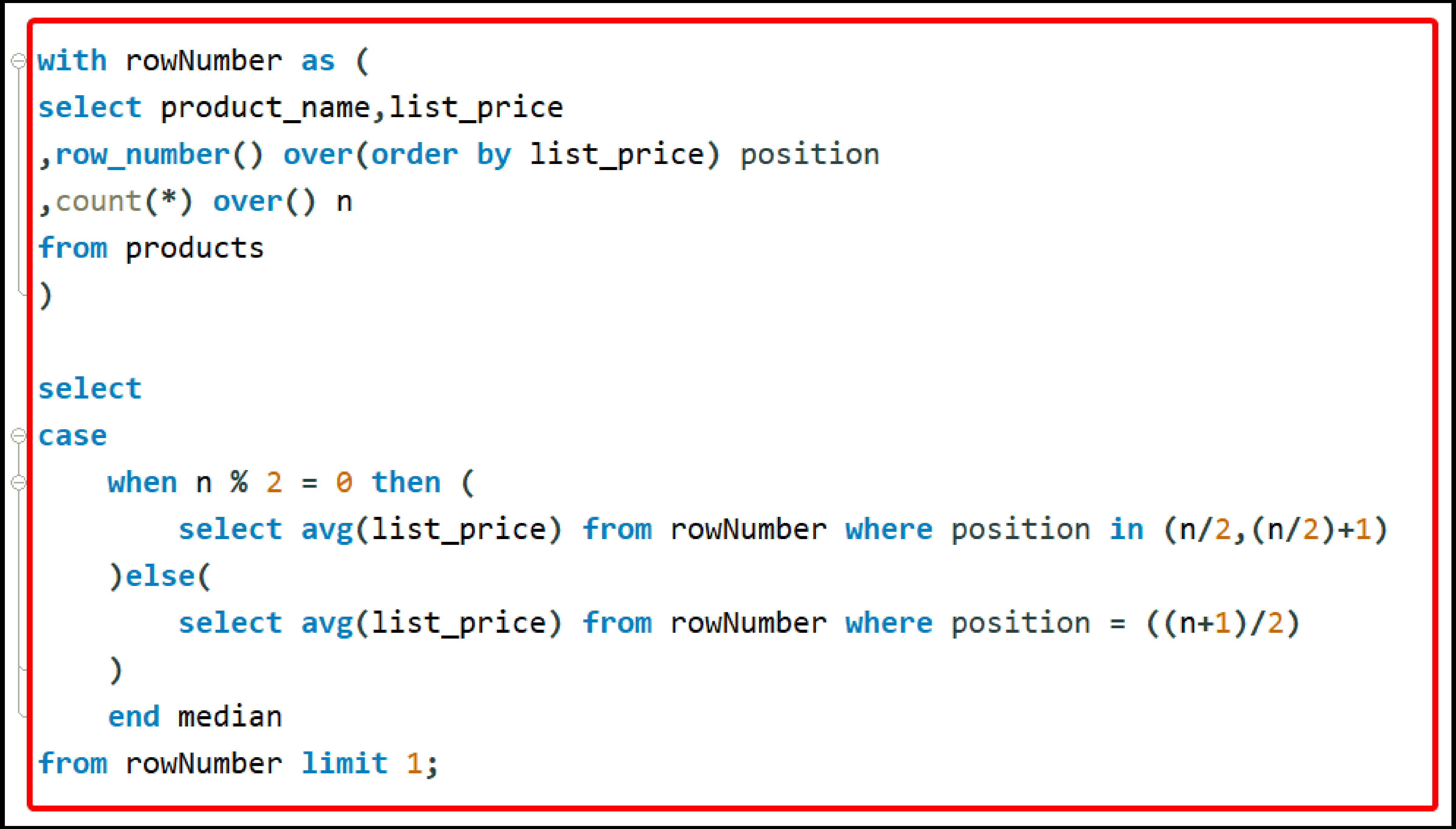
OUTPUT



Product_name	Quantity
Electra Cruiser 1 (24-Inch) - 2016	296
Electra Townie Original 7D EQ - 2016	290
Electra Townie Original 21D - 2016	289

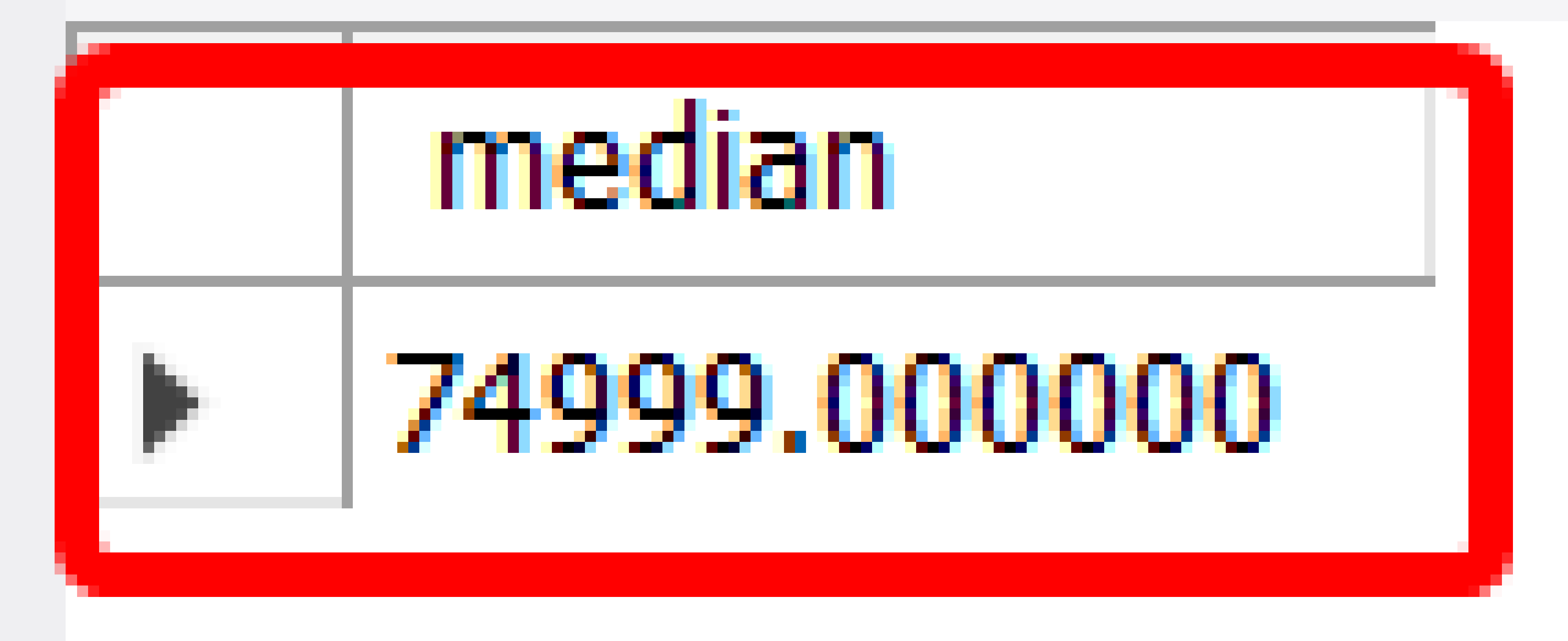
9.Find the median value of the price list.

QUERY



```
with rowNumber as (
select product_name,list_price
,row_number() over(order by list_price) position
,count(*) over() n
from products
)

select
case
    when n % 2 = 0 then (
        select avg(list_price) from rowNumber where position in (n/2,(n/2)+1)
    )else(
        select avg(list_price) from rowNumber where position = ((n+1)/2)
    )
    end median
from rowNumber limit 1;
```

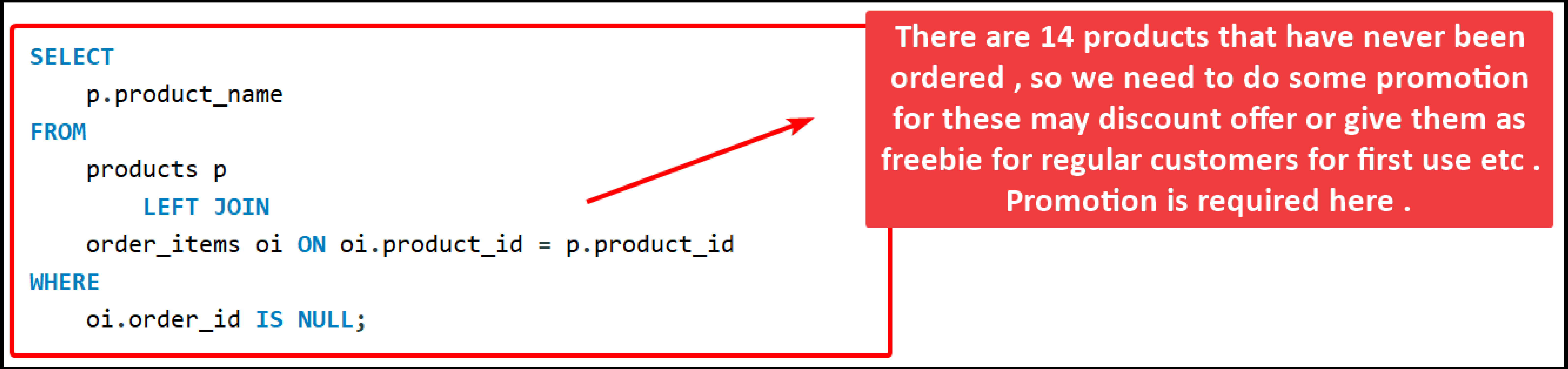
OUTPUT



median
74999.000000

10.List all products that have never been ordered.

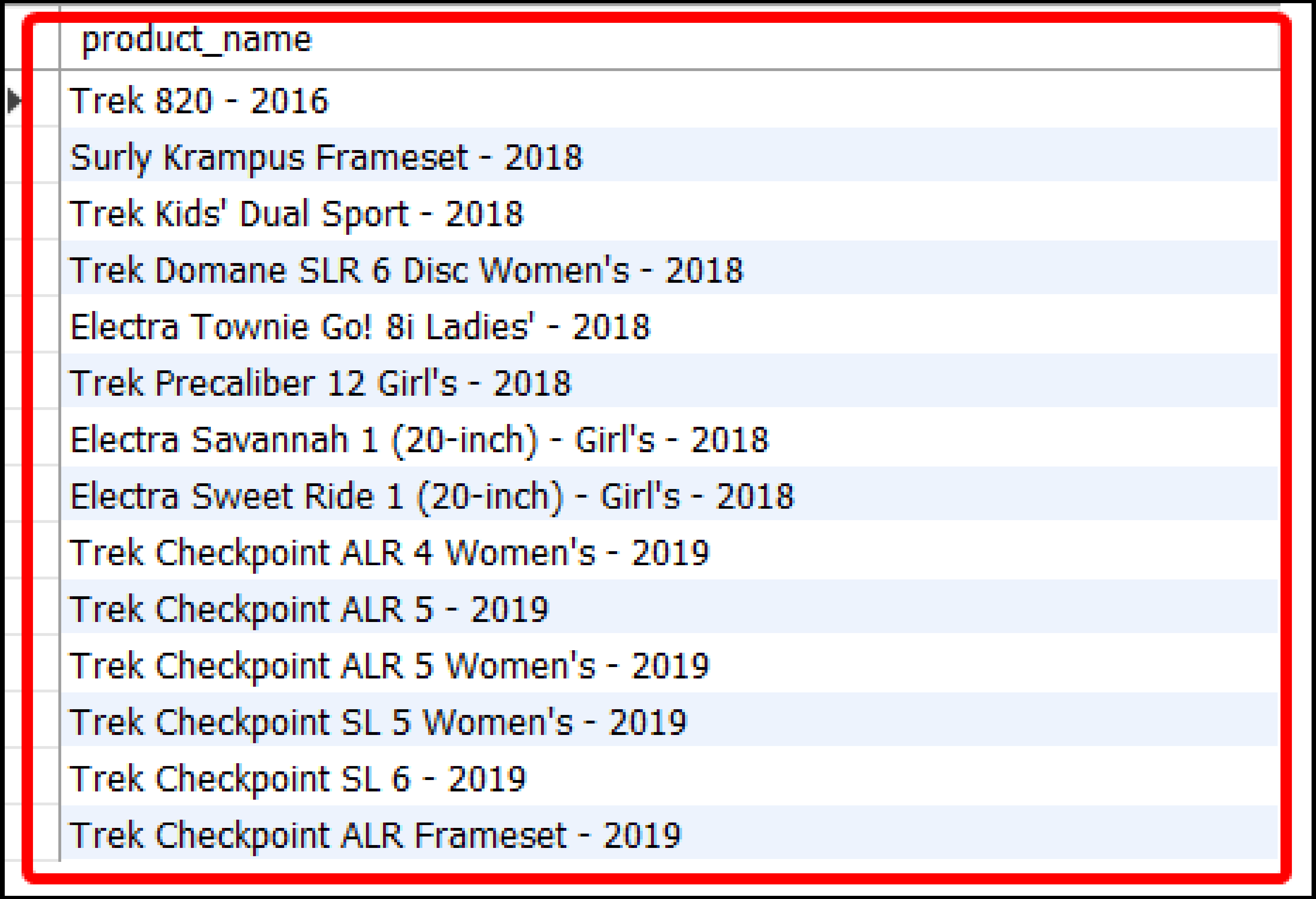
QUERY



```
SELECT
    p.product_name
FROM
    products p
        LEFT JOIN
    order_items oi ON oi.product_id = p.product_id
WHERE
    oi.order_id IS NULL;
```

There are 14 products that have never been ordered , so we need to do some promotion for these may discount offer or give them as freebie for regular customers for first use etc . Promotion is required here .

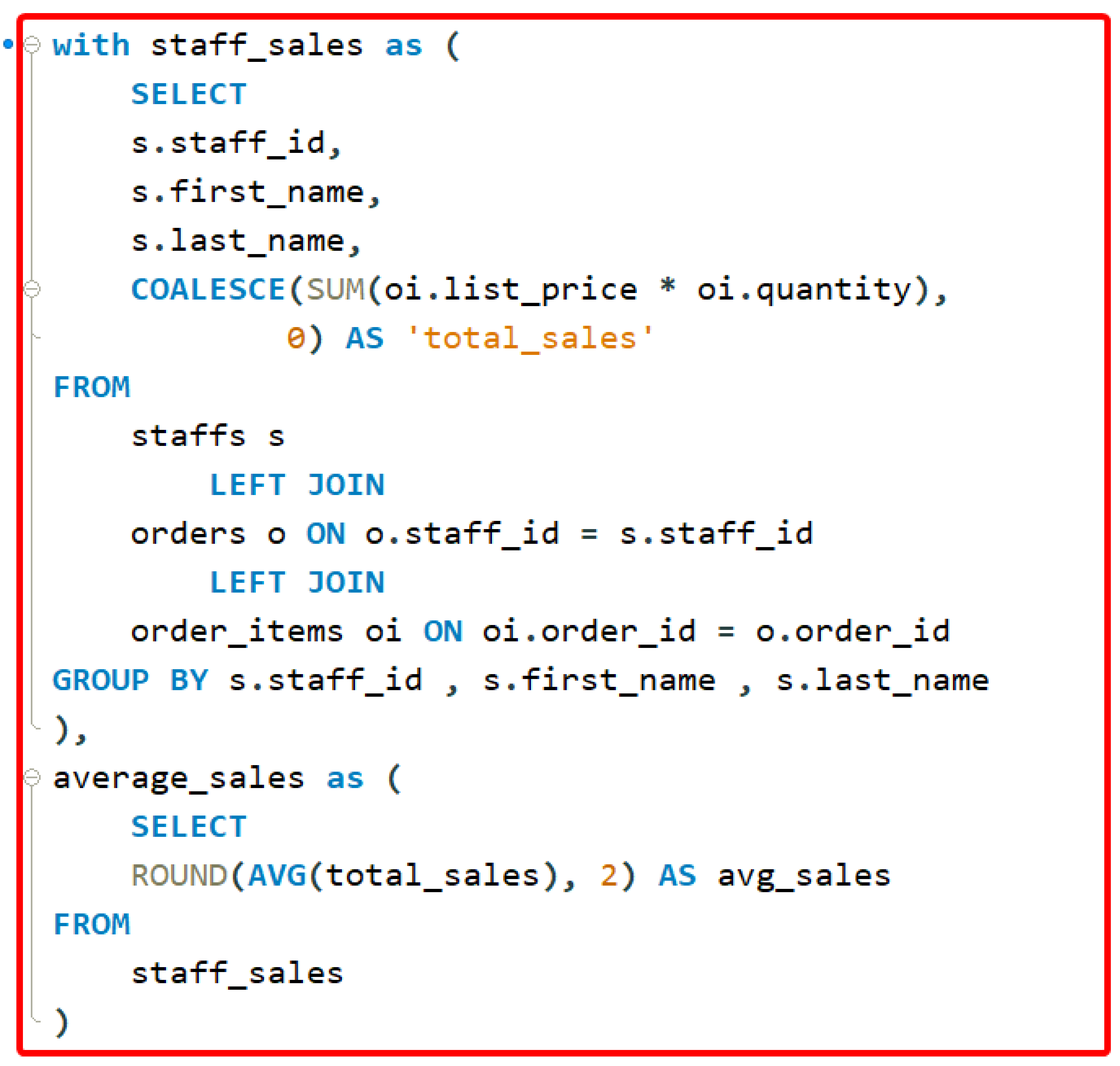
OUTPUT



product_name
Trek 820 - 2016
Surly Krampus Frameset - 2018
Trek Kids' Dual Sport - 2018
Trek Domane SLR 6 Disc Women's - 2018
Electra Townie Go! 8i Ladies' - 2018
Trek Precaliber 12 Girl's - 2018
Electra Savannah 1 (20-inch) - Girl's - 2018
Electra Sweet Ride 1 (20-inch) - Girl's - 2018
Trek Checkpoint ALR 4 Women's - 2019
Trek Checkpoint ALR 5 - 2019
Trek Checkpoint ALR 5 Women's - 2019
Trek Checkpoint SL 5 Women's - 2019
Trek Checkpoint SL 6 - 2019
Trek Checkpoint ALR Frameset - 2019

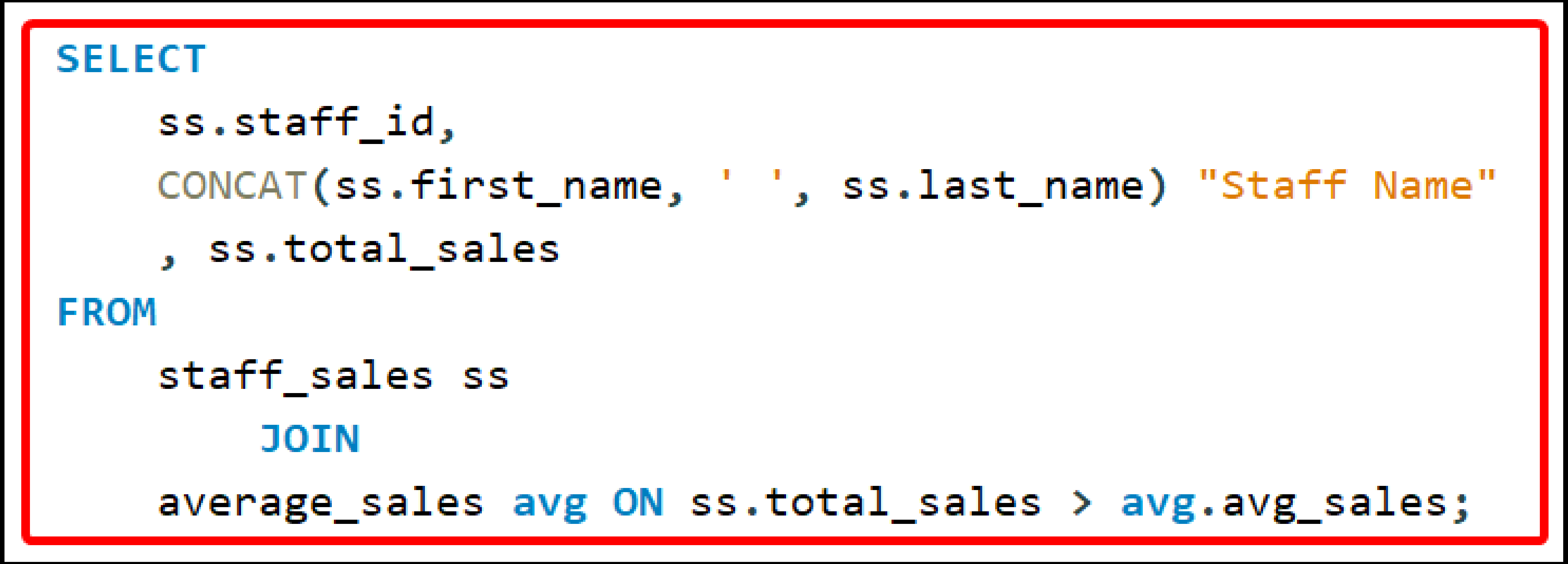
11.List the names of staff members who have made more sales than the average number of sales by all staff members.

QUERY



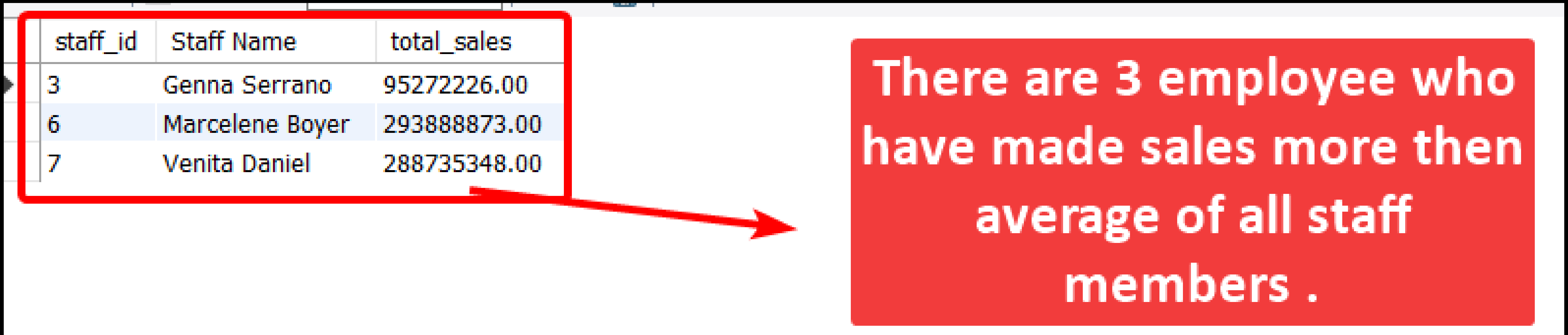
```
with staff_sales as (
    SELECT
    s.staff_id,
    s.first_name,
    s.last_name,
    COALESCE(SUM(oi.list_price * oi.quantity),
            0) AS 'total_sales'
FROM
    staffs s
        LEFT JOIN
    orders o ON o.staff_id = s.staff_id
        LEFT JOIN
    order_items oi ON oi.order_id = o.order_id
GROUP BY s.staff_id , s.first_name , s.last_name
),
average_sales as (
    SELECT
    ROUND(AVG(total_sales), 2) AS avg_sales
FROM
    staff_sales
)
```

QUERY



```sql
SELECT
    ss.staff_id,
    CONCAT(ss.first_name, ' ', ss.last_name) "Staff Name"
    , ss.total_sales
FROM
    staff_sales ss
        JOIN
    average_sales avg ON ss.total_sales > avg.avg_sales;
```

OUTPUT

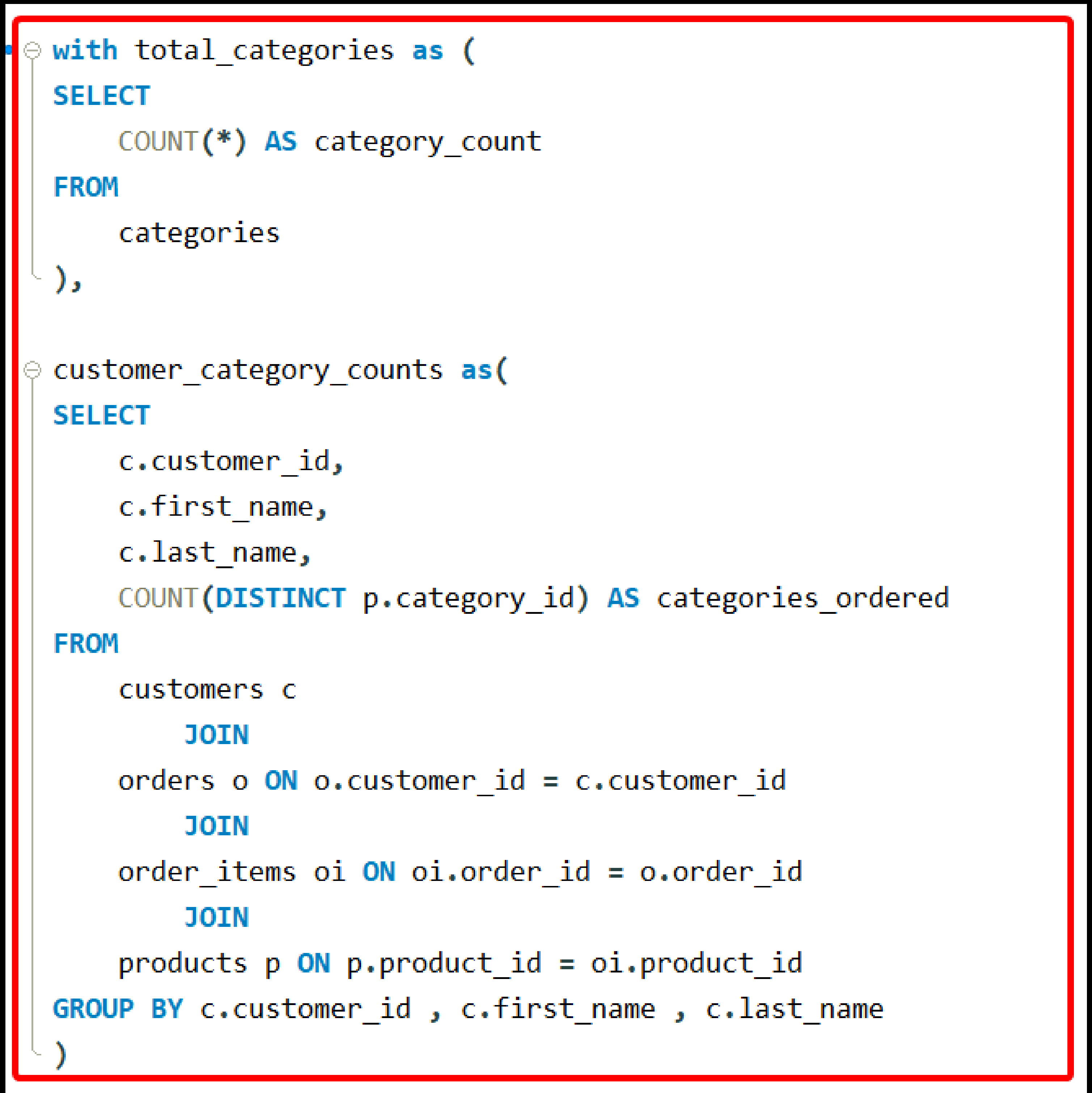


staff_id	Staff Name	total_sales
3	Genna Serrano	95272226.00
6	Marcelene Boyer	293888873.00
7	Venita Daniel	288735348.00

There are 3 employee who have made sales more then average of all staff members .

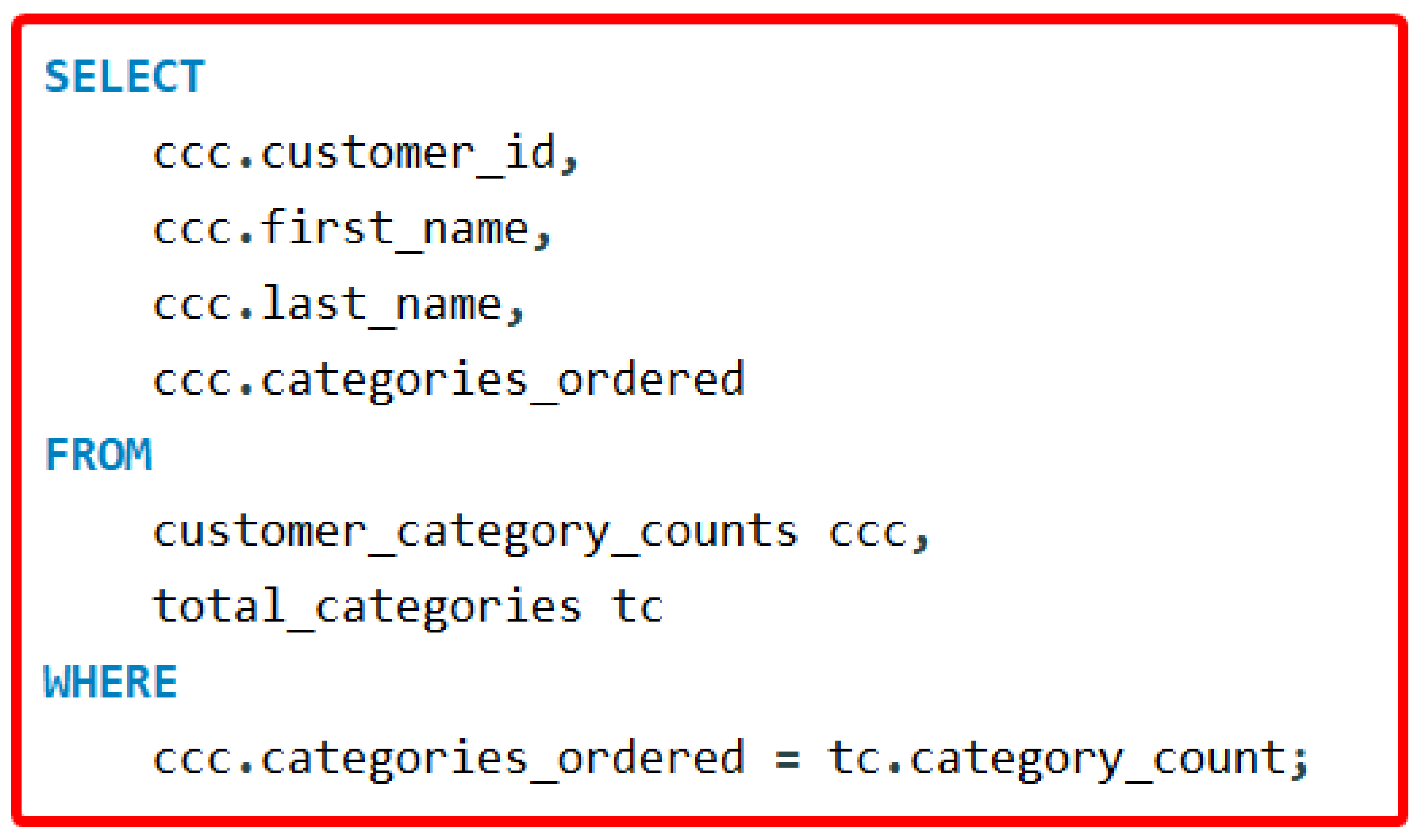
12.Identify the customers who have ordered all types of products (i.e., from every category).

QUERY



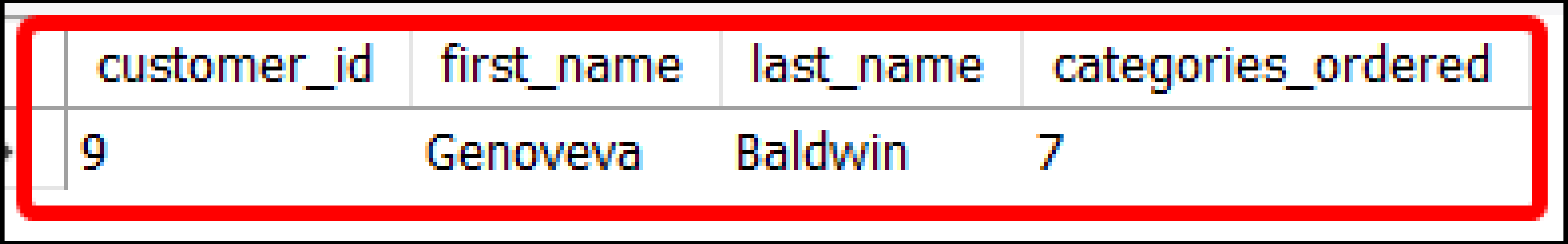
```sql
with total_categories as (
SELECT
    COUNT(*) AS category_count
FROM
    categories
),

customer_category_counts as(
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    COUNT(DISTINCT p.category_id) AS categories_ordered
FROM
    customers c
        JOIN
    orders o ON o.customer_id = c.customer_id
        JOIN
    order_items oi ON oi.order_id = o.order_id
        JOIN
    products p ON p.product_id = oi.product_id
GROUP BY c.customer_id , c.first_name , c.last_name
)
```

QUERY



```
SELECT
    ccc.customer_id,
    ccc.first_name,
    ccc.last_name,
    ccc.categories_ordered
FROM
    customer_category_counts ccc,
    total_categories tc
WHERE
    ccc.categories_ordered = tc.category_count;
```

OUTPUT



customer_id	first_name	last_name	categories_ordered
9	Genoveva	Baldwin	7

THANK YOU

2025 Jenson USA Presentation

BY FAROOK MOHAMMAD