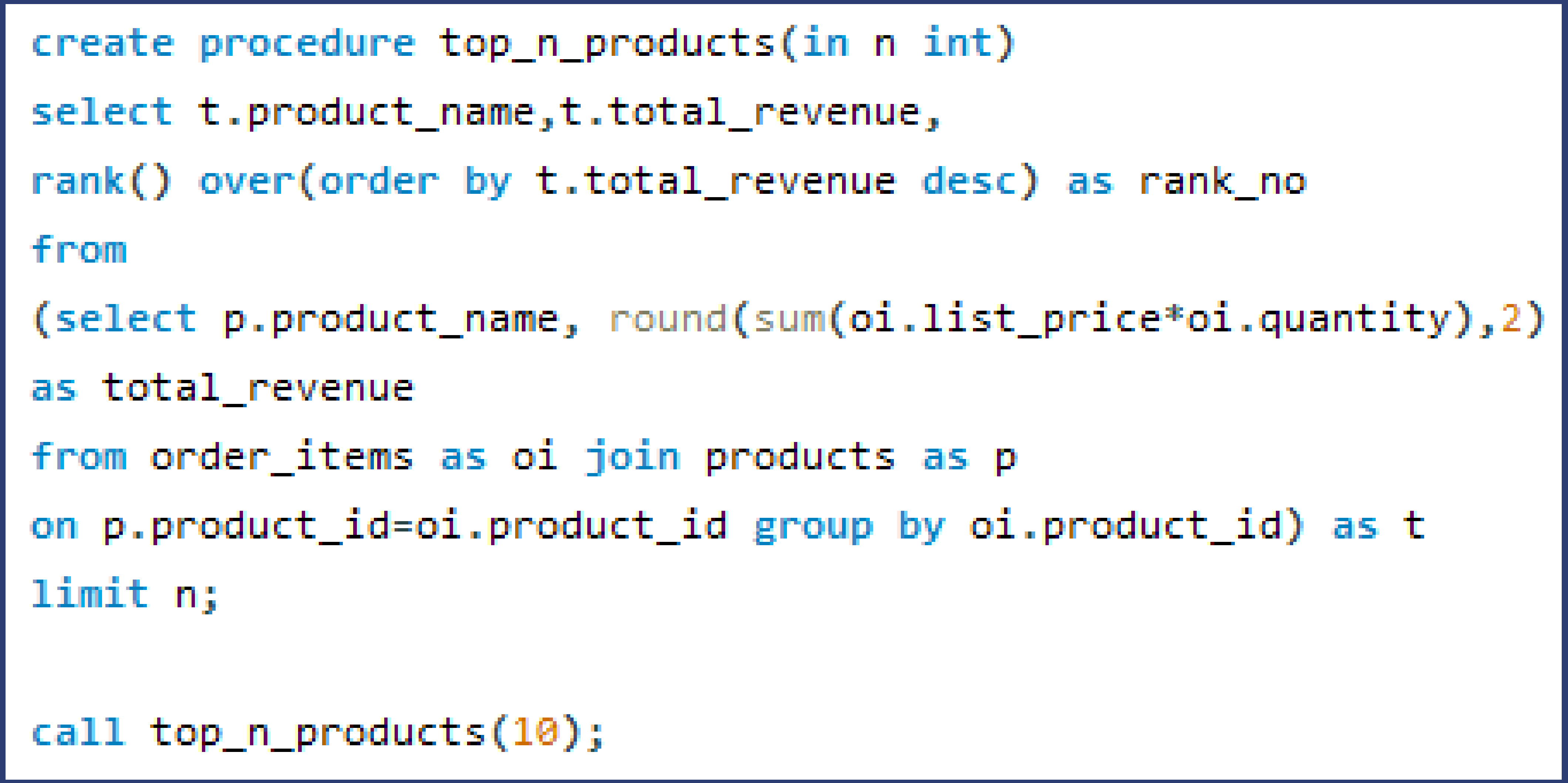
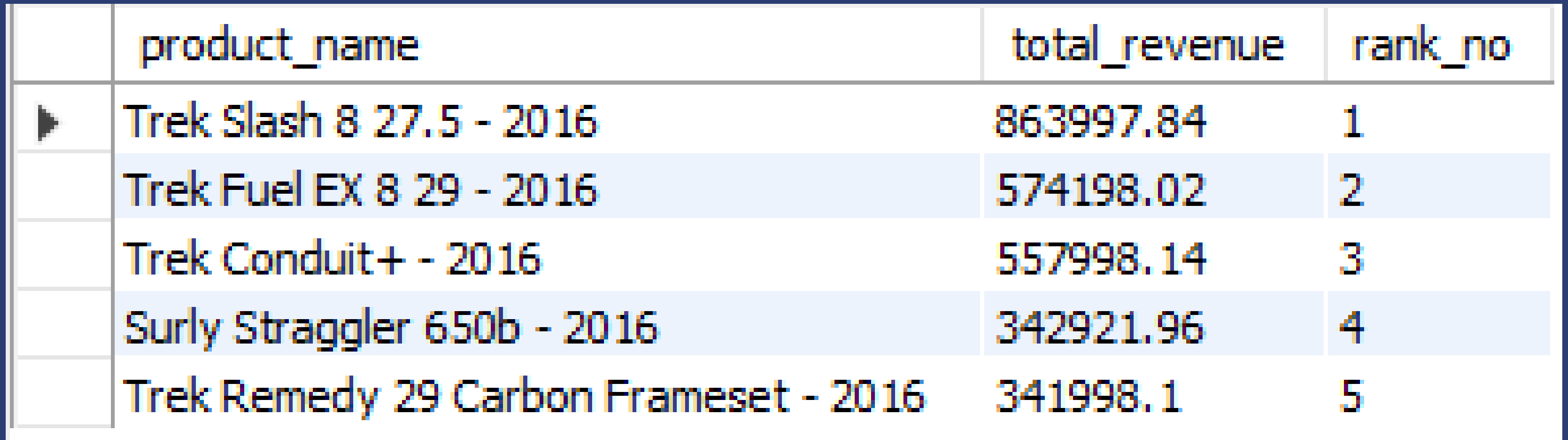
# PROJECT

RETAIL BIKE STORE ANALYSIS

# Ranking products based on total revenue and display the top n products



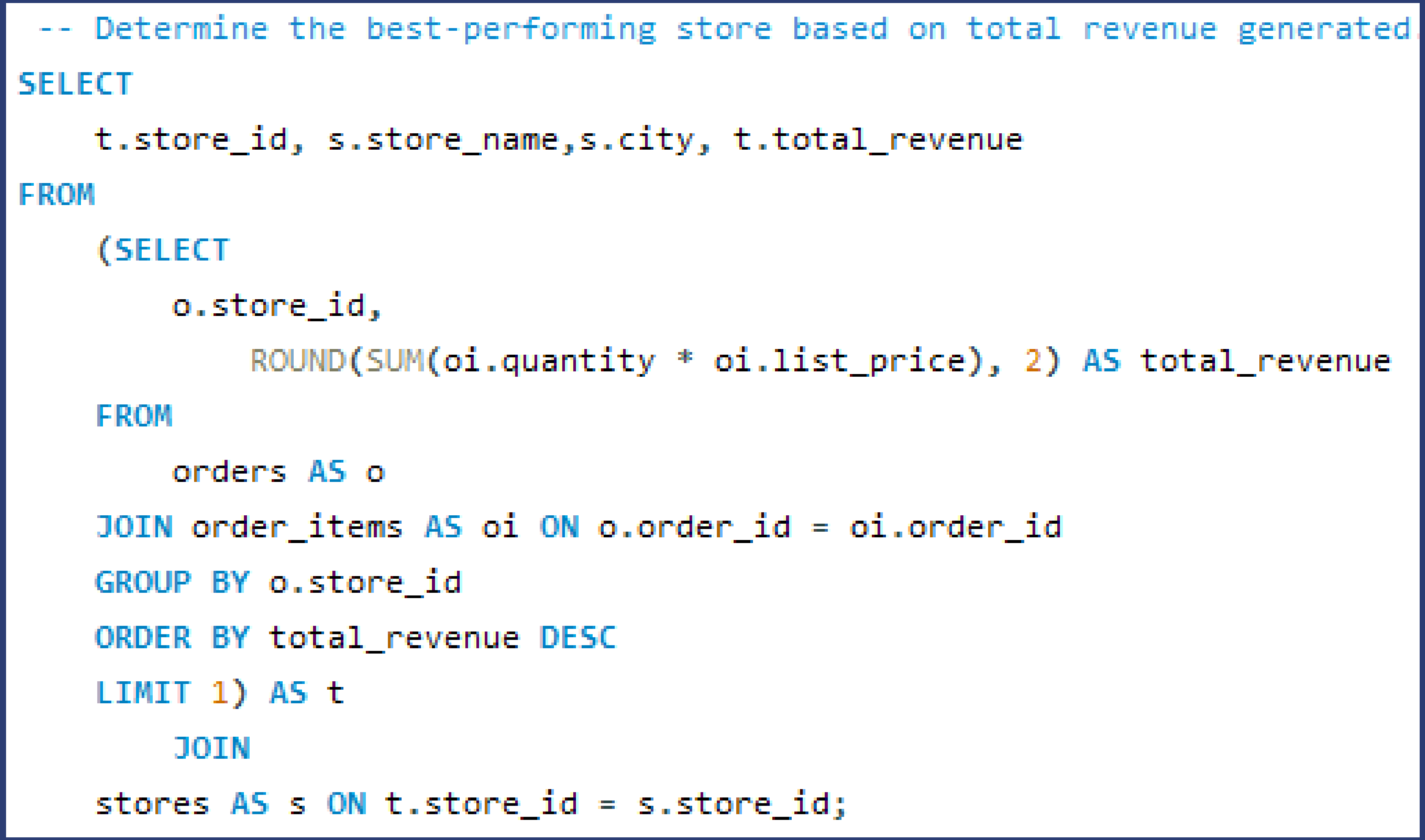
```
create procedure top_n_products(in n int)
select t.product_name,t.total_revenue,
rank() over(order by t.total_revenue desc) as rank_no
from
(select p.product_name, round(sum(oi.list_price*oi.quantity),2)
as total_revenue
from order_items as oi join products as p
on p.product_id=oi.product_id group by oi.product_id) as t
limit n;

call top_n_products(10);
```



| | product_name | total_revenue | rank_no |
|---|---|---|---|
| ▶ | Trek Slash 8 27.5 - 2016 | 863997.84 | 1 |
| | Trek Fuel EX 8 29 - 2016 | 574198.02 | 2 |
| | Trek Conduit+ - 2016 | 557998.14 | 3 |
| | Surly Straggler 650b - 2016 | 342921.96 | 4 |
| | Trek Remedy 29 Carbon Frameset - 2016 | 341998.1 | 5 |

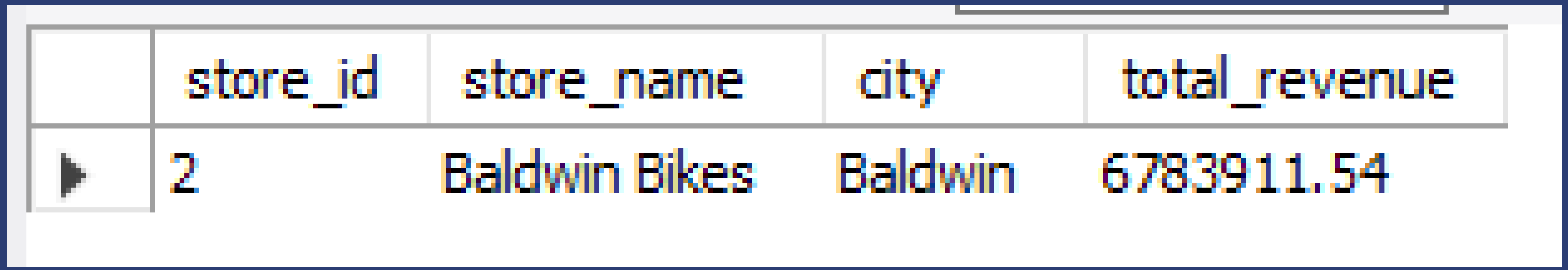
- The largest portion of total revenue is produced by the top 5 products.
- The Trek Slash 8 contributes more to total revenue than any other product.
- Strategic inventory planning and marketing focus are aided by this insight.

# Determining the best-performing store based on total revenue generated.



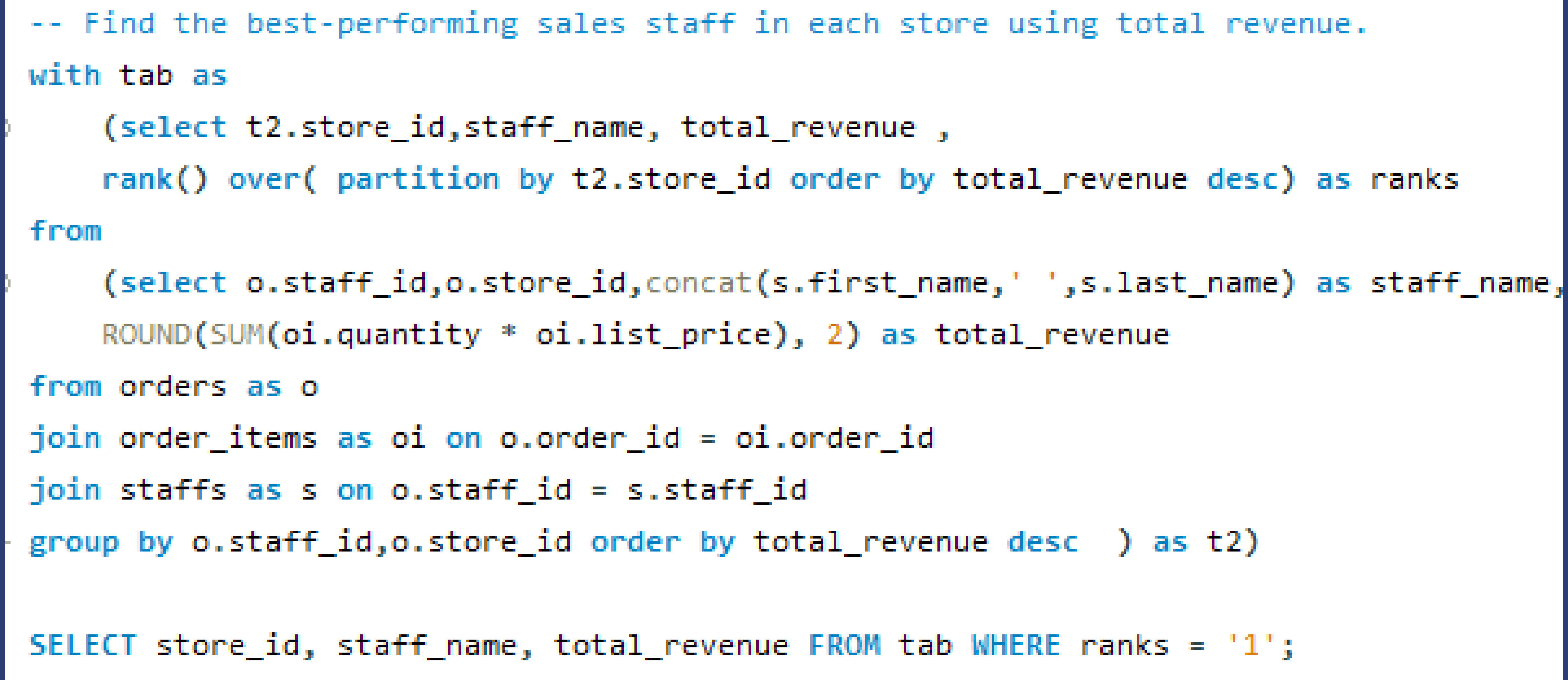
```sql
 -- Determine the best-performing store based on total revenue generated
SELECT
    t.store_id, s.store_name,s.city, t.total_revenue
FROM
    (SELECT
        o.store_id,
            ROUND(SUM(oi.quantity * oi.list_price), 2) AS total_revenue
    FROM
        orders AS o
    JOIN order_items AS oi ON o.order_id = oi.order_id
    GROUP BY o.store_id
    ORDER BY total_revenue DESC
    LIMIT 1) AS t
        JOIN
    stores AS s ON t.store_id = s.store_id;
```

- Out of all the stores, Store ID 2 has produced the most total revenue.
- This indicates better sales performance and stronger customer demand in *Baldwin*

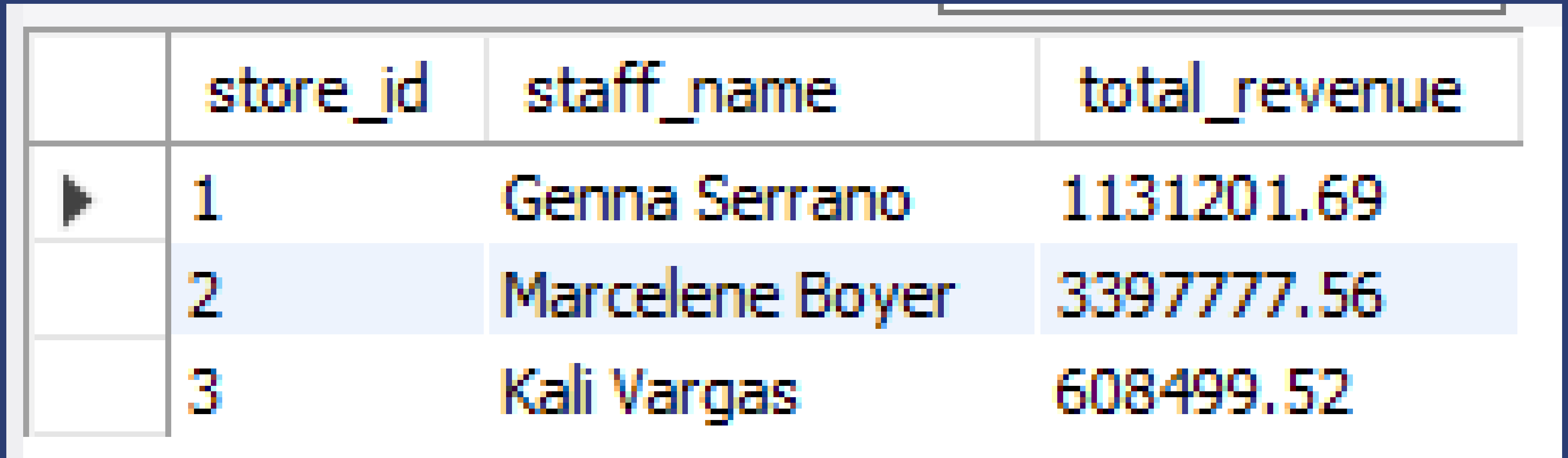


|  | store_id | store_name | city | total_revenue |
|---|---|---|---|---|
| ▶ | 2 | Baldwin Bikes | Baldwin | 6783911.54 |

# best-performing sales staff in each store using total revenue



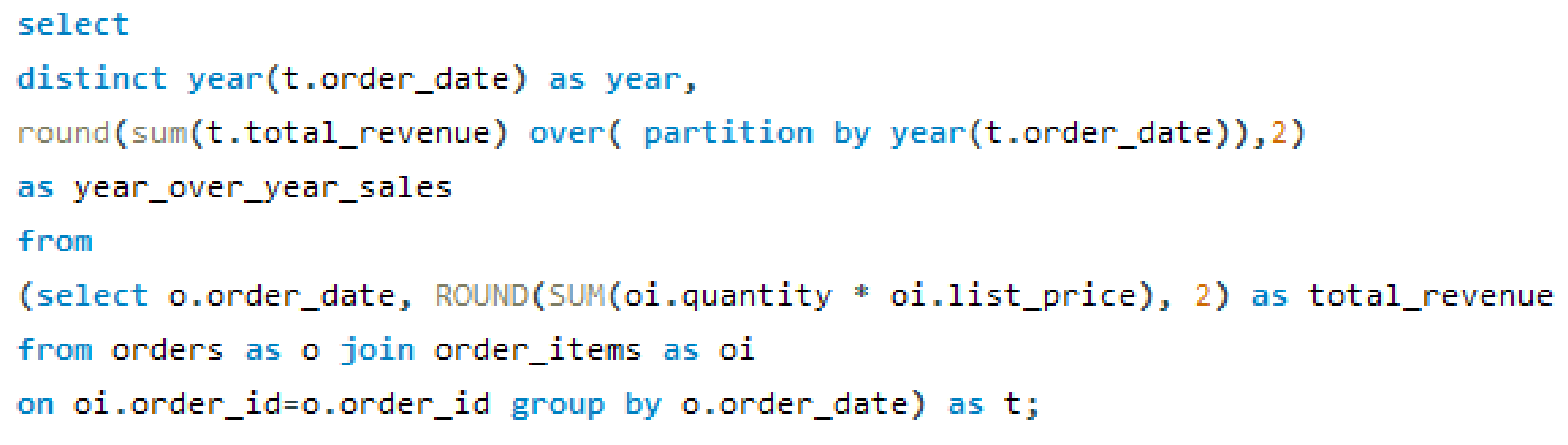
```sql
-- Find the best-performing sales staff in each store using total revenue.
with tab as
    (select t2.store_id,staff_name, total_revenue ,
    rank() over( partition by t2.store_id order by total_revenue desc) as ranks
from
    (select o.staff_id,o.store_id,concat(s.first_name,' ',s.last_name) as staff_name,
    ROUND(SUM(oi.quantity * oi.list_price), 2) as total_revenue
from orders as o
join order_items as oi on o.order_id = oi.order_id
join staffs as s on o.staff_id = s.staff_id
group by o.staff_id,o.store_id order by total_revenue desc  ) as t2)

SELECT store_id, staff_name, total_revenue FROM tab WHERE ranks = '1';
```



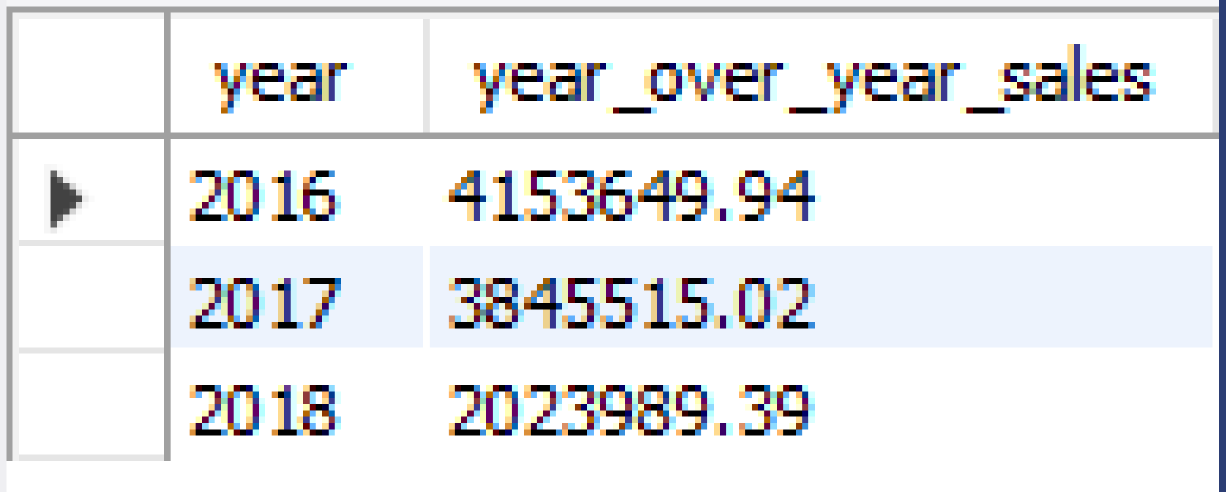
| | store_id | staff_name | total_revenue |
|---|---|---|---|
| ▶ | 1 | Genna Serrano | 1131201.69 |
| | 2 | Marcelene Boyer | 3397777.56 |
| | 3 | Kali Vargas | 608499.52 |

- Based on overall sales, each store has a single top-performing salesperson.
- *Marcelene Boyer* (Store 2) leads across all stores, showing exceptional performance.
- Management can analyze her strategies to replicate success in other stores.

# Generating a report showing year-over-year sales revenue growth



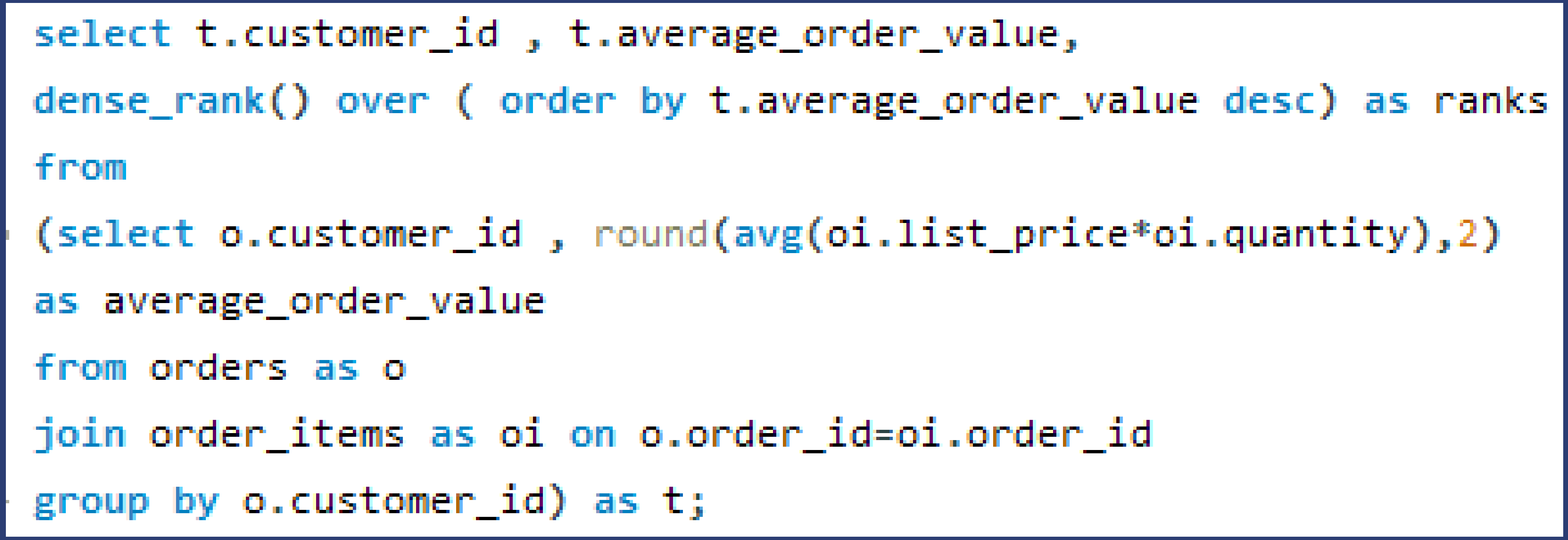
```
select
distinct year(t.order_date) as year,
round(sum(t.total_revenue) over( partition by year(t.order_date)),2)
as year_over_year_sales
from
(select o.order_date, ROUND(SUM(oi.quantity * oi.list_price), 2) as total_revenue
from orders as o join order_items as oi
on oi.order_id=o.order_id group by o.order_date) as t;
```



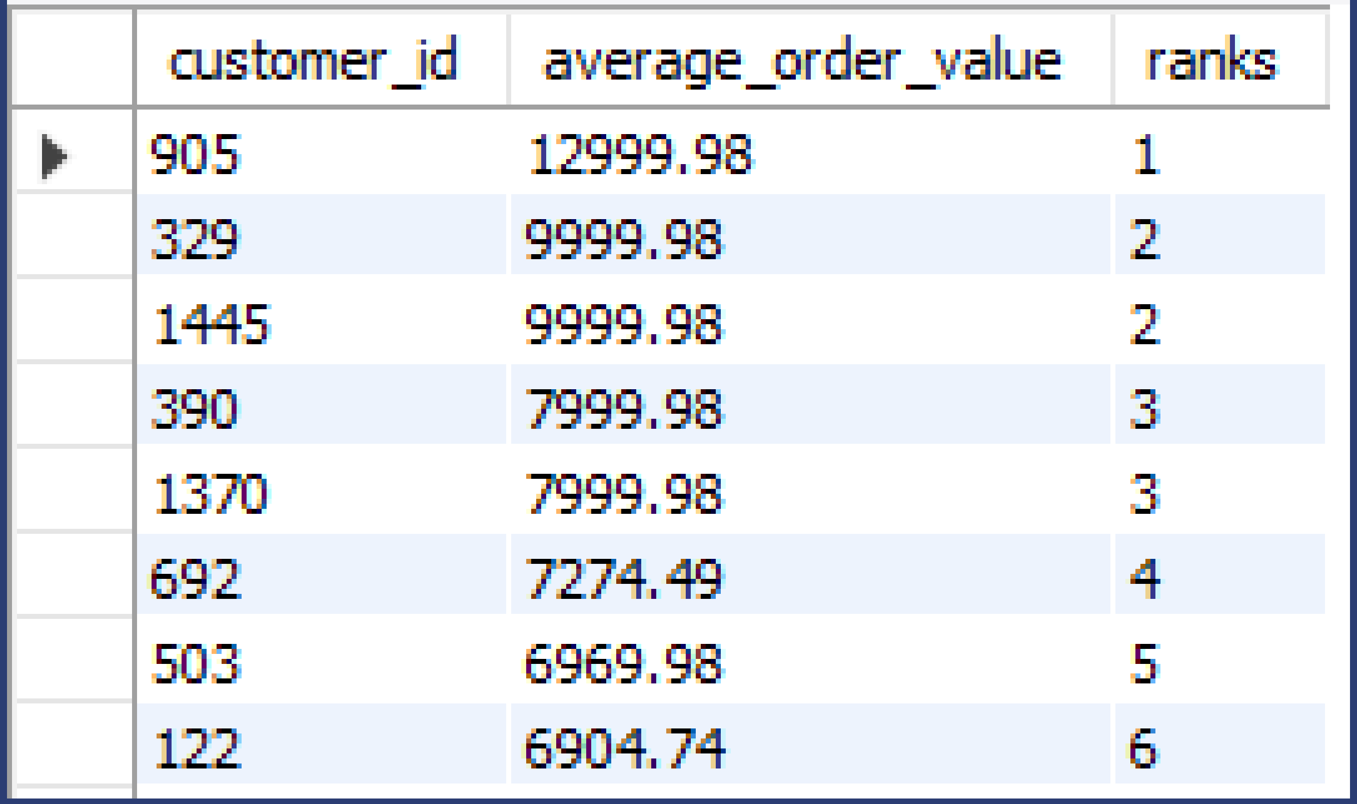
| year | year_over_year_sales |
|---|---|
| 2016 | 4153649.94 |
| 2017 | 3845515.02 |
| 2018 | 2023989.39 |

- Over three years, the company's revenue has decreased drastically.
- Compared to 2016, 2017 saw a 7.4% YoY sales decreases.
- Similarly, in 2018 saw a 51.28% YoY sales decreases.
- To increase their sales growth, management should change their marketing strategies and stock expansion expenditures.

# average order value (AOV) for each customer from highest to lowest



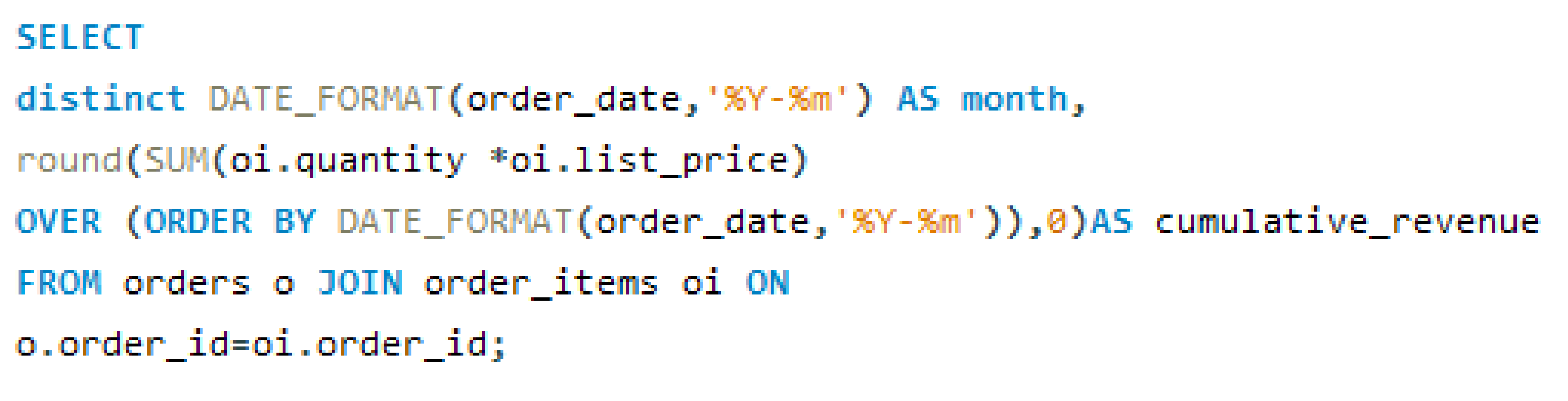
```
select t.customer_id , t.average_order_value,
dense_rank() over ( order by t.average_order_value desc) as ranks
from
(select o.customer_id , round(avg(oi.list_price*oi.quantity),2)
as average_order_value
from orders as o
join order_items as oi on o.order_id=oi.order_id
group by o.customer_id) as t;
```



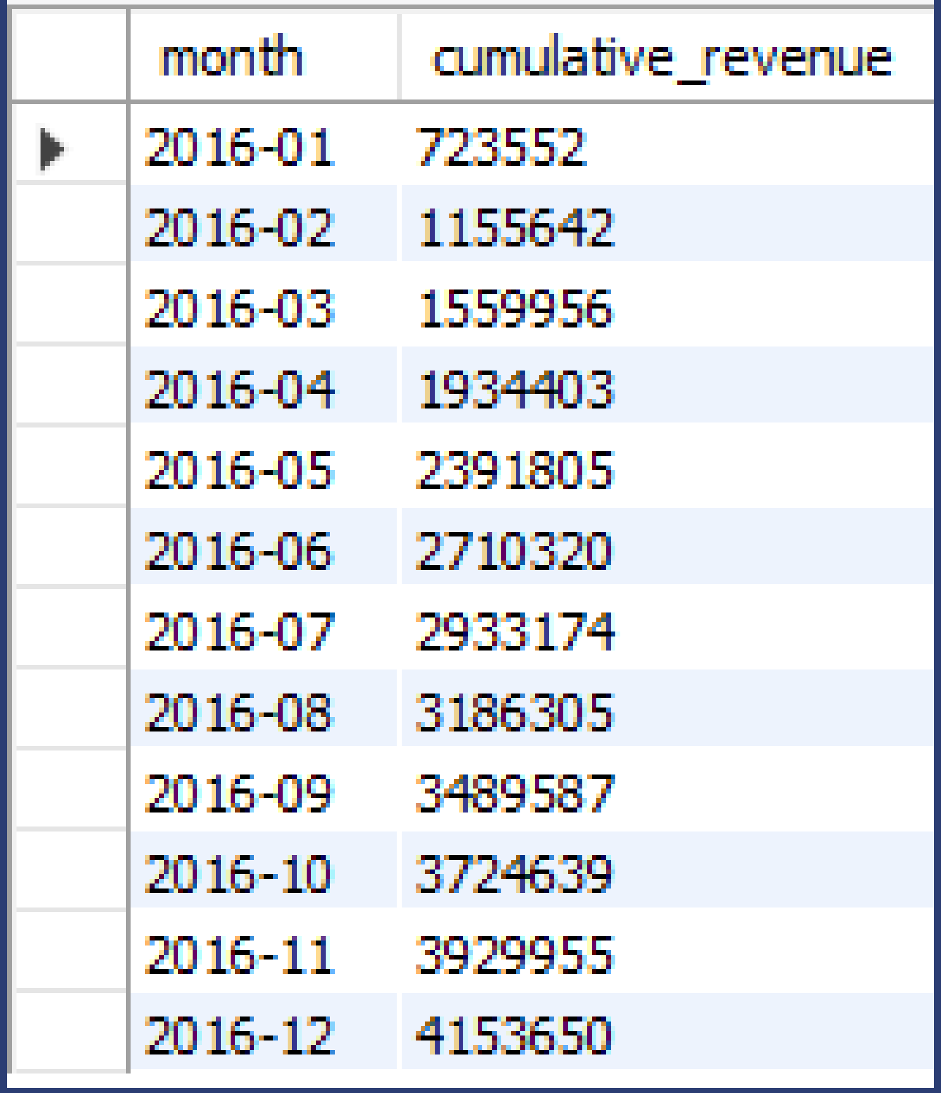
| customer_id | average_order_value | ranks |
|---|---|---|
| 905 | 12999.98 | 1 |
| 329 | 9999.98 | 2 |
| 1445 | 9999.98 | 2 |
| 390 | 7999.98 | 3 |
| 1370 | 7999.98 | 3 |
| 692 | 7274.49 | 4 |
| 503 | 6969.98 | 5 |
| 122 | 6904.74 | 6 |

- Customers with higher AOV are high-value customers contributing more revenue.
- The top 5 customers have an AOV above $6,000 each.
- These customers are ideal candidates for loyalty programs, exclusive offers, and premium marketing strategies.

# cumulative revenue month by month in 2016



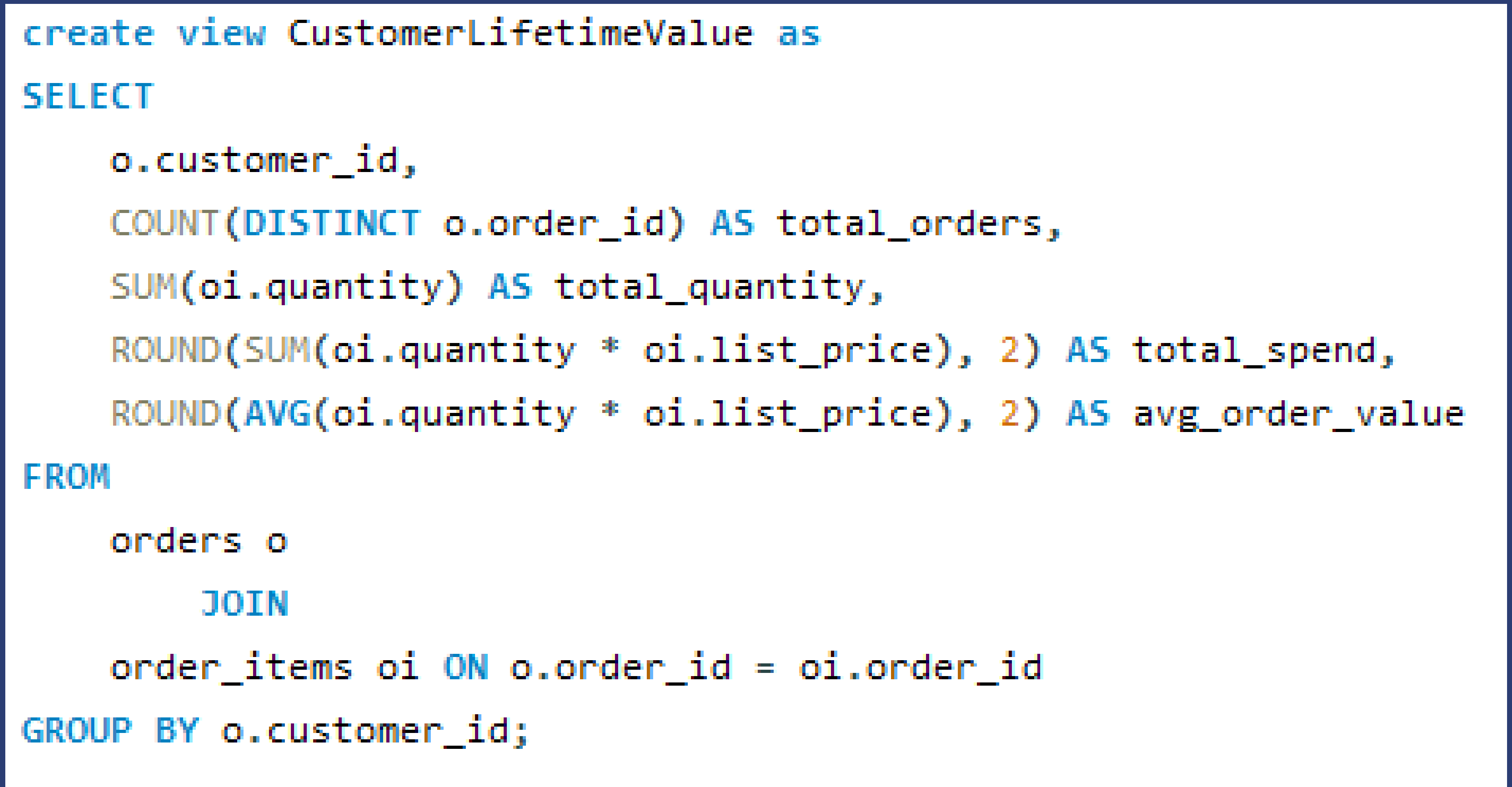
```
SELECT
distinct DATE_FORMAT(order_date,'%Y-%m') AS month,
round(SUM(oi.quantity *oi.list_price)
OVER (ORDER BY DATE_FORMAT(order_date,'%Y-%m')),0)AS cumulative_revenue
FROM orders o JOIN order_items oi ON
o.order_id=oi.order_id;
```



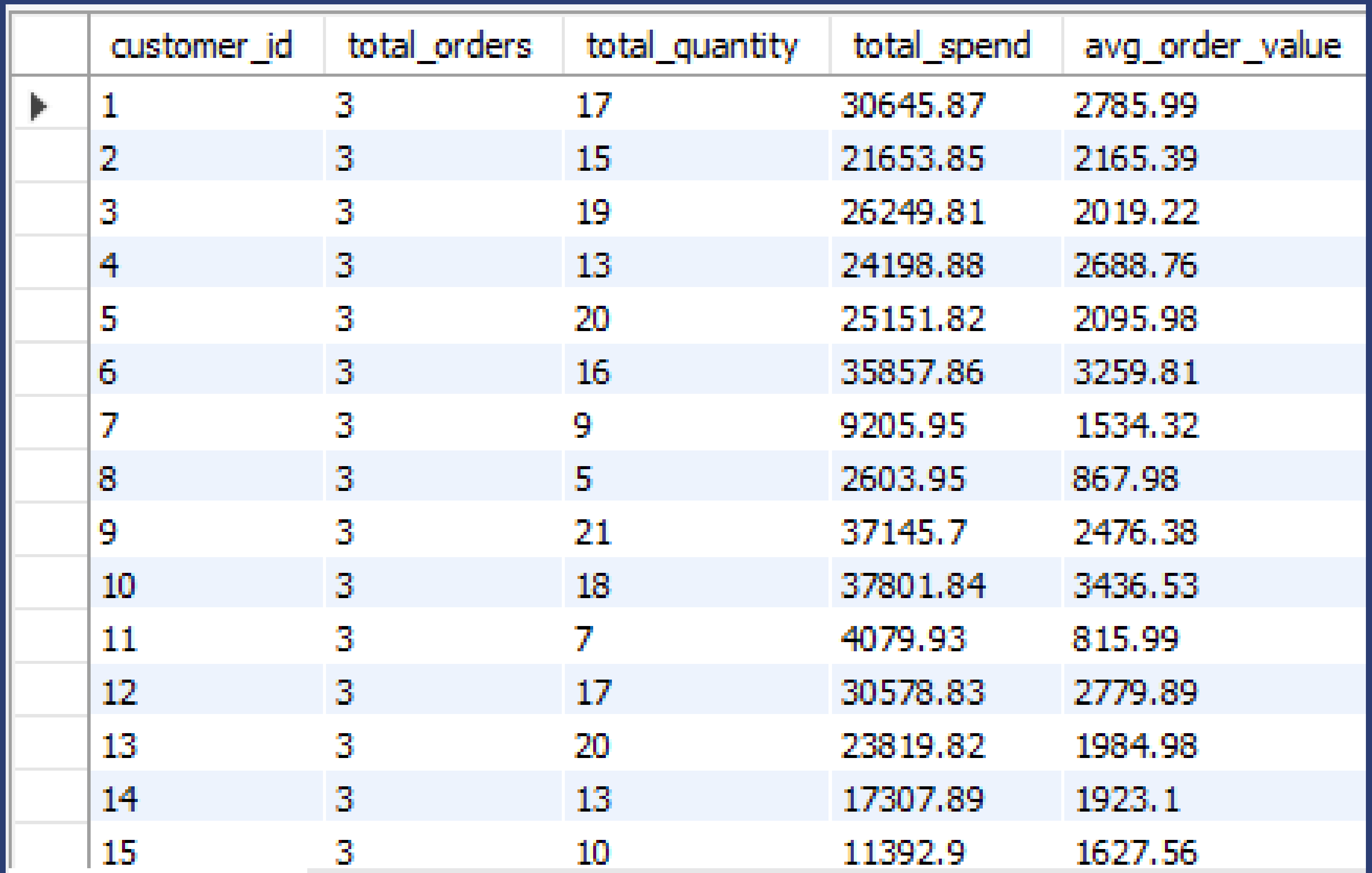
| month | cumulative_revenue |
|---|---|
| 2016-01 | 723552 |
| 2016-02 | 1155642 |
| 2016-03 | 1559956 |
| 2016-04 | 1934403 |
| 2016-05 | 2391805 |
| 2016-06 | 2710320 |
| 2016-07 | 2933174 |
| 2016-08 | 3186305 |
| 2016-09 | 3489587 |
| 2016-10 | 3724639 |
| 2016-11 | 3929955 |
| 2016-12 | 4153650 |

- Cumulative revenue steadily increased month-over-month in 2016.
- By December 2016, the total revenue reached $4.15M.
- Revenue spikes may indicate seasonal demands, promotions or holiday effects.
- This helps management track growth trends and plan inventory for peak months.

# ... A CLV report showing total orders,quantity,spend and average order value ...

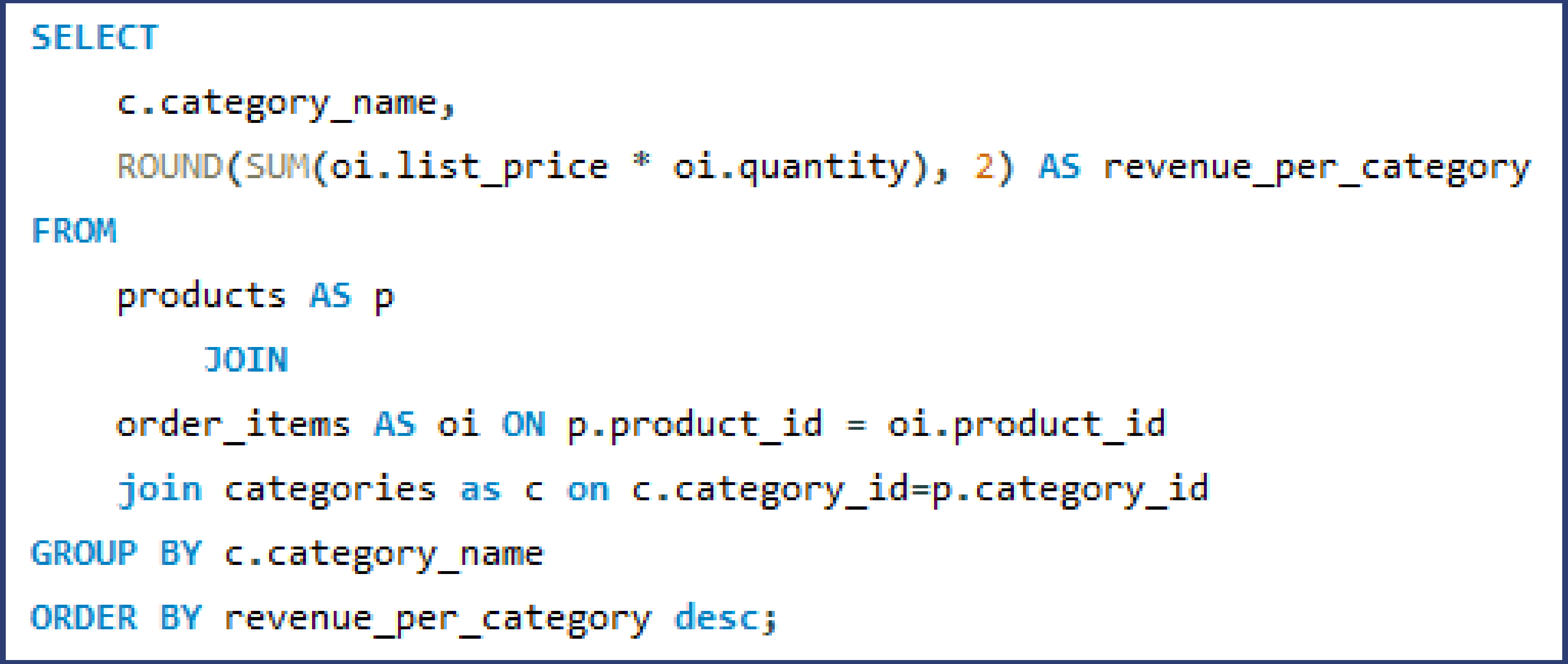


```
create view CustomerLifetimeValue as
SELECT
    o.customer_id,
    COUNT(DISTINCT o.order_id) AS total_orders,
    SUM(oi.quantity) AS total_quantity,
    ROUND(SUM(oi.quantity * oi.list_price), 2) AS total_spend,
    ROUND(AVG(oi.quantity * oi.list_price), 2) AS avg_order_value
FROM
    orders o
        JOIN
    order_items oi ON o.order_id = oi.order_id
GROUP BY o.customer_id;
```

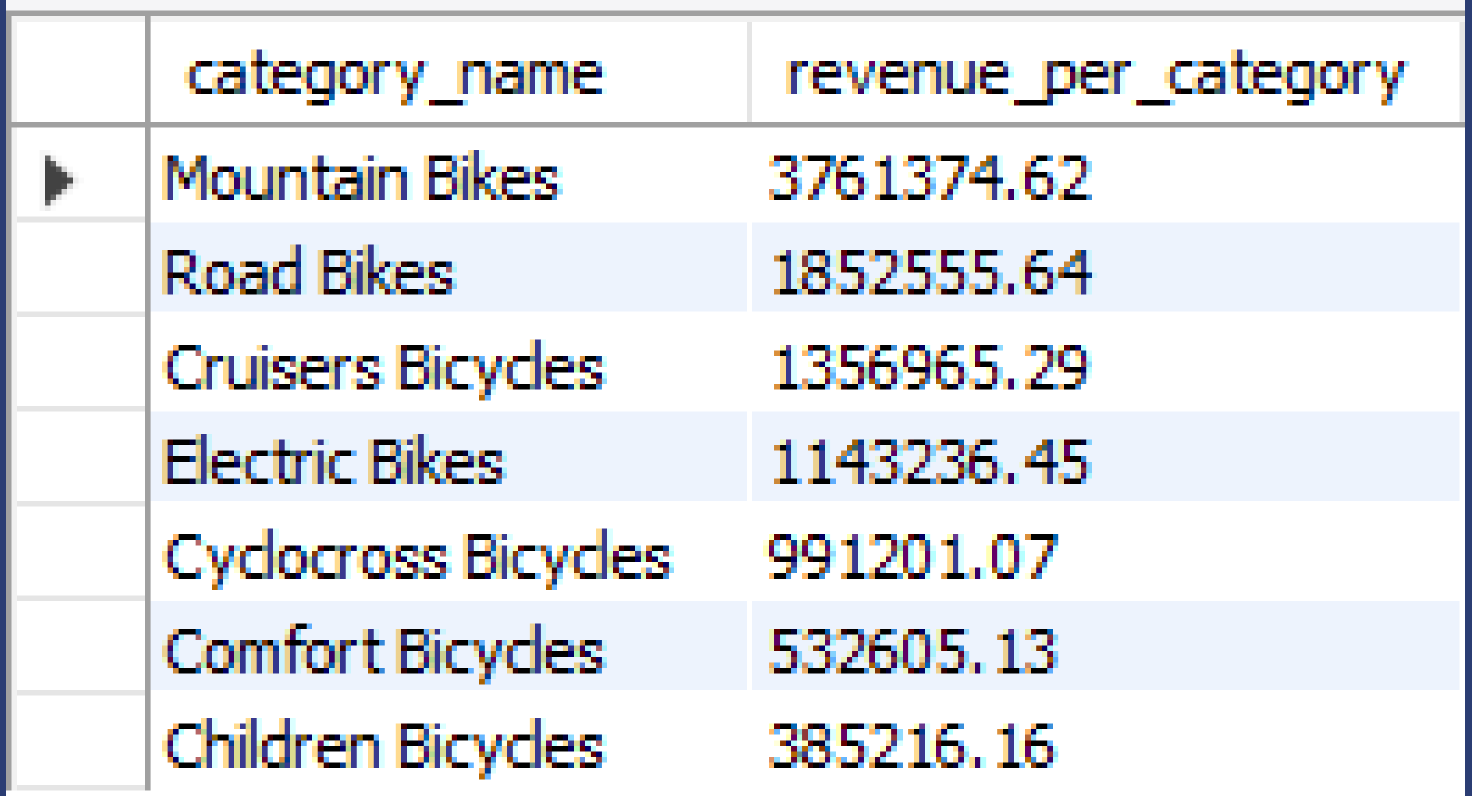


| customer_id | total_orders | total_quantity | total_spend | avg_order_value |
|---|---|---|---|---|
| 1 | 3 | 17 | 30645.87 | 2785.99 |
| 2 | 3 | 15 | 21653.85 | 2165.39 |
| 3 | 3 | 19 | 26249.81 | 2019.22 |
| 4 | 3 | 13 | 24198.88 | 2688.76 |
| 5 | 3 | 20 | 25151.82 | 2095.98 |
| 6 | 3 | 16 | 35857.86 | 3259.81 |
| 7 | 3 | 9 | 9205.95 | 1534.32 |
| 8 | 3 | 5 | 2603.95 | 867.98 |
| 9 | 3 | 21 | 37145.7 | 2476.38 |
| 10 | 3 | 18 | 37801.84 | 3436.53 |
| 11 | 3 | 7 | 4079.93 | 815.99 |
| 12 | 3 | 17 | 30578.83 | 2779.89 |
| 13 | 3 | 20 | 23819.82 | 1984.98 |
| 14 | 3 | 13 | 17307.89 | 1923.1 |
| 15 | 3 | 10 | 11392.9 | 1627.56 |

# ••• Total revenue generated by each product category •••



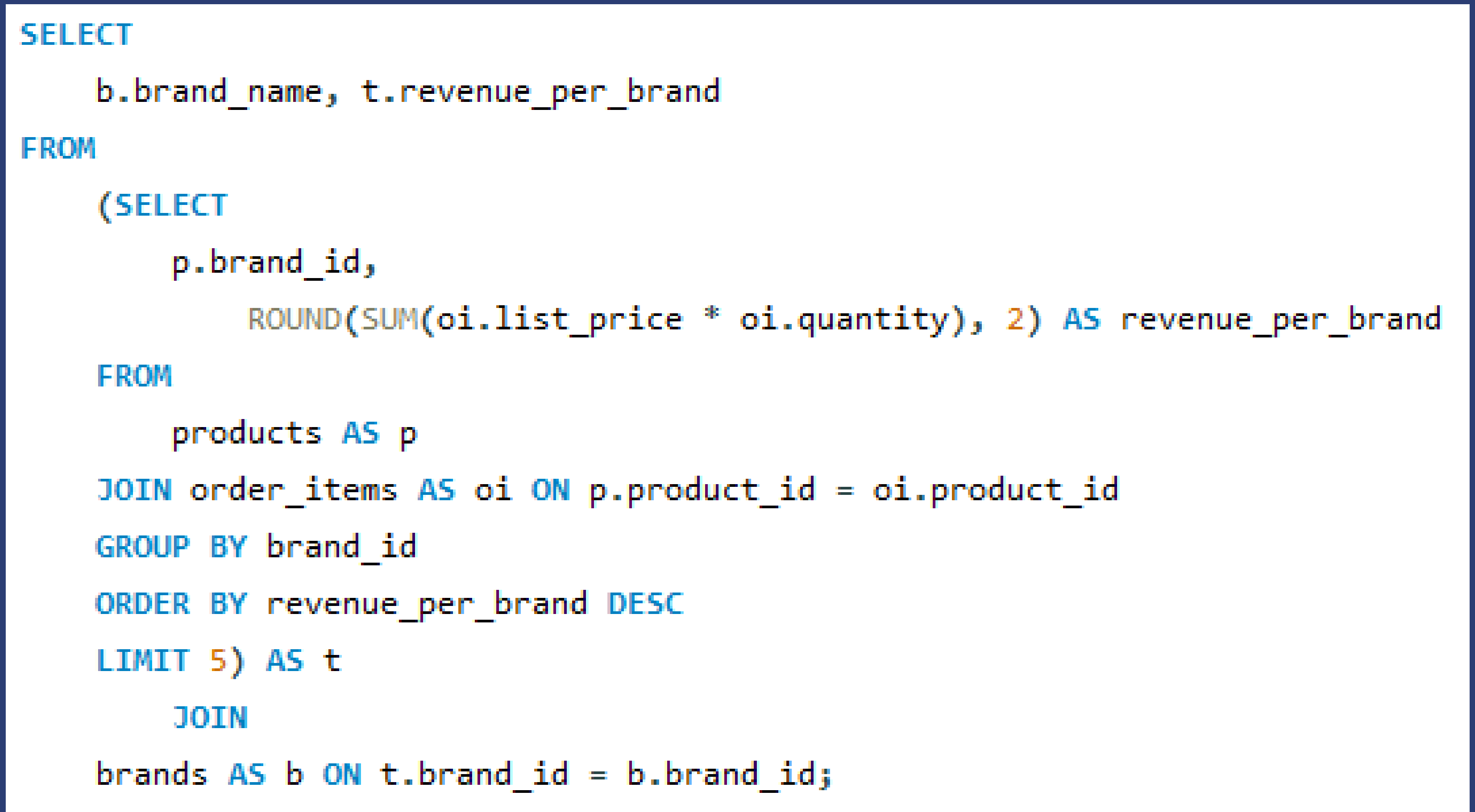
```
SELECT
    c.category_name,
    ROUND(SUM(oi.list_price * oi.quantity), 2) AS revenue_per_category
FROM
    products AS p
        JOIN
    order_items AS oi ON p.product_id = oi.product_id
    join categories as c on c.category_id=p.category_id
GROUP BY c.category_name
ORDER BY revenue_per_category desc;
```



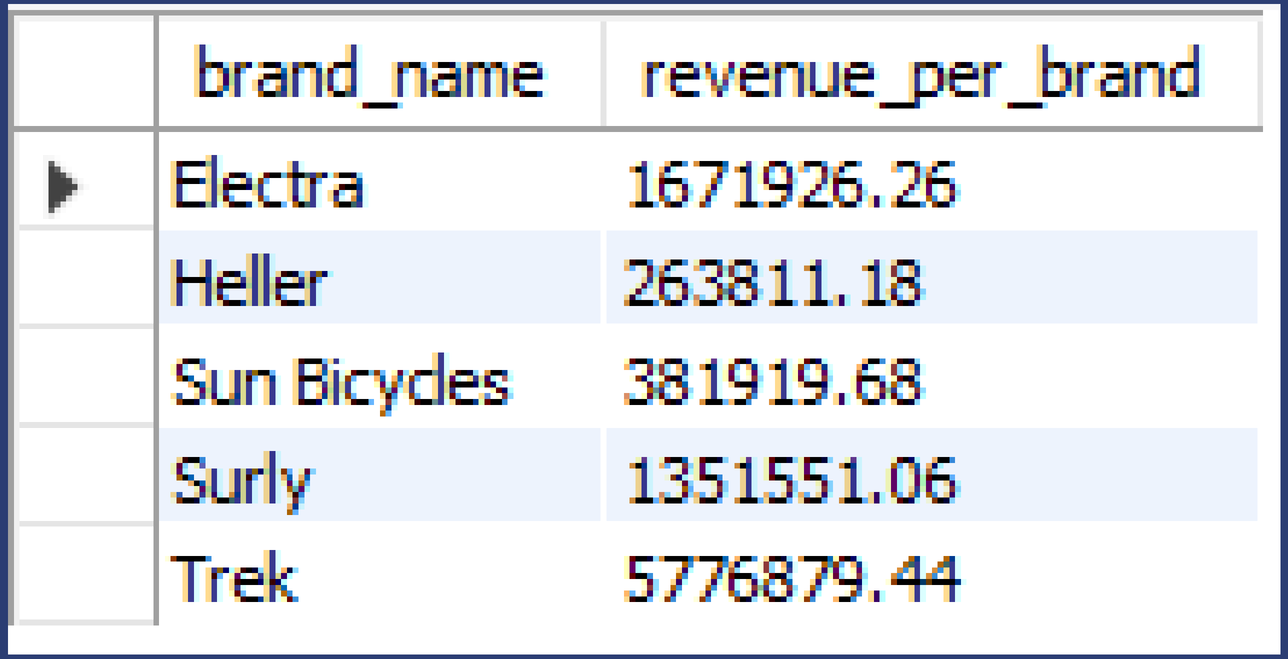
| | category_name | revenue_per_category |
|---|---|---|
| ▶ | Mountain Bikes | 3761374.62 |
| | Road Bikes | 1852555.64 |
| | Cruisers Bicycles | 1356965.29 |
| | Electric Bikes | 1143236.45 |
| | Cyclocross Bicycles | 991201.07 |
| | Comfort Bicycles | 532605.13 |
| | Children Bicycles | 385216.16 |

- Mountain Bikes has highest revenue compared to all other categories.
- So Management has to increase the stock expenditures in all the stores.
- Also has to concentrate on Children Bicycles sales and should increase its marketing strategies.

# ••• Total revenue generated by each product category •••



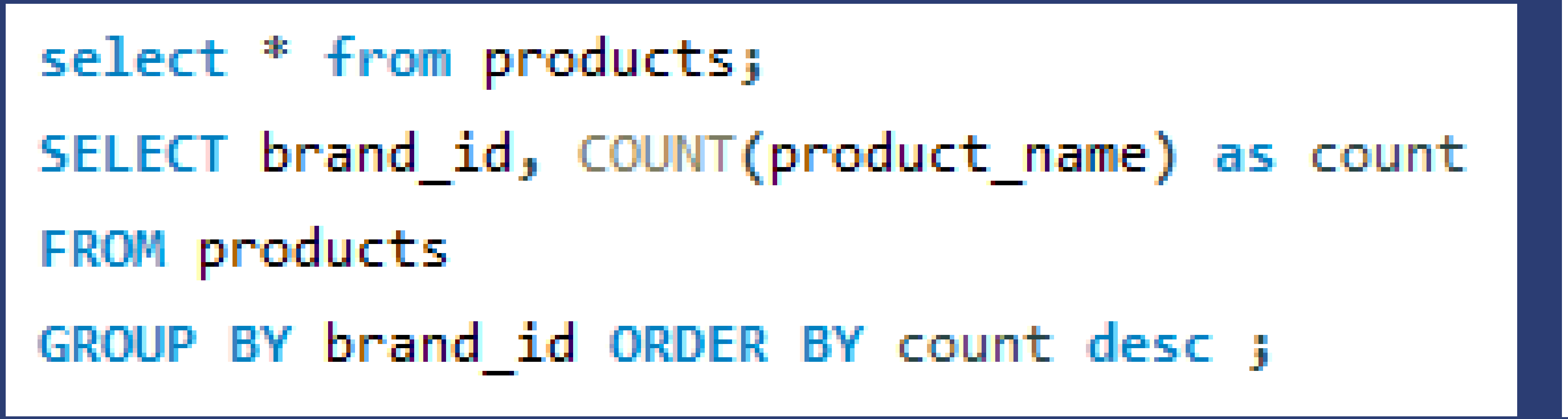
```sql
SELECT
    b.brand_name, t.revenue_per_brand
FROM
    (SELECT
        p.brand_id,
            ROUND(SUM(oi.list_price * oi.quantity), 2) AS revenue_per_brand
    FROM
        products AS p
    JOIN order_items AS oi ON p.product_id = oi.product_id
    GROUP BY brand_id
    ORDER BY revenue_per_brand DESC
    LIMIT 5) AS t
        JOIN
    brands AS b ON t.brand_id = b.brand_id;
```



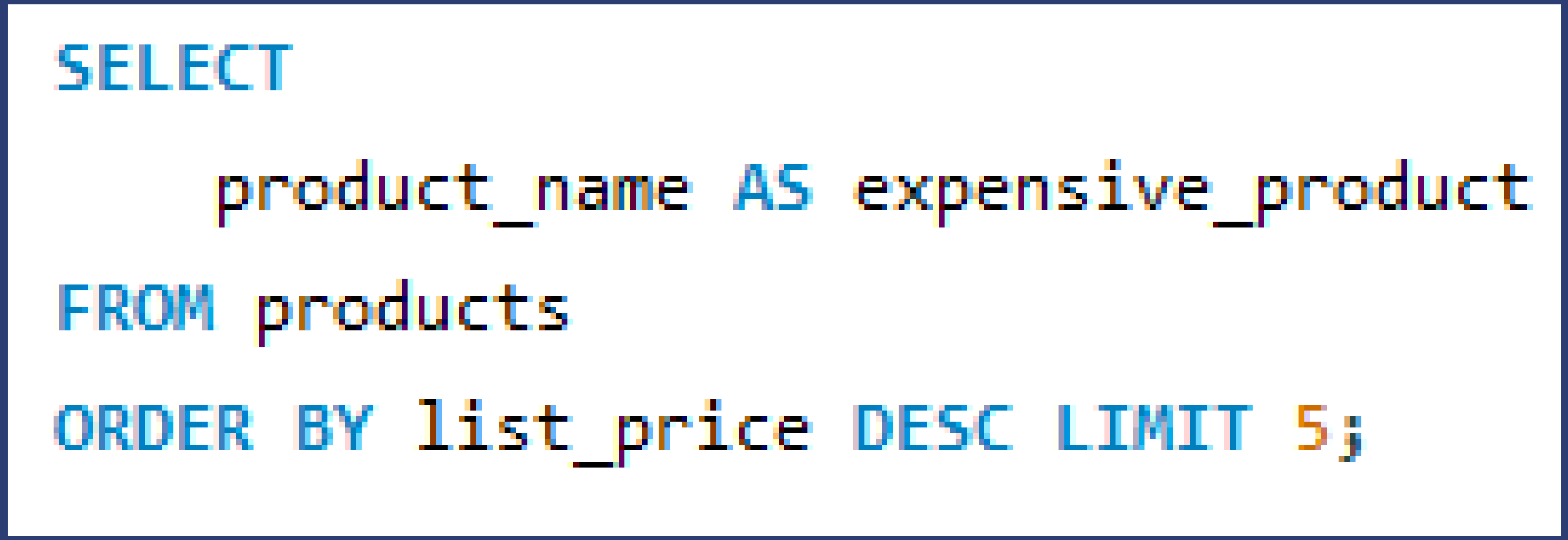
|  | brand_name | revenue_per_brand |
|---|---|---|
| ▶ | Electra | 1671926.26 |
|  | Heller | 263811.18 |
|  | Sun Bicycles | 381919.68 |
|  | Surly | 1351551.06 |
|  | Trek | 5776879.44 |

- Trek brand has highest revenue among all other brands.
- So Management has to increase the stock expenditures in all the stores.
- Also has to concentrate on Heller and Sun Bicycles brands to increase the overall revenue

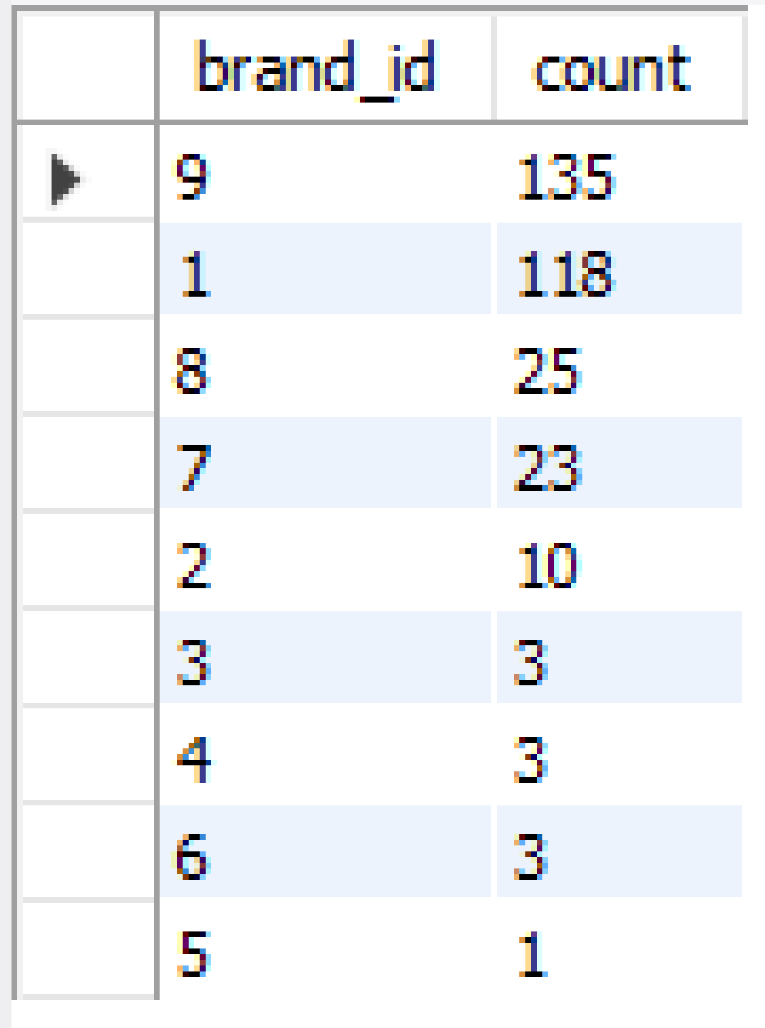
# ... Count of products under each Brands and Expensive Products ...



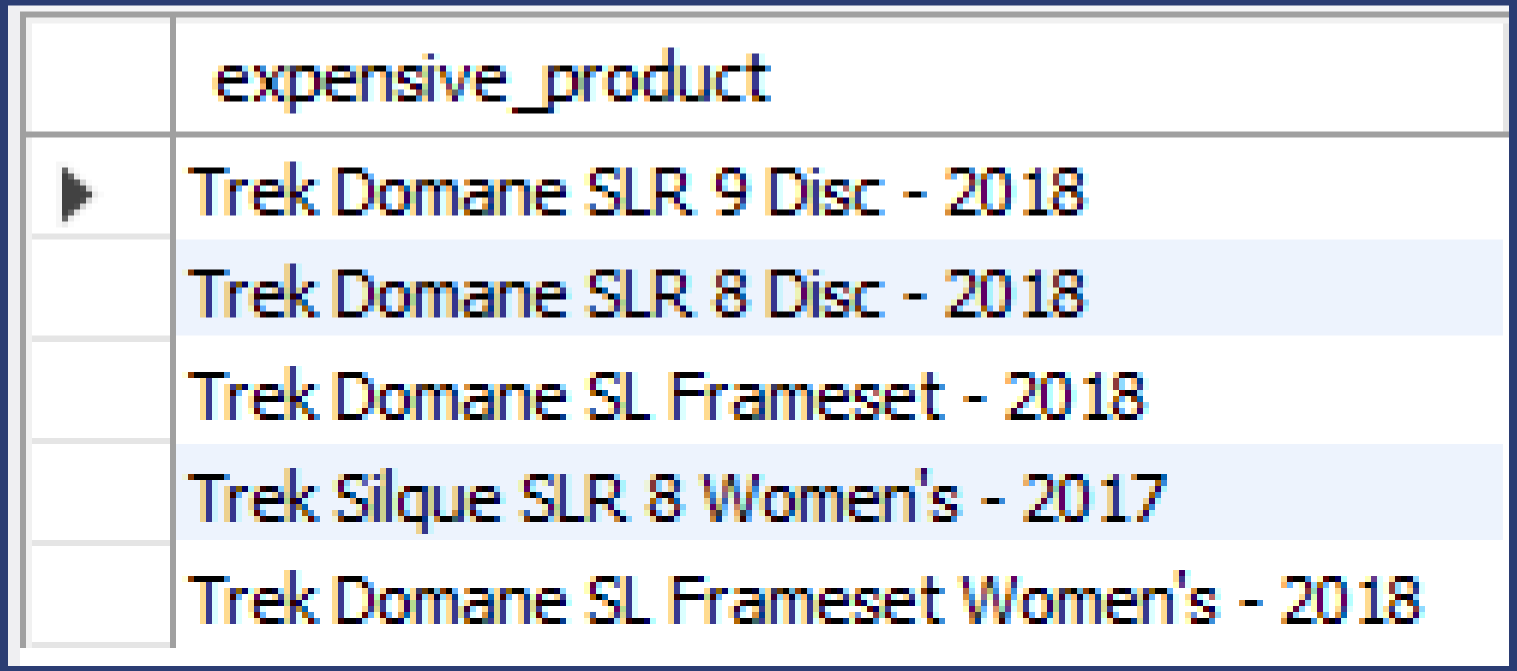
```
select * from products;
SELECT brand_id, COUNT(product_name) as count
FROM products
GROUP BY brand_id ORDER BY count desc ;
```



```
SELECT
    product_name AS expensive_product
FROM products
ORDER BY list_price DESC LIMIT 5;
```

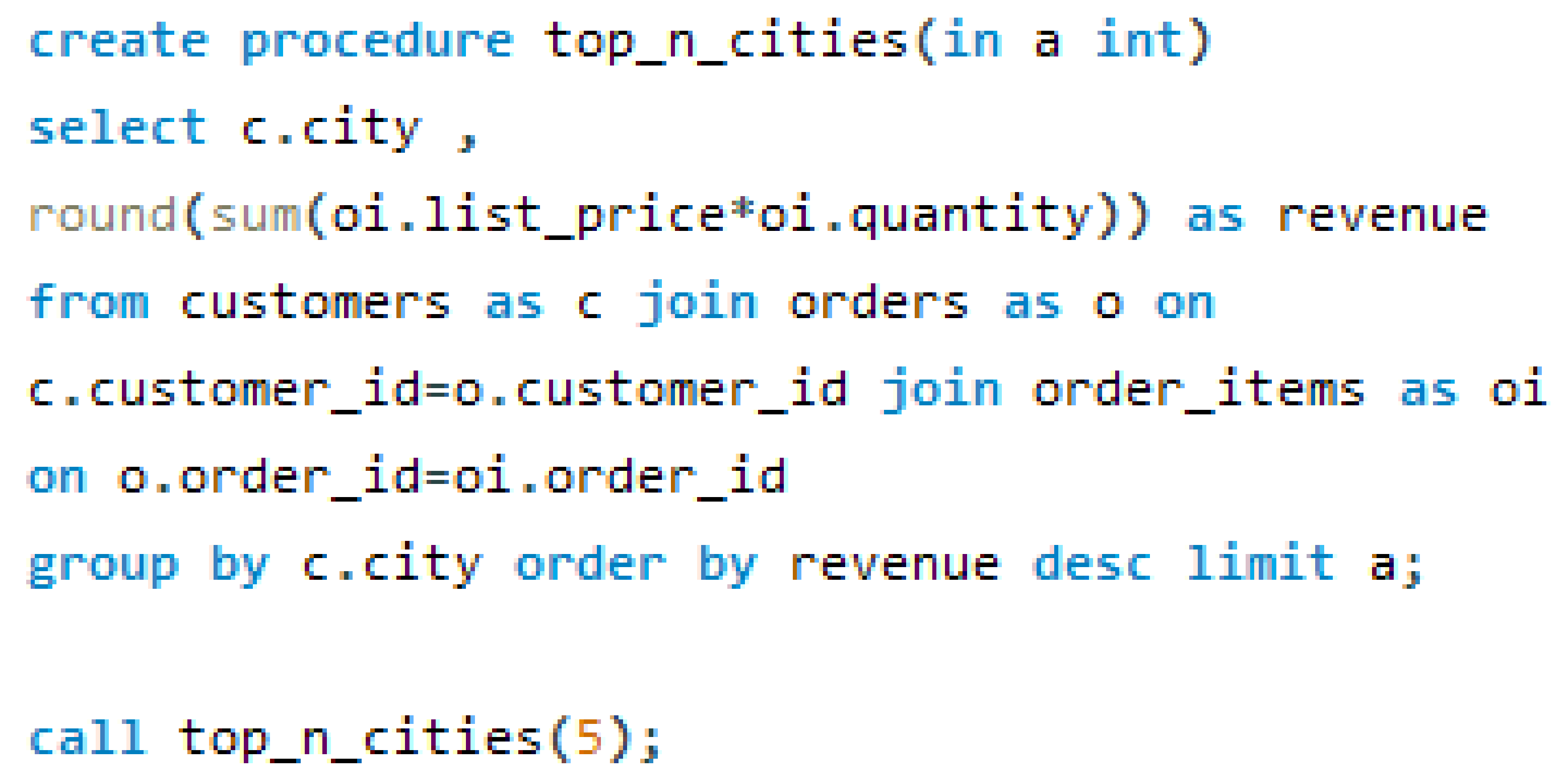


| brand_id | count |
|---|---|
| 9 | 135 |
| 1 | 118 |
| 8 | 25 |
| 7 | 23 |
| 2 | 10 |
| 3 | 3 |
| 4 | 3 |
| 6 | 3 |
| 5 | 1 |

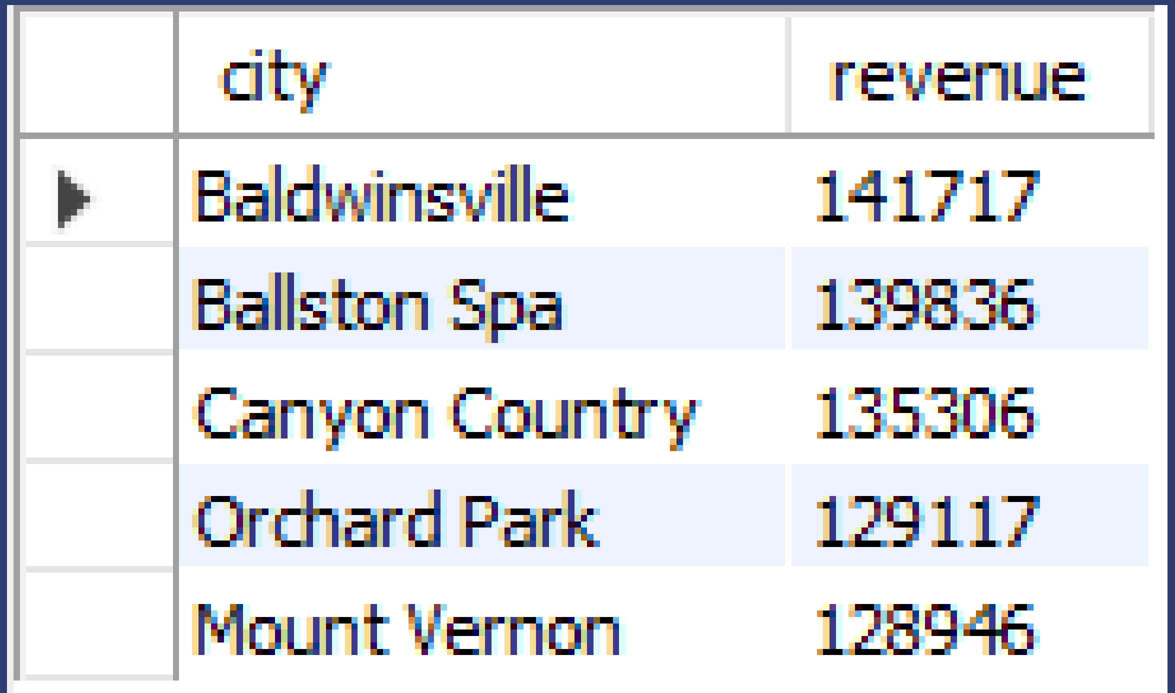


| expensive_product |
|---|
| Trek Domane SLR 9 Disc - 2018 |
| Trek Domane SLR 8 Disc - 2018 |
| Trek Domane SL Frameset - 2018 |
| Trek Silque SLR 8 Women's - 2017 |
| Trek Domane SL Frameset Women's - 2018 |

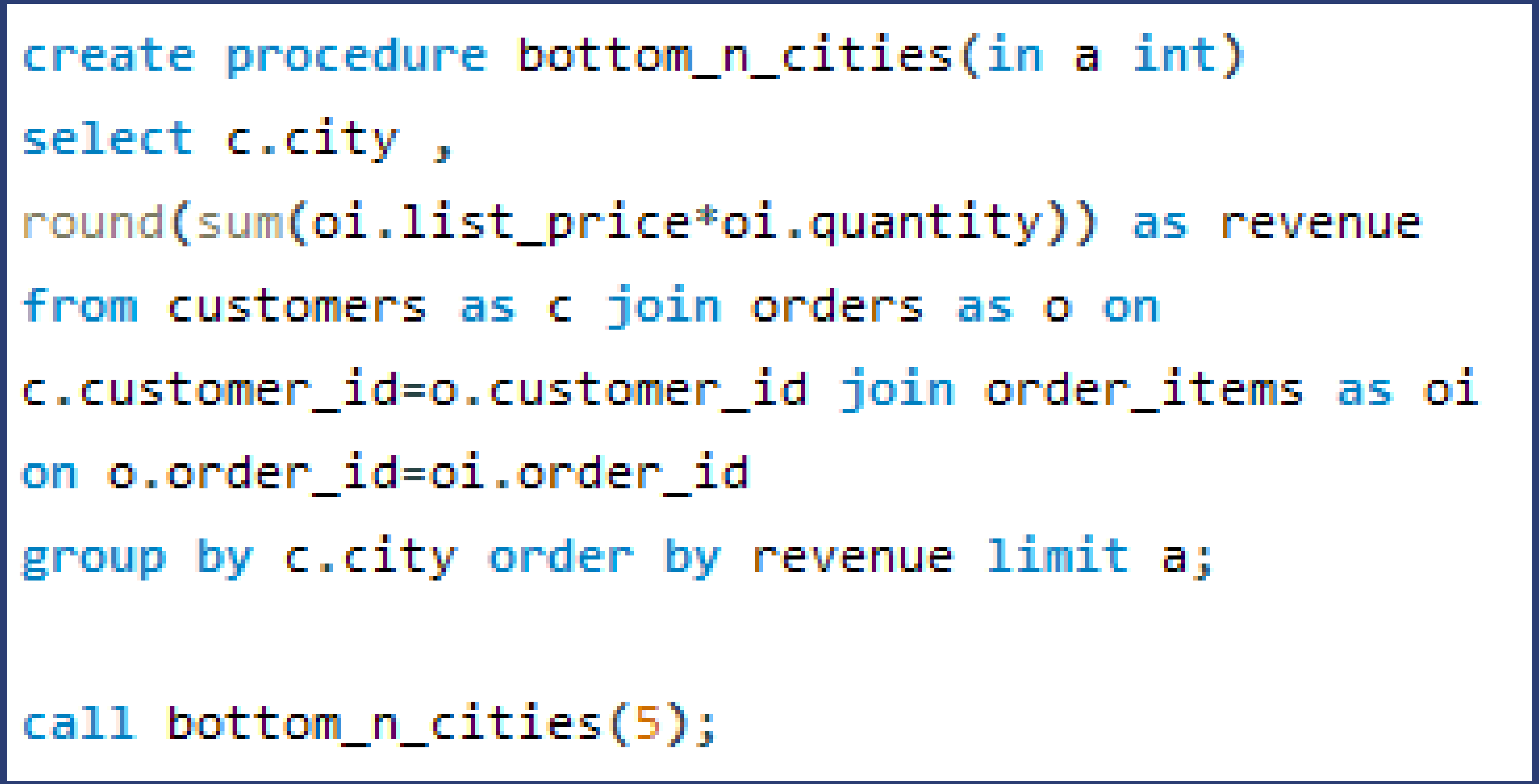
# Top and bottom 5 cities based on total revenue



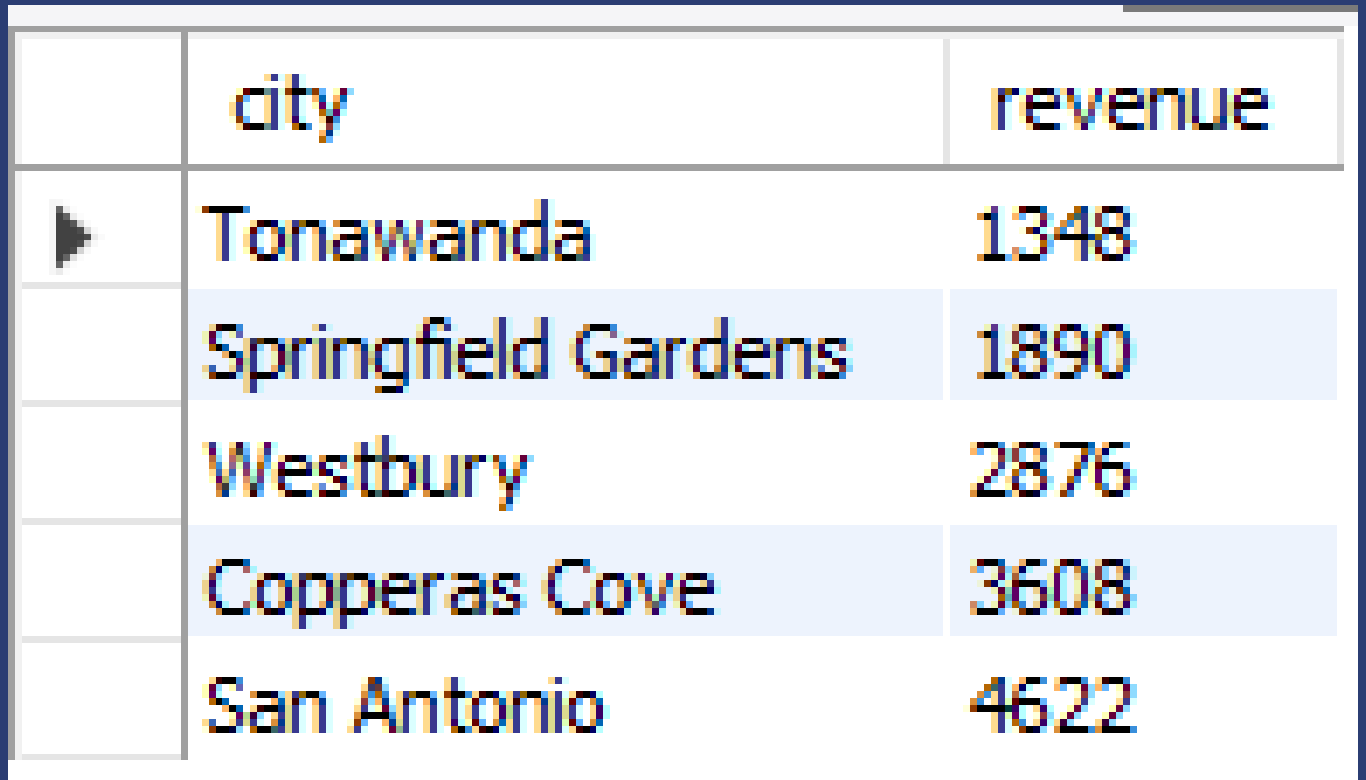
```
create procedure top_n_cities(in a int)
select c.city ,
round(sum(oi.list_price*oi.quantity)) as revenue
from customers as c join orders as o on
c.customer_id=o.customer_id join order_items as oi
on o.order_id=oi.order_id
group by c.city order by revenue desc limit a;

call top_n_cities(5);
```



| city | revenue |
|---|---|
| Baldwinsville | 141717 |
| Ballston Spa | 139836 |
| Canyon Country | 135306 |
| Orchard Park | 129117 |
| Mount Vernon | 128946 |



```
create procedure bottom_n_cities(in a int)
select c.city ,
round(sum(oi.list_price*oi.quantity)) as revenue
from customers as c join orders as o on
c.customer_id=o.customer_id join order_items as oi
on o.order_id=oi.order_id
group by c.city order by revenue limit a;

call bottom_n_cities(5);
```



| city | revenue |
|---|---|
| Tonawanda | 1348 |
| Springfield Gardens | 1890 |
| Westbury | 2876 |
| Copperas Cove | 3608 |
| San Antonio | 4622 |

# ••• RECOMMENDATIONS

- Analyze seasonal spikes to launch promotions and discounts during peak periods to increase the YoY revenues.

- Maintain optimal stock levels for high-demand products like *trek slash and trek fuels* to avoid losing the promising customers.

- Top performers should be acknowledged and given incentives or bonuses and organize workshops with top performers to upskill other sales employees.

- Also analyze the customers demography and concentrate on the high revenue generated cities like *Baldwinsville*

# THANK YOU