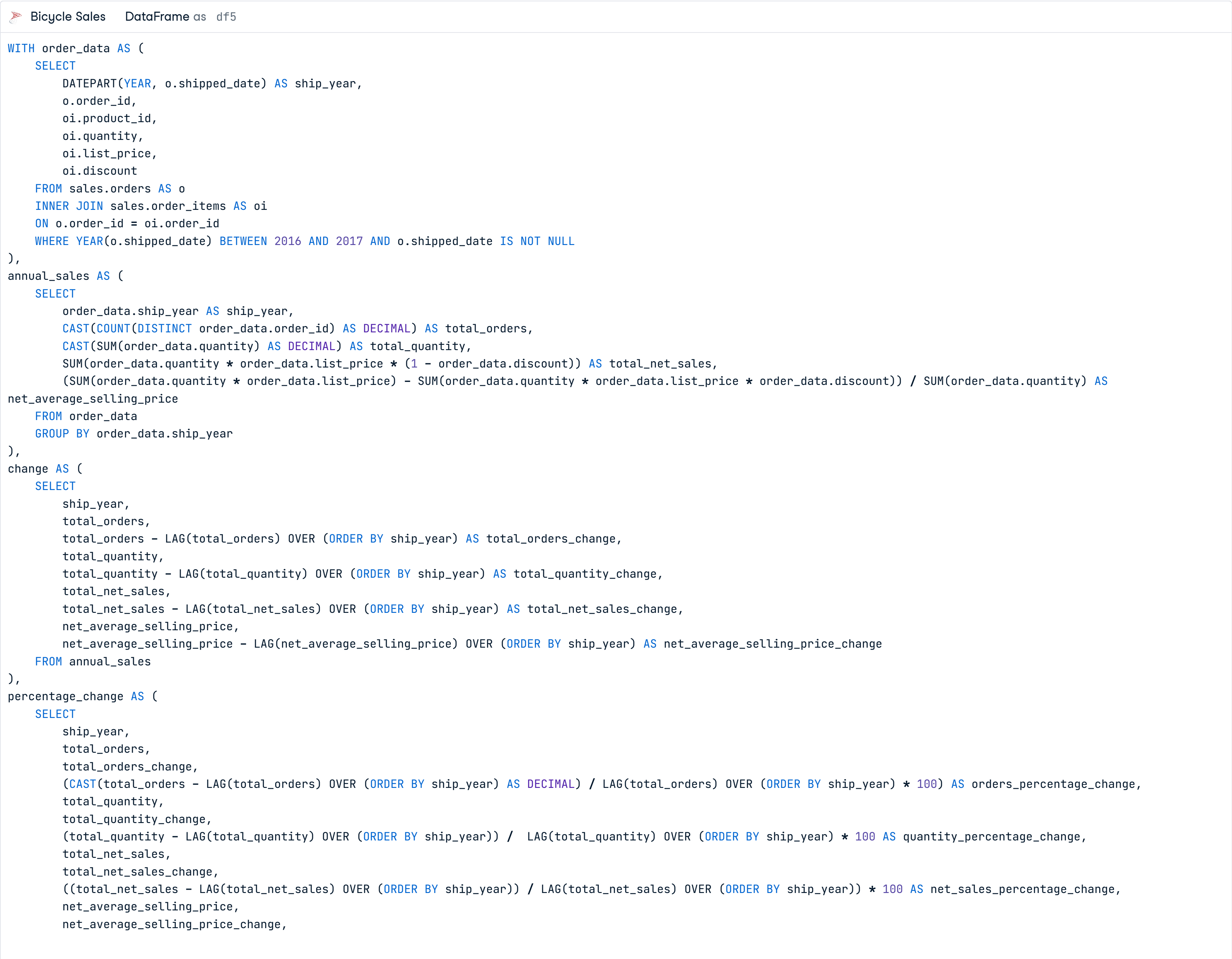
ANNUAL PERFORMANCE YOY



Bicycle Sales DataFrame as df5

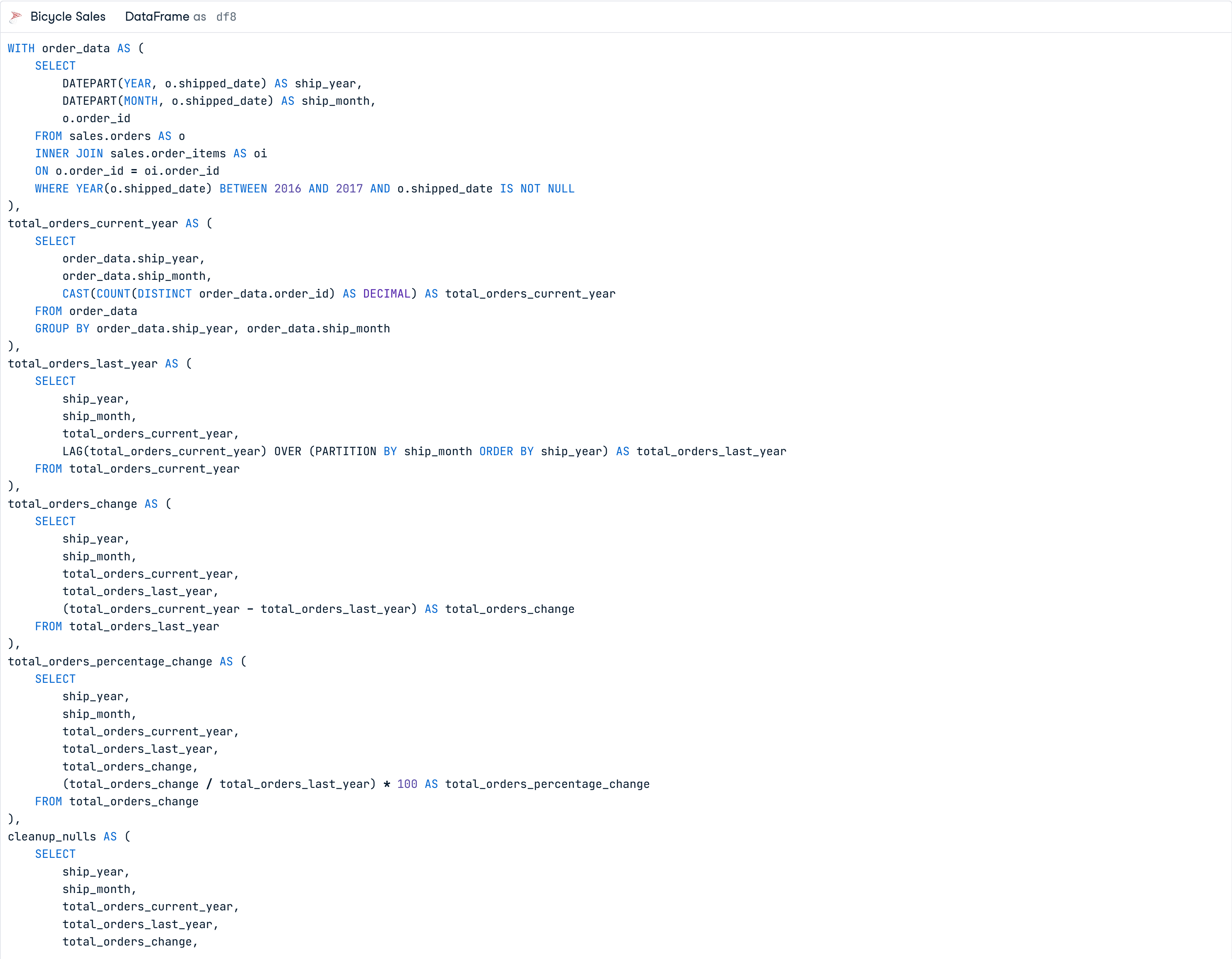
```
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        o.order_id,
        oi.product_id,
        oi.quantity,
        oi.list_price,
        oi.discount
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
annual_sales AS (
    SELECT
        order_data.ship_year AS ship_year,
        CAST(COUNT(DISTINCT order_data.order_id) AS DECIMAL) AS total_orders,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales,
        (SUM(order_data.quantity * order_data.list_price) - SUM(order_data.quantity * order_data.list_price * order_data.discount)) / SUM(order_data.quantity) AS
net_average_selling_price
    FROM order_data
    GROUP BY order_data.ship_year
),
change AS (
    SELECT
        ship_year,
        total_orders,
        total_orders - LAG(total_orders) OVER (ORDER BY ship_year) AS total_orders_change,
        total_quantity,
        total_quantity - LAG(total_quantity) OVER (ORDER BY ship_year) AS total_quantity_change,
        total_net_sales,
        total_net_sales - LAG(total_net_sales) OVER (ORDER BY ship_year) AS total_net_sales_change,
        net_average_selling_price,
        net_average_selling_price - LAG(net_average_selling_price) OVER (ORDER BY ship_year) AS net_average_selling_price_change
    FROM annual_sales
),
percentage_change AS (
    SELECT
        ship_year,
        total_orders,
        total_orders_change,
        (CAST(total_orders - LAG(total_orders) OVER (ORDER BY ship_year) AS DECIMAL) / LAG(total_orders) OVER (ORDER BY ship_year) * 100) AS orders_percentage_change,
        total_quantity,
        total_quantity_change,
        (total_quantity - LAG(total_quantity) OVER (ORDER BY ship_year)) /  LAG(total_quantity) OVER (ORDER BY ship_year) * 100 AS quantity_percentage_change,
        total_net_sales,
        total_net_sales_change,
        ((total_net_sales - LAG(total_net_sales) OVER (ORDER BY ship_year)) / LAG(total_net_sales) OVER (ORDER BY ship_year)) * 100 AS net_sales_percentage_change,
        net_average_selling_price,
        net_average_selling_price_change,
        ((net_average_selling_price - LAG(net_average_selling_price) OVER (ORDER BY ship_year)) / LAG(net_average_selling_price) OVER (ORDER BY ship_year)) * 100 AS
net_asp_percentage_change
    FROM change
)
SELECT
        ship_year,
        total_orders,
        total_orders_change,
        ROUND(orders_percentage_change,2) AS orders_percentage_change,
        total_quantity,
        total_quantity_change,
        ROUND(quantity_percentage_change,2) AS quantity_percentage_change,
        total_net_sales,
        total_net_sales_change,
        ROUND(net_sales_percentage_change,2) AS net_sales_percentage_change,
        net_average_selling_price,
        net_average_selling_price_change,
        ROUND(net_asp_percentage_change,2) AS net_asp_percentage_change
FROM percentage_change
```

	s.	total...	total_orders_cha...	orders_percentage_change	total_qu...	total_quantity_chan...	quantity_percentage_change	total_net_...
0	2016	620			2602			23728
1	2017	669	49	7.9	3022	420	16.14	33382

Rows: 2

Expand

MONTHLY ORDER PERFORMANCE YOY



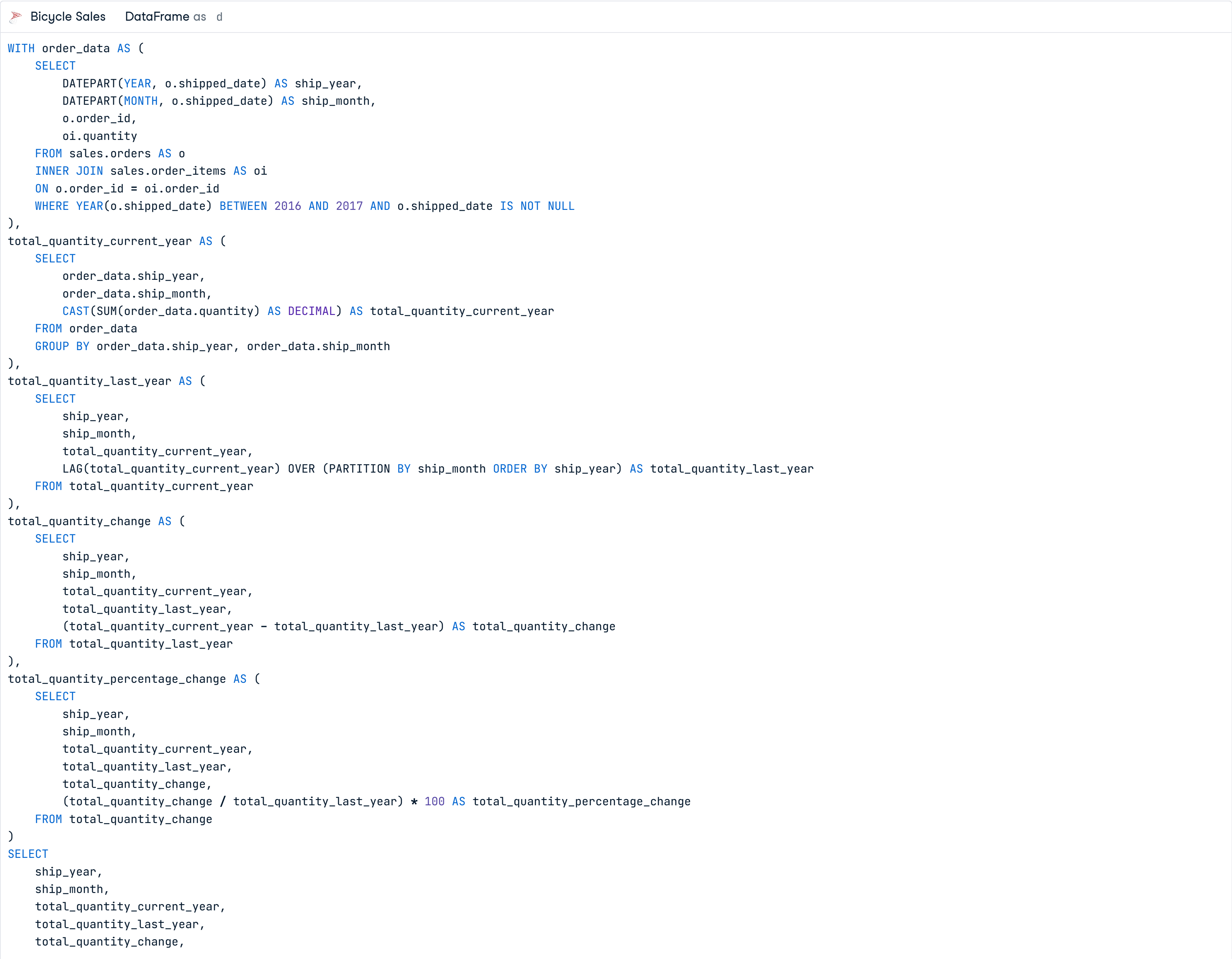
```
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        DATEPART(MONTH, o.shipped_date) AS ship_month,
        o.order_id
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
total_orders_current_year AS (
    SELECT
        order_data.ship_year,
        order_data.ship_month,
        CAST(COUNT(DISTINCT order_data.order_id) AS DECIMAL) AS total_orders_current_year
    FROM order_data
    GROUP BY order_data.ship_year, order_data.ship_month
),
total_orders_last_year AS (
    SELECT
        ship_year,
        ship_month,
        total_orders_current_year,
        LAG(total_orders_current_year) OVER (PARTITION BY ship_month ORDER BY ship_year) AS total_orders_last_year
    FROM total_orders_current_year
),
total_orders_change AS (
    SELECT
        ship_year,
        ship_month,
        total_orders_current_year,
        total_orders_last_year,
        (total_orders_current_year - total_orders_last_year) AS total_orders_change
    FROM total_orders_last_year
),
total_orders_percentage_change AS (
    SELECT
        ship_year,
        ship_month,
        total_orders_current_year,
        total_orders_last_year,
        total_orders_change,
        (total_orders_change / total_orders_last_year) * 100 AS total_orders_percentage_change
    FROM total_orders_change
),
cleanup_nulls AS (
    SELECT
        ship_year,
        ship_month,
        total_orders_current_year,
        total_orders_last_year,
        total_orders_change,
        CASE
            WHEN total_orders_percentage_change IS NULL THEN 0
            ELSE total_orders_percentage_change
            END AS total_orders_percentage_change
    FROM total_orders_percentage_change
)
SELECT
    ship_year,
    ship_month,
    total_orders_current_year,
    total_orders_last_year,
    total_orders_change,
    ROUND(total_orders_percentage_change,2) AS total_orders_percentage_change
FROM cleanup_nulls
WHERE ship_year = 2017;
```

	s.	s...	total_orders_current_year	total_orders_last_year	total_orders_cha...	total_orders_percentage_change
0	2017	1	47	47	0	0
1	2017	2	50	48	2	4.17
2	2017	3	69	54	15	27.78
3	2017	4	57	40	17	42.5
4	2017	5	54	52	2	3.85
5	2017	6	64	42	22	52.38
6	2017	7	45	50	-5	-10
7	2017	8	67	55	12	21.82
8	2017	9	49	66	-17	-25.76
9	2017	10	66	67	-1	-1.49
10	2017	11	54	45	9	20
11	2017	12	47	54	-7	-12.96

Rows: 12

Expand

MONTHLY QUANTITY PERFORMANCE YOY



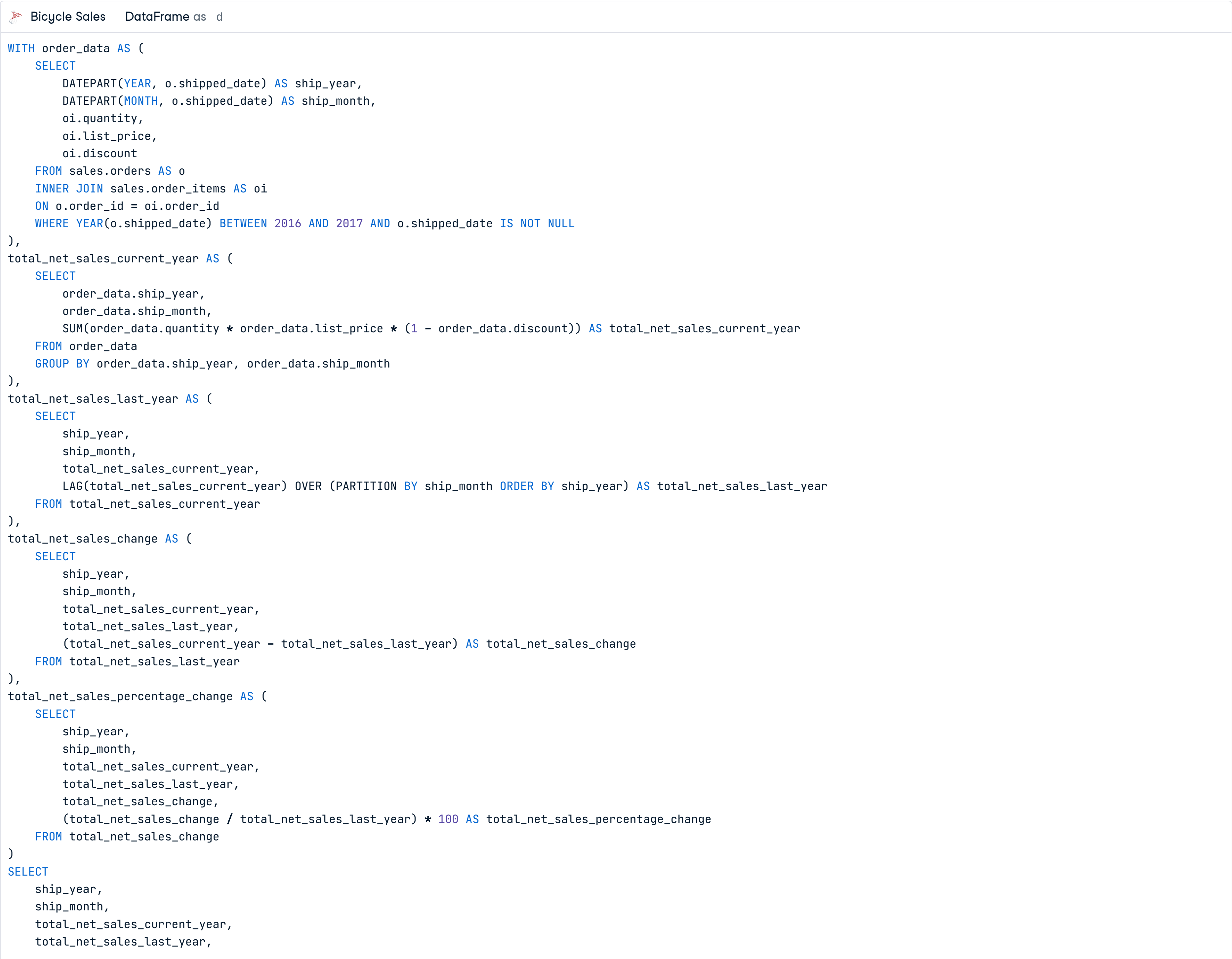
```sql
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        DATEPART(MONTH, o.shipped_date) AS ship_month,
        o.order_id,
        oi.quantity
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
total_quantity_current_year AS (
    SELECT
        order_data.ship_year,
        order_data.ship_month,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity_current_year
    FROM order_data
    GROUP BY order_data.ship_year, order_data.ship_month
),
total_quantity_last_year AS (
    SELECT
        ship_year,
        ship_month,
        total_quantity_current_year,
        LAG(total_quantity_current_year) OVER (PARTITION BY ship_month ORDER BY ship_year) AS total_quantity_last_year
    FROM total_quantity_current_year
),
total_quantity_change AS (
    SELECT
        ship_year,
        ship_month,
        total_quantity_current_year,
        total_quantity_last_year,
        (total_quantity_current_year - total_quantity_last_year) AS total_quantity_change
    FROM total_quantity_last_year
),
total_quantity_percentage_change AS (
    SELECT
        ship_year,
        ship_month,
        total_quantity_current_year,
        total_quantity_last_year,
        total_quantity_change,
        (total_quantity_change / total_quantity_last_year) * 100 AS total_quantity_percentage_change
    FROM total_quantity_change
)
SELECT
    ship_year,
    ship_month,
    total_quantity_current_year,
    total_quantity_last_year,
    total_quantity_change,
    ROUND(total_quantity_percentage_change,2) AS total_quantity_percentage_change
FROM total_quantity_percentage_change
WHERE ship_year = 2017;
```

	s.	s...	total_quantity_current_year	total_quantity_last_year	total_quantity_chan...	total_quantity_percentage_change
0	2017	1	217	208	9	4.33
1	2017	2	230	214	16	7.48
2	2017	3	300	217	83	38.25
3	2017	4	258	162	96	59.26
4	2017	5	233	234	-1	-0.43
5	2017	6	295	182	113	62.09
6	2017	7	219	211	8	3.79
7	2017	8	294	214	80	37.38
8	2017	9	219	284	-65	-22.89
9	2017	10	294	261	33	12.64
10	2017	11	246	187	59	31.55
11	2017	12	217	228	-11	-4.82

Rows: 12

Expand

MONTHLY SALES PERFORMANCE YOY



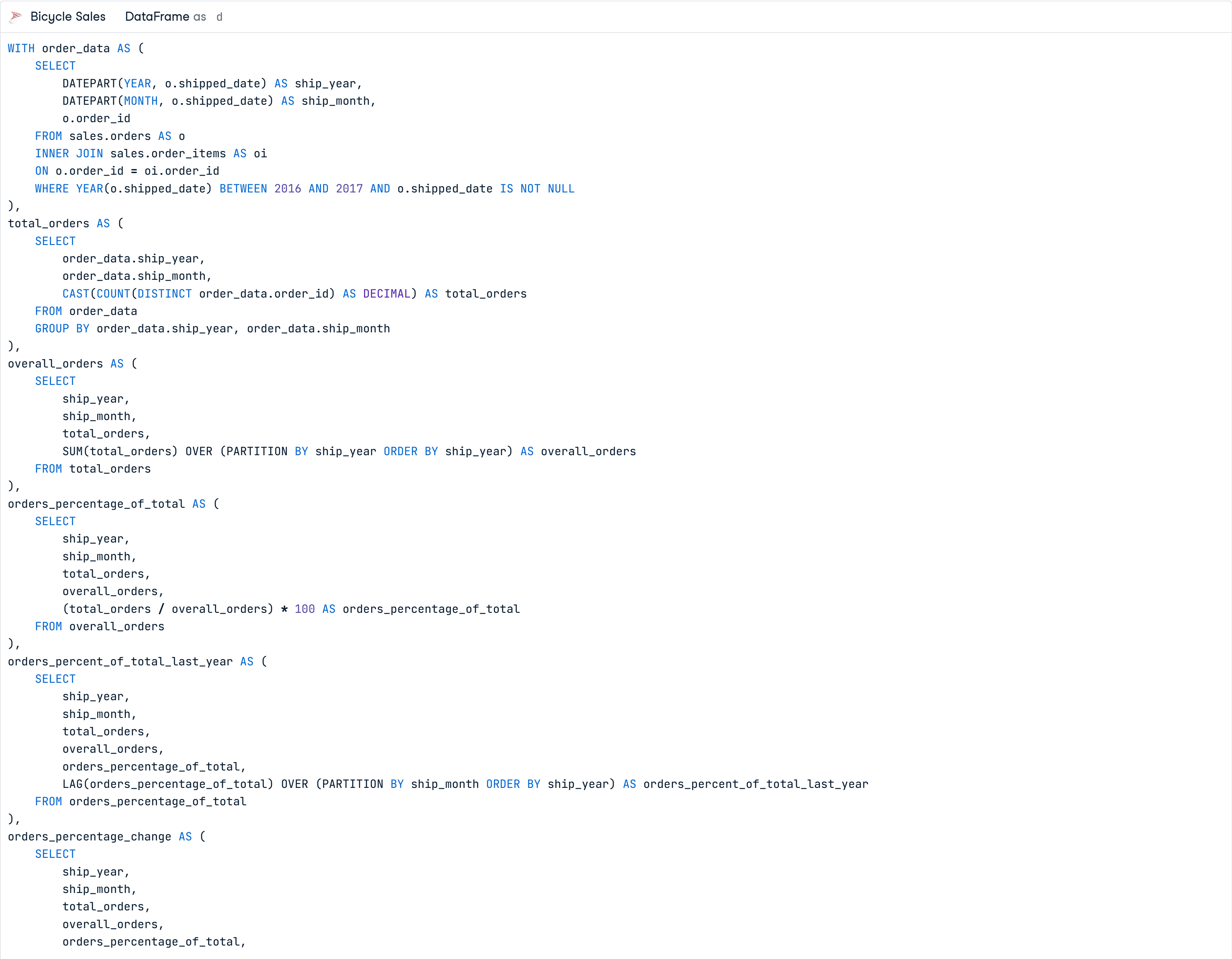
```sql
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        DATEPART(MONTH, o.shipped_date) AS ship_month,
        oi.quantity,
        oi.list_price,
        oi.discount
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
total_net_sales_current_year AS (
    SELECT
        order_data.ship_year,
        order_data.ship_month,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales_current_year
    FROM order_data
    GROUP BY order_data.ship_year, order_data.ship_month
),
total_net_sales_last_year AS (
    SELECT
        ship_year,
        ship_month,
        total_net_sales_current_year,
        LAG(total_net_sales_current_year) OVER (PARTITION BY ship_month ORDER BY ship_year) AS total_net_sales_last_year
    FROM total_net_sales_current_year
),
total_net_sales_change AS (
    SELECT
        ship_year,
        ship_month,
        total_net_sales_current_year,
        total_net_sales_last_year,
        (total_net_sales_current_year - total_net_sales_last_year) AS total_net_sales_change
    FROM total_net_sales_last_year
),
total_net_sales_percentage_change AS (
    SELECT
        ship_year,
        ship_month,
        total_net_sales_current_year,
        total_net_sales_last_year,
        total_net_sales_change,
        (total_net_sales_change / total_net_sales_last_year) * 100 AS total_net_sales_percentage_change
    FROM total_net_sales_change
)
SELECT
    ship_year,
    ship_month,
    total_net_sales_current_year,
    total_net_sales_last_year,
    total_net_sales_change,
    ROUND(total_net_sales_percentage_change, 2) AS total_net_sales_percentage_change
FROM total_net_sales_percentage_change
WHERE ship_year = 2017;
```

	s.	s...	total_net_sales_current_year	total_net_sales_last_year	total_net_sales_change	total_net_sales_percentage_change
0	2017	1	261778.4433	205433.4404	56345.0029	27.43
1	2017	2	264885.8227	146103.8527	118781.97	81.3
2	2017	3	329165.0944	183252.9298	145912.1646	79.62
3	2017	4	236917.2864	153300.683	83616.6034	54.54
4	2017	5	254514.6103	216475.3955	38039.2148	17.57
5	2017	6	379841.6193	197656.4177	182185.2016	92.17
6	2017	7	199067.6154	198682.4528	385.1626	0.19
7	2017	8	293043.4756	190809.8832	102233.5924	53.58
8	2017	9	281670.424	278616.6699	3053.7541	1.1
9	2017	10	281470.8638	224444.8112	57026.0526	25.41
10	2017	11	295301.2675	180563.9341	114737.3334	63.54
11	2017	12	260593.402	197508.7259	63084.6761	31.94

Rows: 12

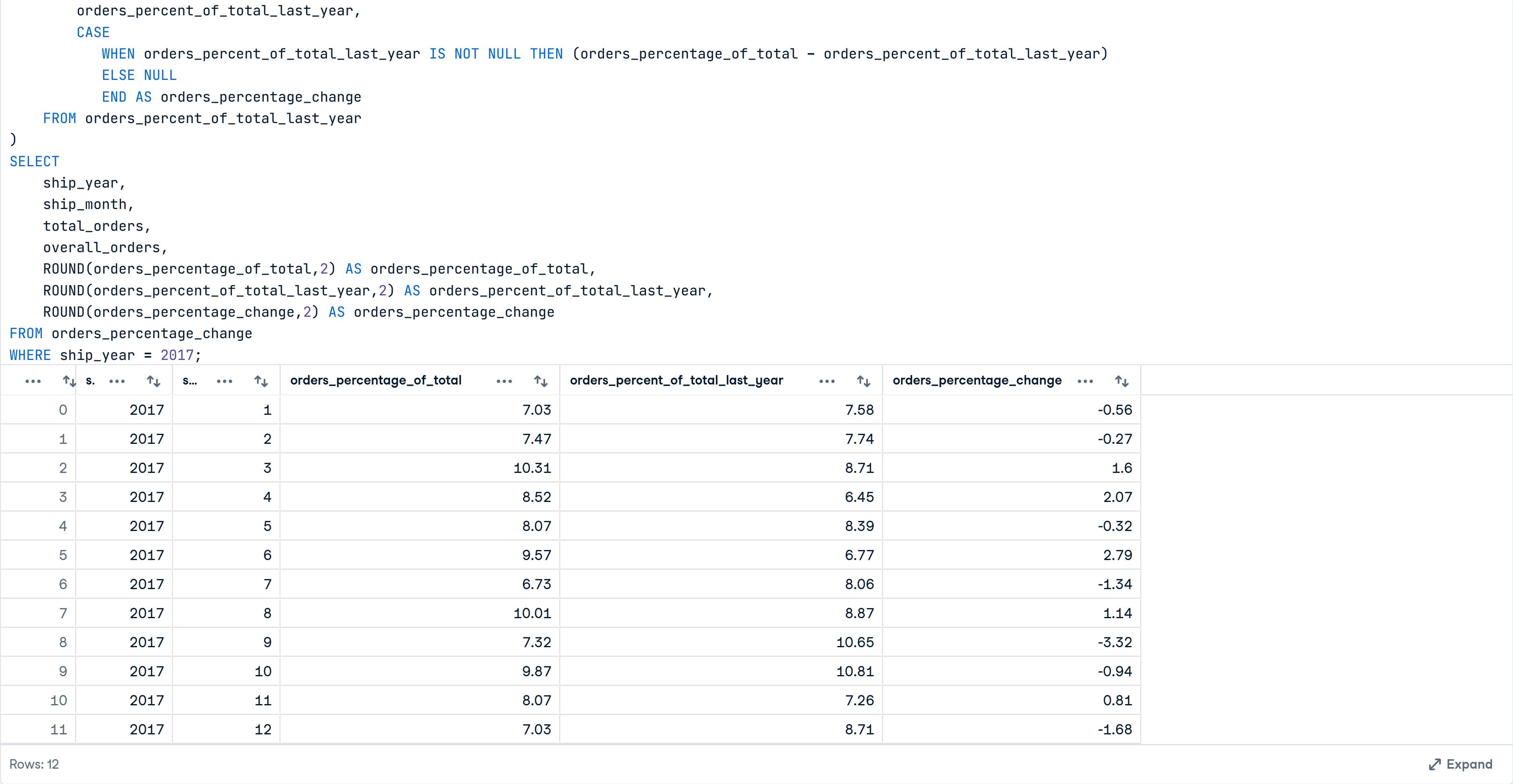
Expand

ORDERS SEASONALITY TRENDS



Bicycle Sales   DataFrame as d

```
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        DATEPART(MONTH, o.shipped_date) AS ship_month,
        o.order_id
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
total_orders AS (
    SELECT
        order_data.ship_year,
        order_data.ship_month,
        CAST(COUNT(DISTINCT order_data.order_id) AS DECIMAL) AS total_orders
    FROM order_data
    GROUP BY order_data.ship_year, order_data.ship_month
),
overall_orders AS (
    SELECT
        ship_year,
        ship_month,
        total_orders,
        SUM(total_orders) OVER (PARTITION BY ship_year ORDER BY ship_year) AS overall_orders
    FROM total_orders
),
orders_percentage_of_total AS (
    SELECT
        ship_year,
        ship_month,
        total_orders,
        overall_orders,
        (total_orders / overall_orders) * 100 AS orders_percentage_of_total
    FROM overall_orders
),
orders_percent_of_total_last_year AS (
    SELECT
        ship_year,
        ship_month,
        total_orders,
        overall_orders,
        orders_percentage_of_total,
        LAG(orders_percentage_of_total) OVER (PARTITION BY ship_month ORDER BY ship_year) AS orders_percent_of_total_last_year
    FROM orders_percentage_of_total
),
orders_percentage_change AS (
    SELECT
        ship_year,
        ship_month,
        total_orders,
        overall_orders,
        orders_percentage_of_total,
```

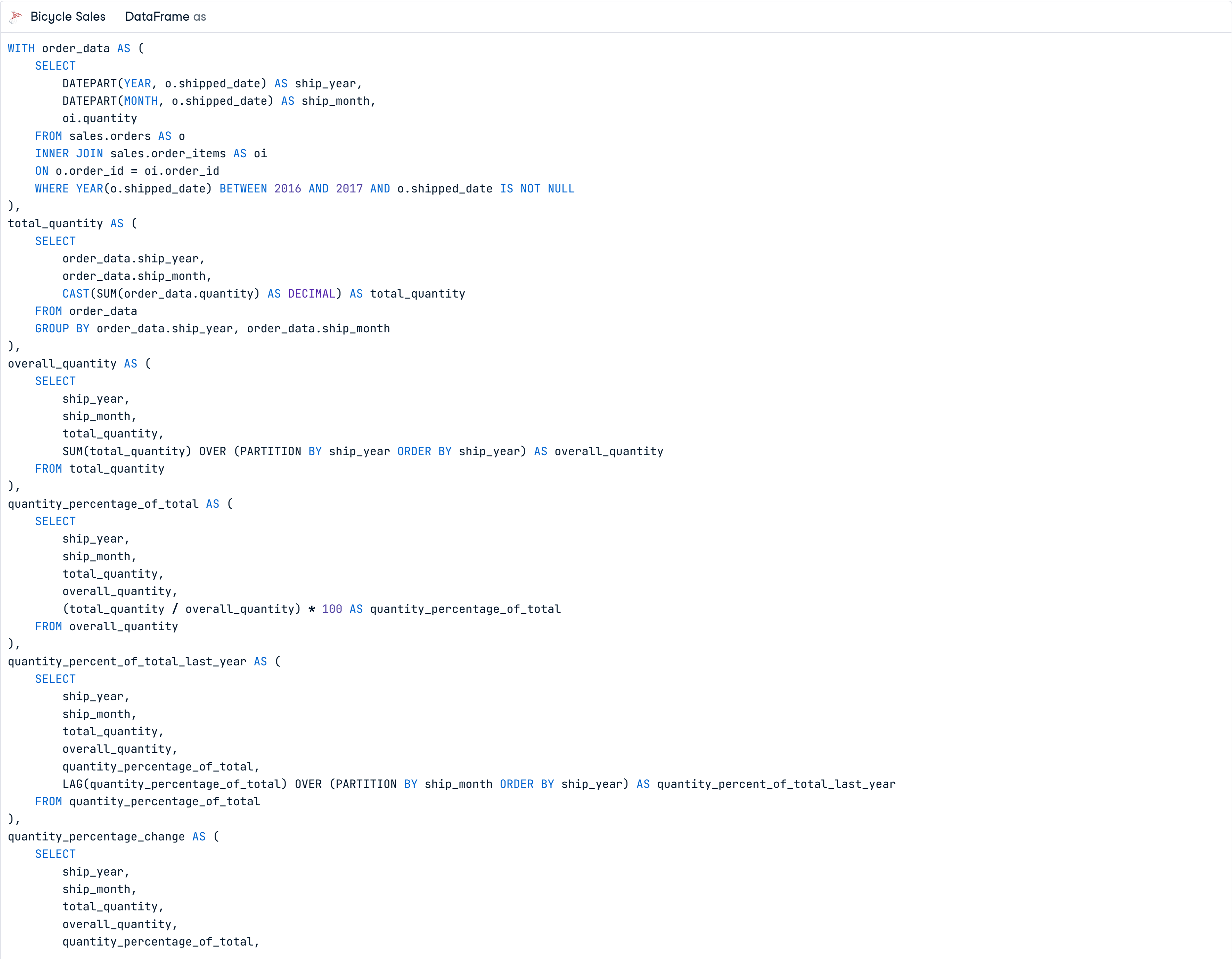


```
        orders_percent_of_total_last_year,
        CASE
           WHEN orders_percent_of_total_last_year IS NOT NULL THEN (orders_percentage_of_total - orders_percent_of_total_last_year)
           ELSE NULL
           END AS orders_percentage_change
    FROM orders_percent_of_total_last_year
)
SELECT
    ship_year,
    ship_month,
    total_orders,
    overall_orders,
    ROUND(orders_percentage_of_total,2) AS orders_percentage_of_total,
    ROUND(orders_percent_of_total_last_year,2) AS orders_percent_of_total_last_year,
    ROUND(orders_percentage_change,2) AS orders_percentage_change
FROM orders_percentage_change
WHERE ship_year = 2017;
```

	s.	s...	orders_percentage_of_total	orders_percent_of_total_last_year	orders_percentage_change
0	2017	1	7.03	7.58	-0.56
1	2017	2	7.47	7.74	-0.27
2	2017	3	10.31	8.71	1.6
3	2017	4	8.52	6.45	2.07
4	2017	5	8.07	8.39	-0.32
5	2017	6	9.57	6.77	2.79
6	2017	7	6.73	8.06	-1.34
7	2017	8	10.01	8.87	1.14
8	2017	9	7.32	10.65	-3.32
9	2017	10	9.87	10.81	-0.94
10	2017	11	8.07	7.26	0.81
11	2017	12	7.03	8.71	-1.68

Rows: 12

QUANTITY SEASONAILITY TRENDS



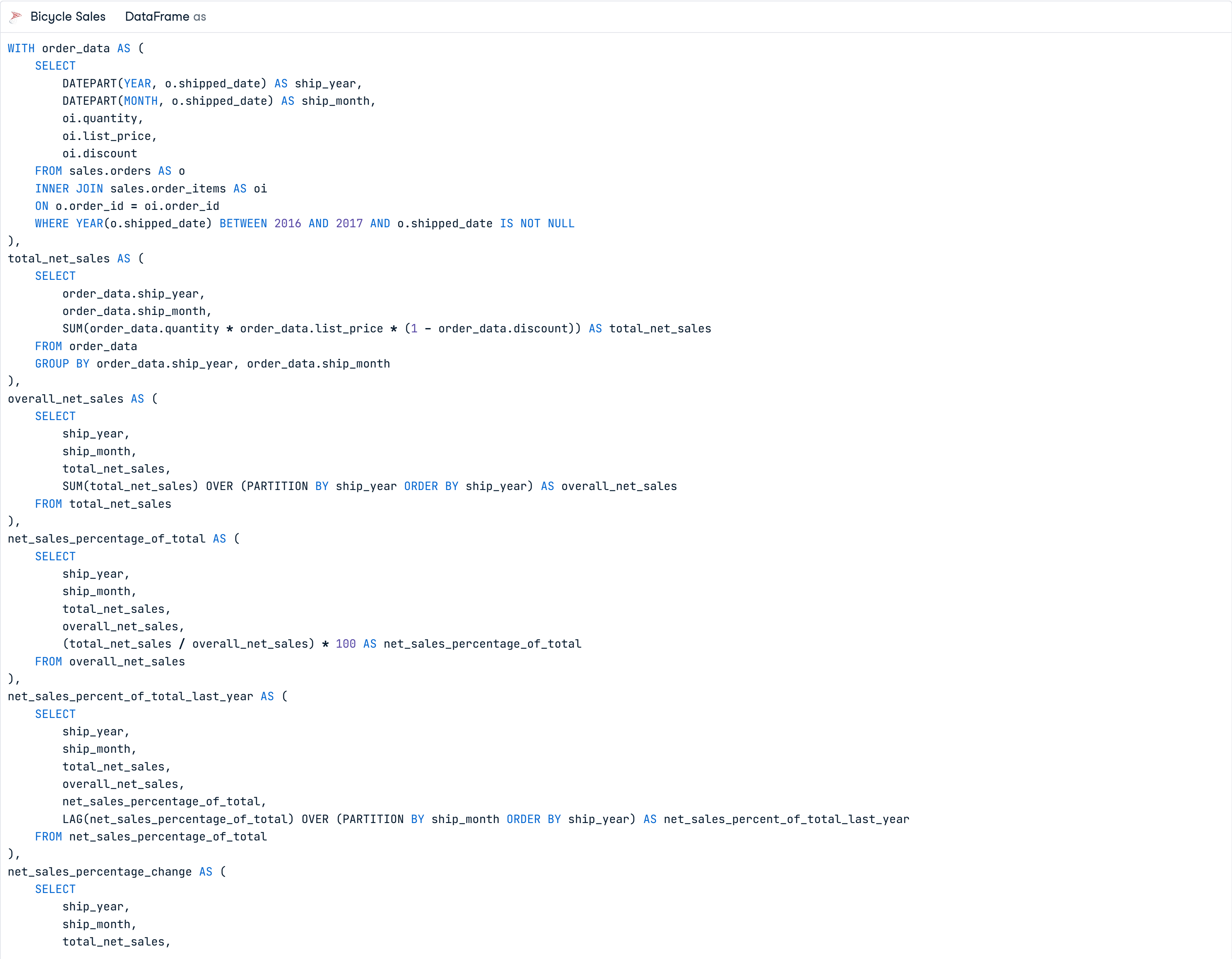
```sql
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        DATEPART(MONTH, o.shipped_date) AS ship_month,
        oi.quantity
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
total_quantity AS (
    SELECT
        order_data.ship_year,
        order_data.ship_month,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity
    FROM order_data
    GROUP BY order_data.ship_year, order_data.ship_month
),
overall_quantity AS (
    SELECT
        ship_year,
        ship_month,
        total_quantity,
        SUM(total_quantity) OVER (PARTITION BY ship_year ORDER BY ship_year) AS overall_quantity
    FROM total_quantity
),
quantity_percentage_of_total AS (
    SELECT
        ship_year,
        ship_month,
        total_quantity,
        overall_quantity,
        (total_quantity / overall_quantity) * 100 AS quantity_percentage_of_total
    FROM overall_quantity
),
quantity_percent_of_total_last_year AS (
    SELECT
        ship_year,
        ship_month,
        total_quantity,
        overall_quantity,
        quantity_percentage_of_total,
        LAG(quantity_percentage_of_total) OVER (PARTITION BY ship_month ORDER BY ship_year) AS quantity_percent_of_total_last_year
    FROM quantity_percentage_of_total
),
quantity_percentage_change AS (
    SELECT
        ship_year,
        ship_month,
        total_quantity,
        overall_quantity,
        quantity_percentage_of_total,
        quantity_percent_of_total_last_year,
        CASE
           WHEN quantity_percent_of_total_last_year IS NOT NULL THEN (quantity_percentage_of_total - quantity_percent_of_total_last_year)
           ELSE NULL
           END AS quantity_percentage_change
    FROM quantity_percent_of_total_last_year
)
SELECT
    ship_year,
    ship_month,
    total_quantity,
    overall_quantity,
    ROUND(quantity_percentage_of_total,2) AS quantity_percentage_of_total,
    ROUND(quantity_percent_of_total_last_year,2) AS quantity_percent_of_total_last_year,
    ROUND(quantity_percentage_change,2) AS quantity_percentage_change
FROM quantity_percentage_change
WHERE ship_year = 2017;
```

	s.	s...	total_qu...	overall_qua...	quantity_percentage_of_total	quantity_percent_of_total_last_year	quantity_percentage_change
0	2017	1	217	3022	7.18	7.99	-0.81
1	2017	2	230	3022	7.61	8.22	-0.61
2	2017	3	300	3022	9.93	8.34	1.59
3	2017	4	258	3022	8.54	6.23	2.31
4	2017	5	233	3022	7.71	8.99	-1.28
5	2017	6	295	3022	9.76	6.99	2.77
6	2017	7	219	3022	7.25	8.11	-0.86
7	2017	8	294	3022	9.73	8.22	1.5
8	2017	9	219	3022	7.25	10.91	-3.67
9	2017	10	294	3022	9.73	10.03	-0.3
10	2017	11	246	3022	8.14	7.19	0.95
11	2017	12	217	3022	7.18	8.76	-1.58

Rows: 12

Expand

SALES SEASONALITY TRENDS



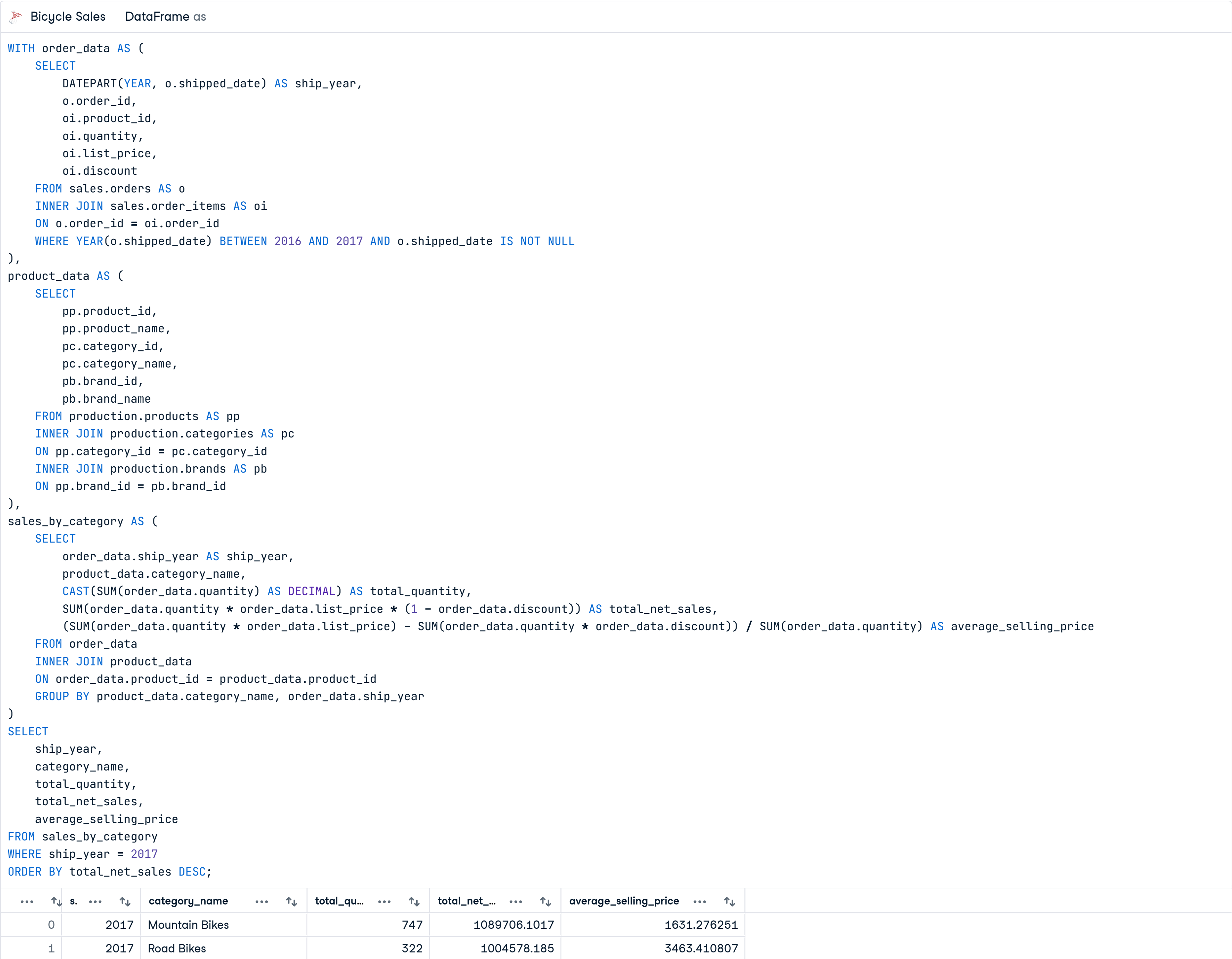
```
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        DATEPART(MONTH, o.shipped_date) AS ship_month,
        oi.quantity,
        oi.list_price,
        oi.discount
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
total_net_sales AS (
    SELECT
        order_data.ship_year,
        order_data.ship_month,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales
    FROM order_data
    GROUP BY order_data.ship_year, order_data.ship_month
),
overall_net_sales AS (
    SELECT
        ship_year,
        ship_month,
        total_net_sales,
        SUM(total_net_sales) OVER (PARTITION BY ship_year ORDER BY ship_year) AS overall_net_sales
    FROM total_net_sales
),
net_sales_percentage_of_total AS (
    SELECT
        ship_year,
        ship_month,
        total_net_sales,
        overall_net_sales,
        (total_net_sales / overall_net_sales) * 100 AS net_sales_percentage_of_total
    FROM overall_net_sales
),
net_sales_percent_of_total_last_year AS (
    SELECT
        ship_year,
        ship_month,
        total_net_sales,
        overall_net_sales,
        net_sales_percentage_of_total,
        LAG(net_sales_percentage_of_total) OVER (PARTITION BY ship_month ORDER BY ship_year) AS net_sales_percent_of_total_last_year
    FROM net_sales_percentage_of_total
),
net_sales_percentage_change AS (
    SELECT
        ship_year,
        ship_month,
        total_net_sales,
        overall_net_sales,
        net_sales_percentage_of_total,
        net_sales_percent_of_total_last_year,
        CASE
           WHEN net_sales_percent_of_total_last_year IS NOT NULL THEN (net_sales_percentage_of_total - net_sales_percent_of_total_last_year)
           ELSE NULL
           END AS net_sales_percentage_change
    FROM net_sales_percent_of_total_last_year
)
SELECT
    ship_year,
    ship_month,
    total_net_sales,
    overall_net_sales,
    net_sales_percentage_of_total,
    net_sales_percent_of_total_last_year,
    net_sales_percentage_change
FROM net_sales_percentage_change
WHERE ship_year =  2017;
```

	s.	s...	net_sales_percentage_of_total	net_sales_percent_of_total_last_year	net_sales_percentage_change
0	2017	1	7.8417	8.6576	-0.8159
1	2017	2	7.9348	6.1573	1.7775
2	2017	3	9.8604	7.7229	2.1375
3	2017	4	7.097	6.4606	0.6364
4	2017	5	7.6241	9.123	-1.4989
5	2017	6	11.3784	8.3299	3.0485
6	2017	7	5.9632	8.3731	-2.4099
7	2017	8	8.7783	8.0413	0.737
8	2017	9	8.4376	11.7418	-3.3042
9	2017	10	8.4316	9.4588	-1.0272
10	2017	11	8.8459	7.6095	1.2364
11	2017	12	7.8062	8.3236	-0.5174

Rows: 12

Expand

SALES BY CATEGORY 2017



```sql
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        o.order_id,
        oi.product_id,
        oi.quantity,
        oi.list_price,
        oi.discount
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
product_data AS (
    SELECT
        pp.product_id,
        pp.product_name,
        pc.category_id,
        pc.category_name,
        pb.brand_id,
        pb.brand_name
    FROM production.products AS pp
    INNER JOIN production.categories AS pc
    ON pp.category_id = pc.category_id
    INNER JOIN production.brands AS pb
    ON pp.brand_id = pb.brand_id
),
sales_by_category AS (
    SELECT
        order_data.ship_year AS ship_year,
        product_data.category_name,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales,
        (SUM(order_data.quantity * order_data.list_price) - SUM(order_data.quantity * order_data.discount)) / SUM(order_data.quantity) AS average_selling_price
    FROM order_data
    INNER JOIN product_data
    ON order_data.product_id = product_data.product_id
    GROUP BY product_data.category_name, order_data.ship_year
)
SELECT
    ship_year,
    category_name,
    total_quantity,
    total_net_sales,
    average_selling_price
FROM sales_by_category
WHERE ship_year = 2017
ORDER BY total_net_sales DESC;
```

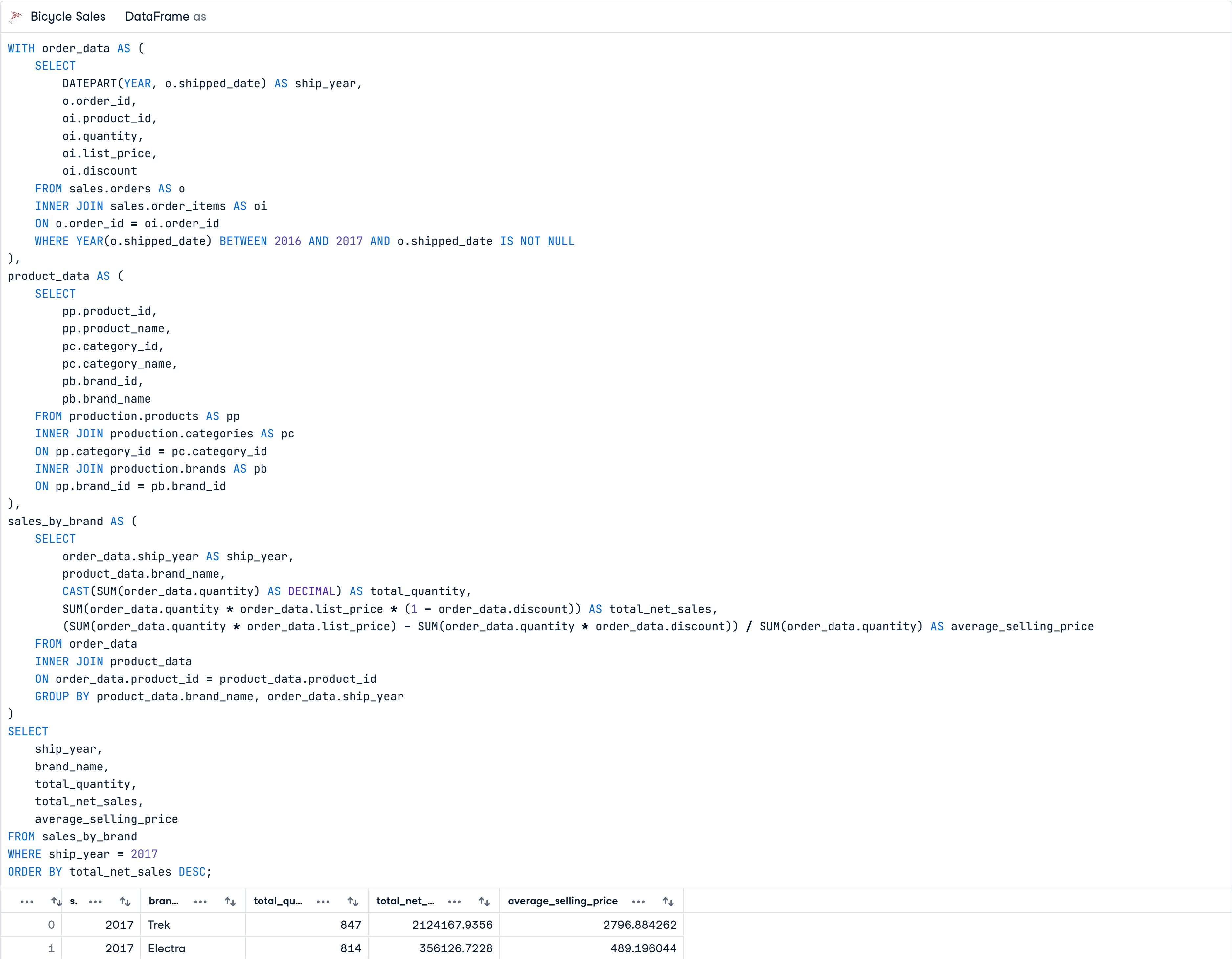
	s.	category_name	total_qu...	total_net_...	average_selling_price
0	2017	Mountain Bikes	747	1089706.1017	1631.276251
1	2017	Road Bikes	322	1004578.185	3463.410807

2	2017	Cruisers Bicycles	790	369810.9223	522.808227
3	2017	Cyclocross Bicycles	124	285443.8524	2593.110725
4	2017	Electric Bikes	95	281671.1508	3294.620736
5	2017	Comfort Bicycles	366	162680.9486	496.979453
6	2017	Children Bicycles	578	144358.7639	280.864757

Rows: 7

Expand

SALES BY BRAND 2017



```
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        o.order_id,
        oi.product_id,
        oi.quantity,
        oi.list_price,
        oi.discount
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
product_data AS (
    SELECT
        pp.product_id,
        pp.product_name,
        pc.category_id,
        pc.category_name,
        pb.brand_id,
        pb.brand_name
    FROM production.products AS pp
    INNER JOIN production.categories AS pc
    ON pp.category_id = pc.category_id
    INNER JOIN production.brands AS pb
    ON pp.brand_id = pb.brand_id
),
sales_by_brand AS (
    SELECT
        order_data.ship_year AS ship_year,
        product_data.brand_name,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales,
        (SUM(order_data.quantity * order_data.list_price) - SUM(order_data.quantity * order_data.discount)) / SUM(order_data.quantity) AS average_selling_price
    FROM order_data
    INNER JOIN product_data
    ON order_data.product_id = product_data.product_id
    GROUP BY product_data.brand_name, order_data.ship_year
)
SELECT
    ship_year,
    brand_name,
    total_quantity,
    total_net_sales,
    average_selling_price
FROM sales_by_brand
WHERE ship_year = 2017
ORDER BY total_net_sales DESC;
```

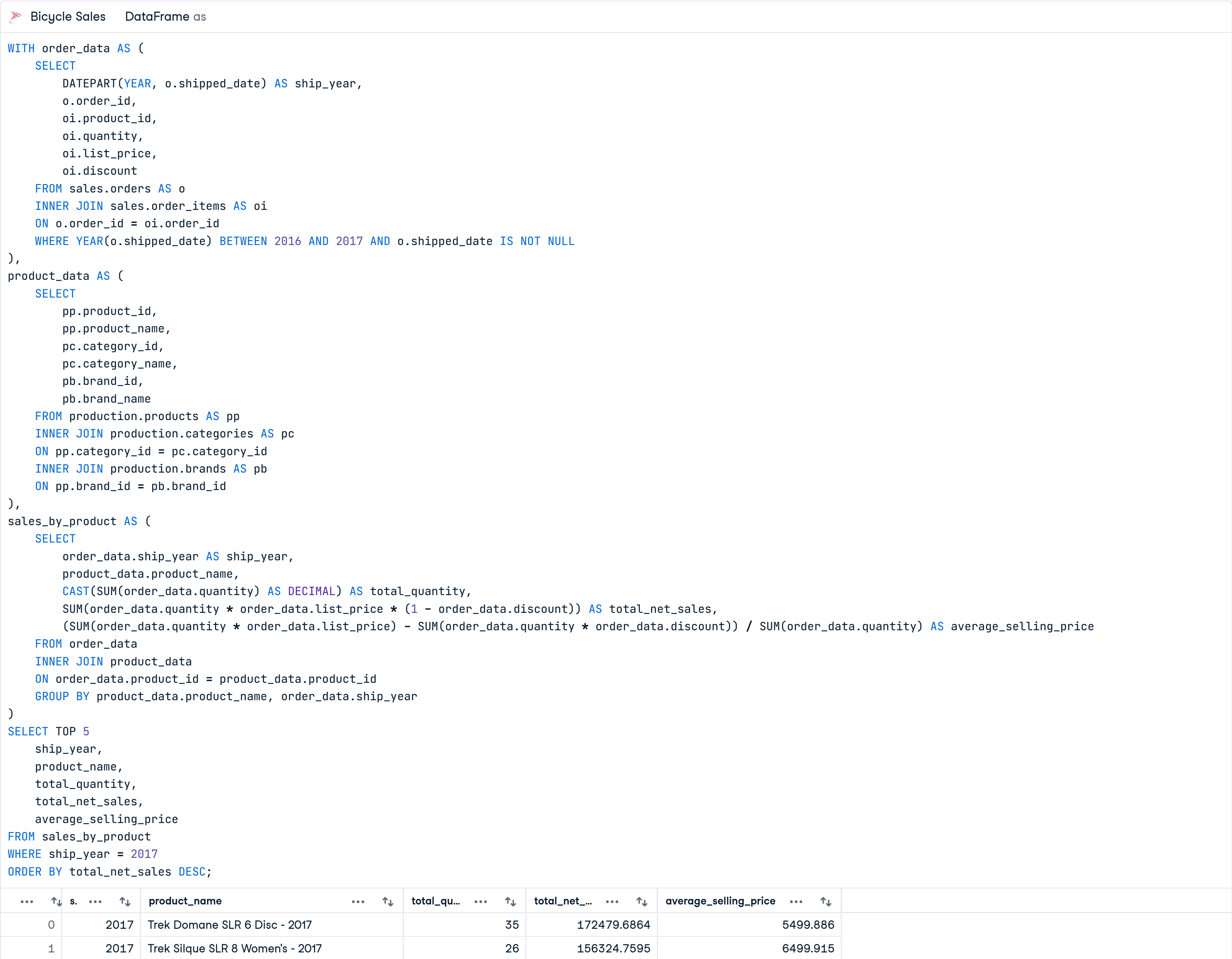
	s.	bran...	total_qu...	total_net_...	average_selling_price
0	2017	Trek	847	2124167.9356	2796.884262
1	2017	Electra	814	356126.7228	489.196044

2	2017	Surly	339	333011.6385	1094.085457
3	2017	Sun Bicycles	615	286033.5058	519.844341
4	2017	Haro	280	156137.5153	625.555964
5	2017	Heller	35	41465.8761	1320.886857
6	2017	Pure Cycles	73	28601.9	440.669863
7	2017	Ritchey	19	12704.8306	749.881578

Rows: 8

Expand

TOP 5 SALES BY PRODUCT 2017



```sql
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        o.order_id,
        oi.product_id,
        oi.quantity,
        oi.list_price,
        oi.discount
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
product_data AS (
    SELECT
        pp.product_id,
        pp.product_name,
        pc.category_id,
        pc.category_name,
        pb.brand_id,
        pb.brand_name
    FROM production.products AS pp
    INNER JOIN production.categories AS pc
    ON pp.category_id = pc.category_id
    INNER JOIN production.brands AS pb
    ON pp.brand_id = pb.brand_id
),
sales_by_product AS (
    SELECT
        order_data.ship_year AS ship_year,
        product_data.product_name,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales,
        (SUM(order_data.quantity * order_data.list_price) - SUM(order_data.quantity * order_data.discount)) / SUM(order_data.quantity) AS average_selling_price
    FROM order_data
    INNER JOIN product_data
    ON order_data.product_id = product_data.product_id
    GROUP BY product_data.product_name, order_data.ship_year
)
SELECT TOP 5
    ship_year,
    product_name,
    total_quantity,
    total_net_sales,
    average_selling_price
FROM sales_by_product
WHERE ship_year = 2017
ORDER BY total_net_sales DESC;
```

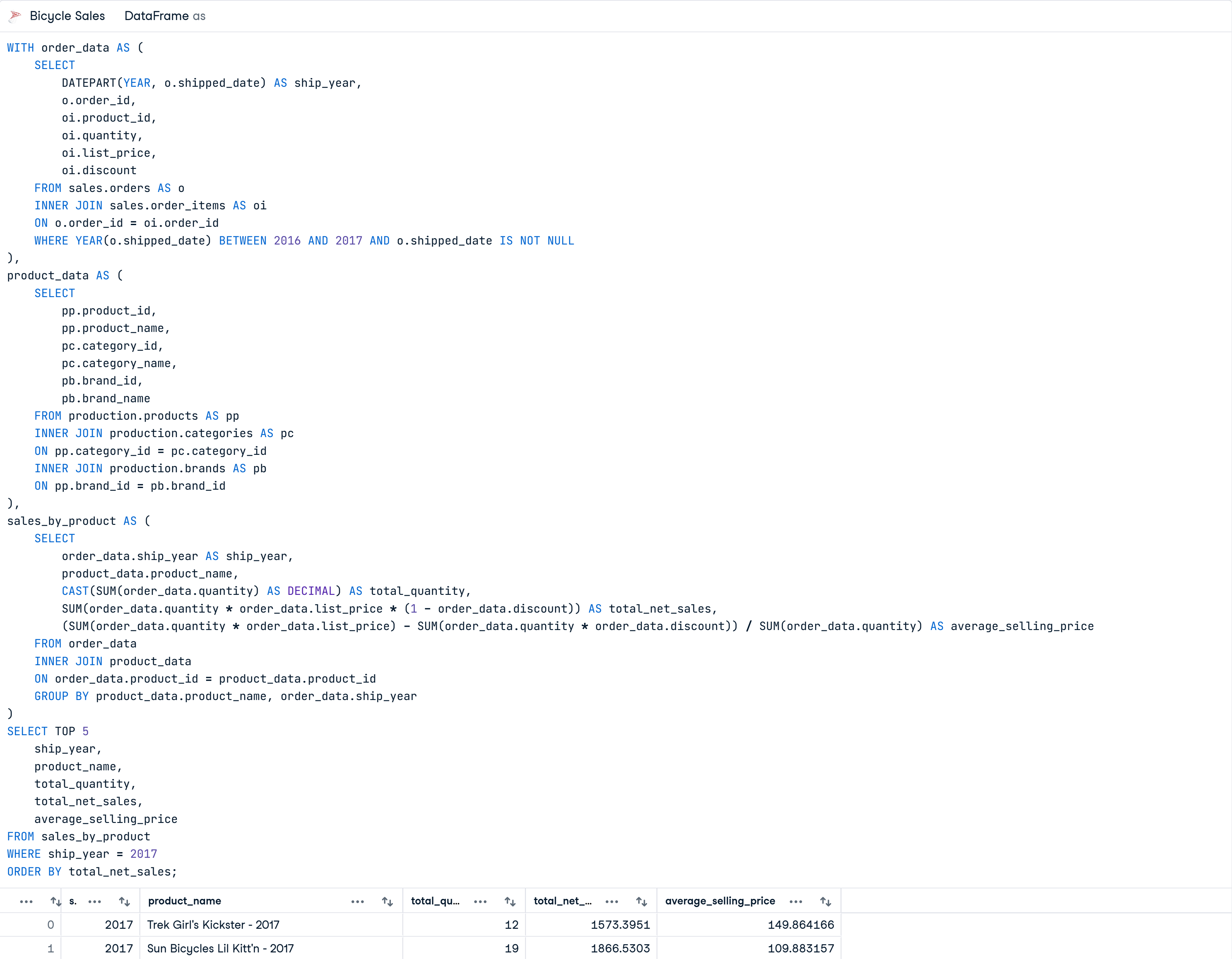
	s.	product_name	total_qu...	total_net_...	average_selling_price
0	2017	Trek Domane SLR 6 Disc - 2017	35	172479.6864	5499.886
1	2017	Trek Silque SLR 8 Women's - 2017	26	156324.7595	6499.915

2	2017	Trek Madone 9.2 - 2017	33	148299.7034	4999.888787
3	2017	Trek Powerfly 8 FS Plus - 2017	32	146299.7074	4999.904375
4	2017	Trek Silque SLR 7 Women's - 2017	25	136319.7728	5999.8988

Rows: 5

Expand

BOTTOM 5 SALES BY PRODUCT 2017



Bicycle Sales DataFrame as

```sql
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        o.order_id,
        oi.product_id,
        oi.quantity,
        oi.list_price,
        oi.discount
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
product_data AS (
    SELECT
        pp.product_id,
        pp.product_name,
        pc.category_id,
        pc.category_name,
        pb.brand_id,
        pb.brand_name
    FROM production.products AS pp
    INNER JOIN production.categories AS pc
    ON pp.category_id = pc.category_id
    INNER JOIN production.brands AS pb
    ON pp.brand_id = pb.brand_id
),
sales_by_product AS (
    SELECT
        order_data.ship_year AS ship_year,
        product_data.product_name,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales,
        (SUM(order_data.quantity * order_data.list_price) - SUM(order_data.quantity * order_data.discount)) / SUM(order_data.quantity) AS average_selling_price
    FROM order_data
    INNER JOIN product_data
    ON order_data.product_id = product_data.product_id
    GROUP BY product_data.product_name, order_data.ship_year
)
SELECT TOP 5
    ship_year,
    product_name,
    total_quantity,
    total_net_sales,
    average_selling_price
FROM sales_by_product
WHERE ship_year = 2017
ORDER BY total_net_sales;
```

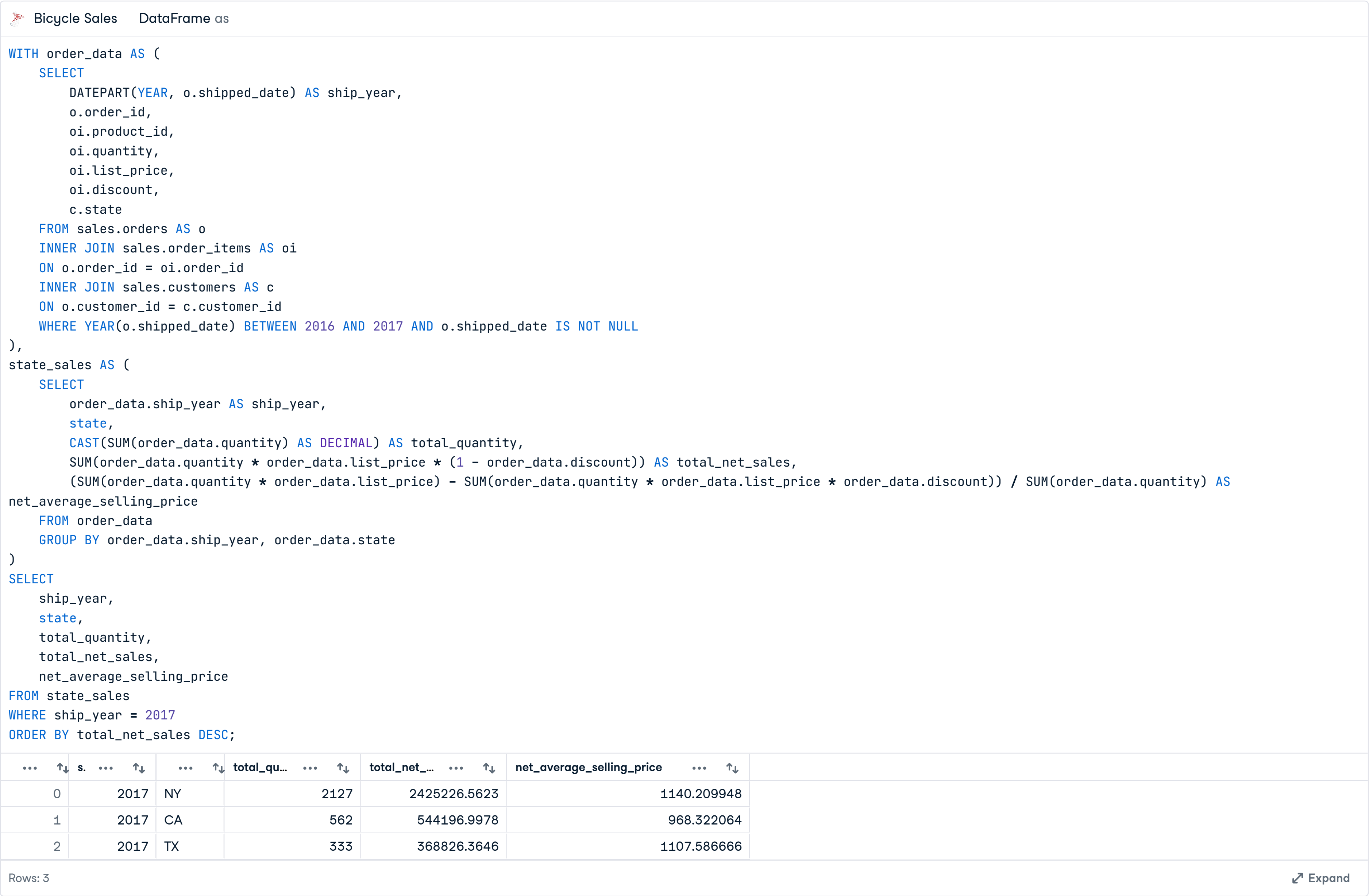
	s.	product_name	total_qu...	total_net_...	average_selling_price
0	2017	Trek Girl's Kickster - 2017	12	1573.3951	149.864166
1	2017	Sun Bicycles Lil Kitt'n - 2017	19	1866.5303	109.883157

2	2017	Trek Boy's Kickster - 2015/2017	23	3116.7922	149.893478
3	2017	Trek Precaliber 16 Boys - 2017	17	3200.2476	209.88647
4	2017	Trek Precaliber 16 Girls - 2017	23	4193.5003	209.85826

Rows: 5

Expand

SALES BY STATE 2017



```sql
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        o.order_id,
        oi.product_id,
        oi.quantity,
        oi.list_price,
        oi.discount,
        c.state
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    INNER JOIN sales.customers AS c
    ON o.customer_id = c.customer_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
state_sales AS (
    SELECT
        order_data.ship_year AS ship_year,
        state,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales,
        (SUM(order_data.quantity * order_data.list_price) - SUM(order_data.quantity * order_data.list_price * order_data.discount)) / SUM(order_data.quantity) AS
net_average_selling_price
    FROM order_data
    GROUP BY order_data.ship_year, order_data.state
)
SELECT
    ship_year,
    state,
    total_quantity,
    total_net_sales,
    net_average_selling_price
FROM state_sales
WHERE ship_year = 2017
ORDER BY total_net_sales DESC;
```

	s.		total_qu...	total_net_...	net_average_selling_price
0	2017	NY	2127	2425226.5623	1140.209948
1	2017	CA	562	544196.9978	968.322064
2	2017	TX	333	368826.3646	1107.586666

Rows: 3

CUSTOMERS RETENTION

Bicycle Sales DataFrame as

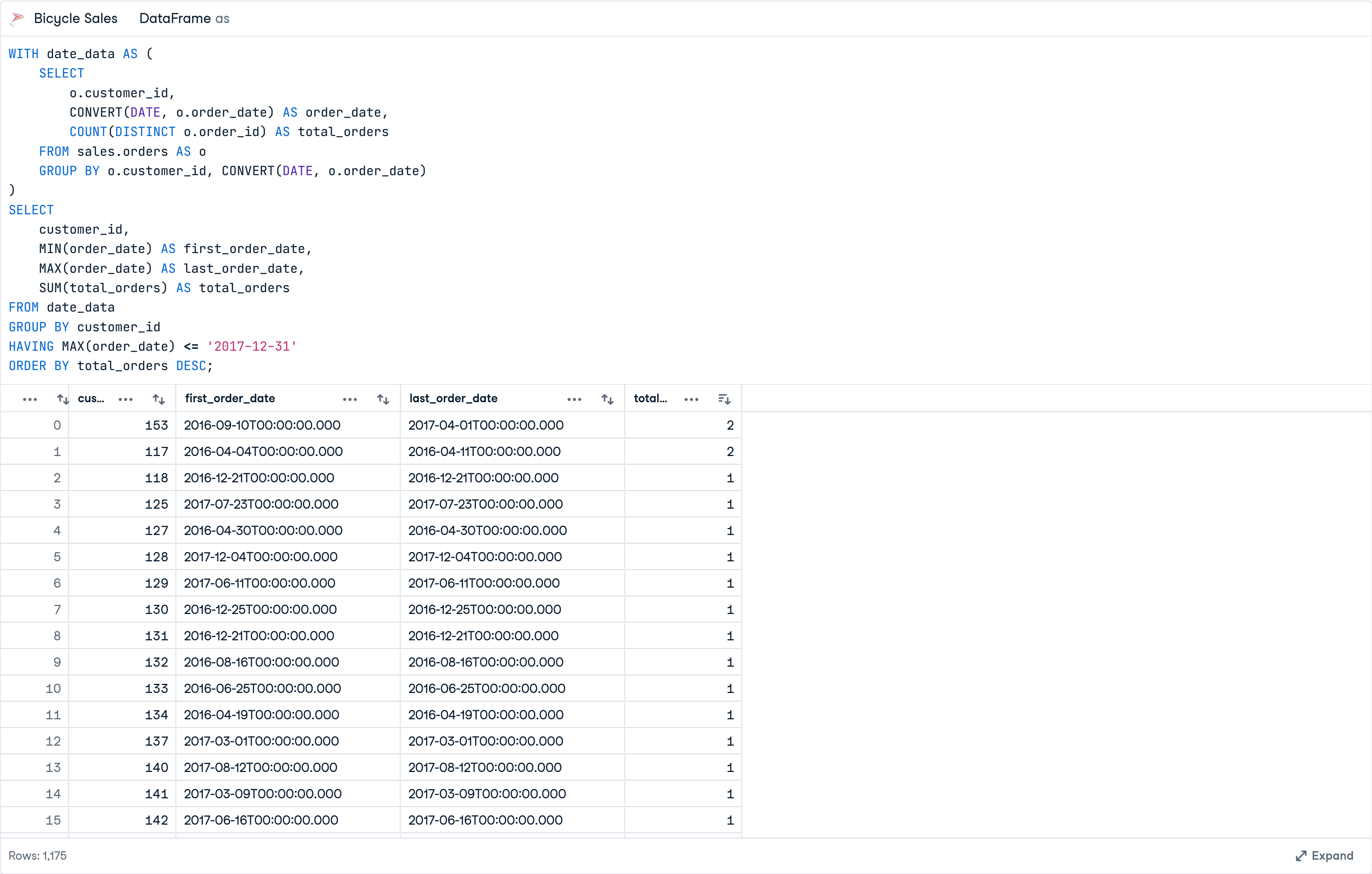
```sql
SELECT
    o.customer_id,
    COUNT(DISTINCT o.order_id) AS number_of_orders
FROM sales.orders AS o
WHERE DATEPART(YEAR, o.order_date) BETWEEN 2016 AND 2017 AND o.order_date IS NOT NULL
GROUP BY o.customer_id, DATEPART(YEAR, o.order_date)
HAVING COUNT(DISTINCT o.order_id) > 1
ORDER BY number_of_orders DESC;
```

	cus...	number_of_...
0	15	3
1	3	3
2	6	2
3	7	2
4	9	2
5	10	2
6	11	2
7	18	2
8	21	2
9	33	2
10	35	2
11	53	2
12	56	2
13	63	2
14	68	2
15	71	2

Rows: 34

Expand

CUSTOMER PURCHASE TRENDS

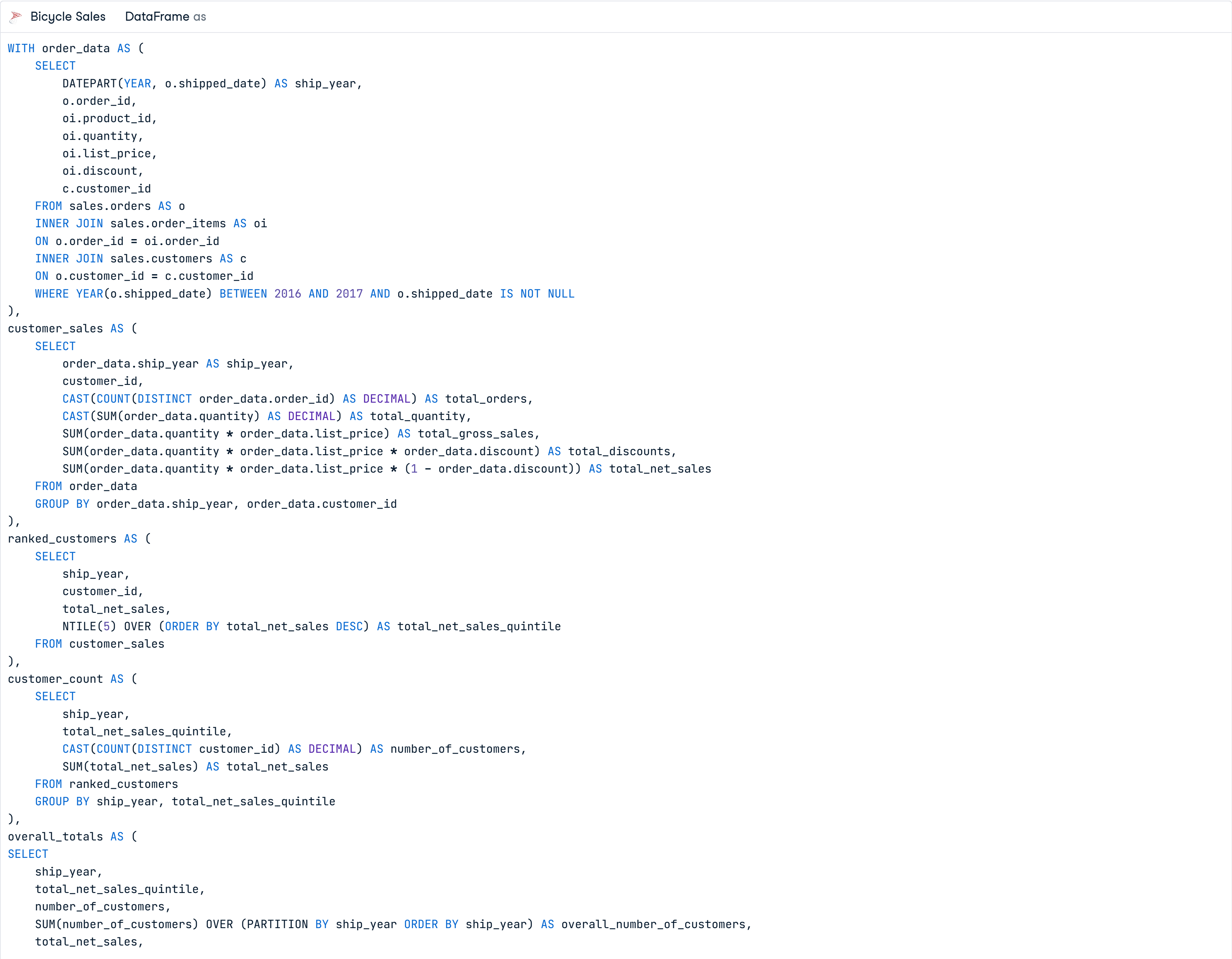


```sql
WITH date_data AS (
    SELECT
        o.customer_id,
        CONVERT(DATE, o.order_date) AS order_date,
        COUNT(DISTINCT o.order_id) AS total_orders
    FROM sales.orders AS o
    GROUP BY o.customer_id, CONVERT(DATE, o.order_date)
)
SELECT
    customer_id,
    MIN(order_date) AS first_order_date,
    MAX(order_date) AS last_order_date,
    SUM(total_orders) AS total_orders
FROM date_data
GROUP BY customer_id
HAVING MAX(order_date) <= '2017-12-31'
ORDER BY total_orders DESC;
```

	cus...	first_order_date	last_order_date	total...
0	153	2016-09-10T00:00:00.000	2017-04-01T00:00:00.000	2
1	117	2016-04-04T00:00:00.000	2016-04-11T00:00:00.000	2
2	118	2016-12-21T00:00:00.000	2016-12-21T00:00:00.000	1
3	125	2017-07-23T00:00:00.000	2017-07-23T00:00:00.000	1
4	127	2016-04-30T00:00:00.000	2016-04-30T00:00:00.000	1
5	128	2017-12-04T00:00:00.000	2017-12-04T00:00:00.000	1
6	129	2017-06-11T00:00:00.000	2017-06-11T00:00:00.000	1
7	130	2016-12-25T00:00:00.000	2016-12-25T00:00:00.000	1
8	131	2016-12-21T00:00:00.000	2016-12-21T00:00:00.000	1
9	132	2016-08-16T00:00:00.000	2016-08-16T00:00:00.000	1
10	133	2016-06-25T00:00:00.000	2016-06-25T00:00:00.000	1
11	134	2016-04-19T00:00:00.000	2016-04-19T00:00:00.000	1
12	137	2017-03-01T00:00:00.000	2017-03-01T00:00:00.000	1
13	140	2017-08-12T00:00:00.000	2017-08-12T00:00:00.000	1
14	141	2017-03-09T00:00:00.000	2017-03-09T00:00:00.000	1
15	142	2017-06-16T00:00:00.000	2017-06-16T00:00:00.000	1

Rows: 1,175

SALES AND CUSTOMERS BY QUINTILE



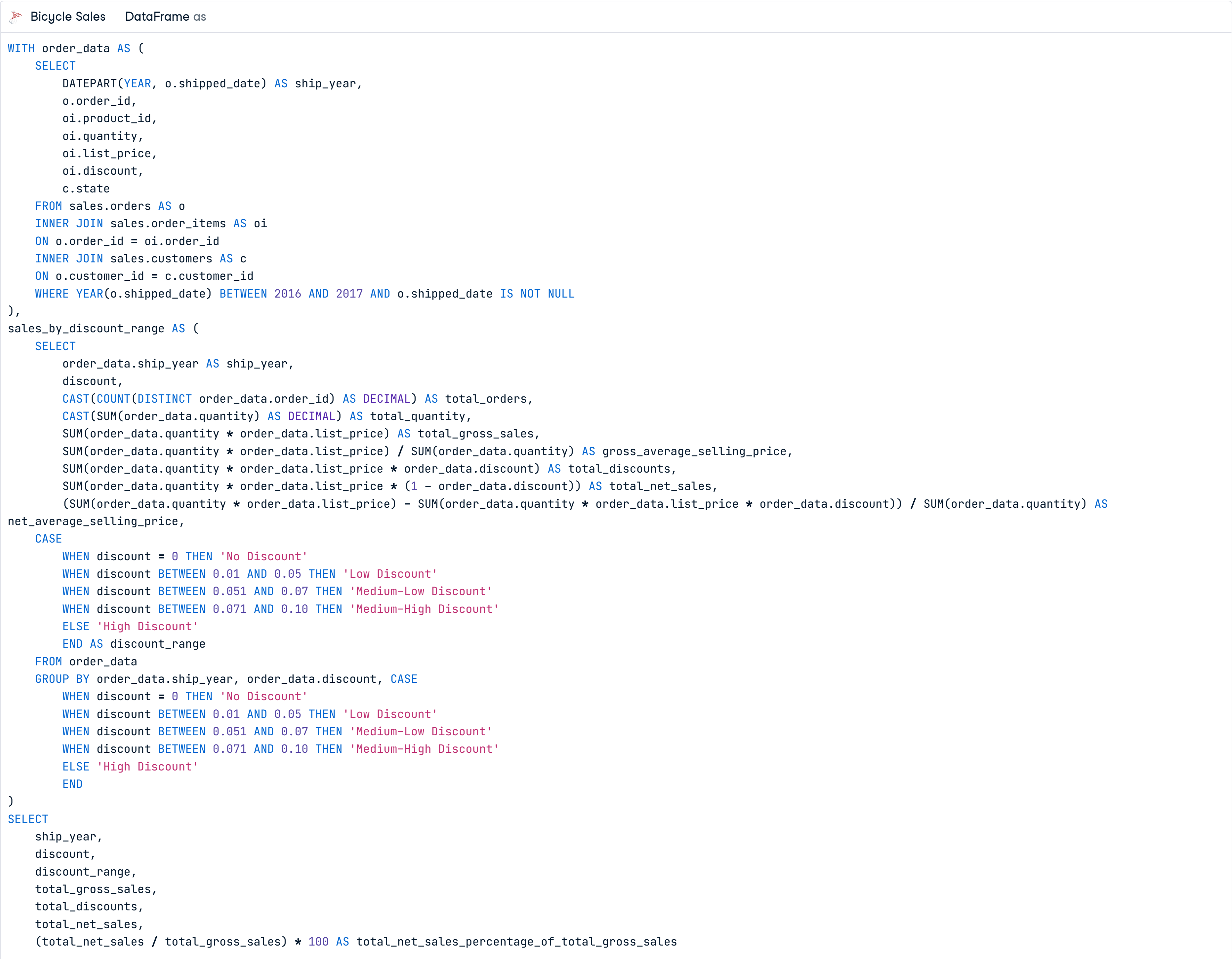
```sql
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        o.order_id,
        oi.product_id,
        oi.quantity,
        oi.list_price,
        oi.discount,
        c.customer_id
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    INNER JOIN sales.customers AS c
    ON o.customer_id = c.customer_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
customer_sales AS (
    SELECT
        order_data.ship_year AS ship_year,
        customer_id,
        CAST(COUNT(DISTINCT order_data.order_id) AS DECIMAL) AS total_orders,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity,
        SUM(order_data.quantity * order_data.list_price) AS total_gross_sales,
        SUM(order_data.quantity * order_data.list_price * order_data.discount) AS total_discounts,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales
    FROM order_data
    GROUP BY order_data.ship_year, order_data.customer_id
),
ranked_customers AS (
    SELECT
        ship_year,
        customer_id,
        total_net_sales,
        NTILE(5) OVER (ORDER BY total_net_sales DESC) AS total_net_sales_quintile
    FROM customer_sales
),
customer_count AS (
    SELECT
        ship_year,
        total_net_sales_quintile,
        CAST(COUNT(DISTINCT customer_id) AS DECIMAL) AS number_of_customers,
        SUM(total_net_sales) AS total_net_sales
    FROM ranked_customers
    GROUP BY ship_year, total_net_sales_quintile
),
overall_totals AS (
SELECT
    ship_year,
    total_net_sales_quintile,
    number_of_customers,
    SUM(number_of_customers) OVER (PARTITION BY ship_year ORDER BY ship_year) AS overall_number_of_customers,
    total_net_sales,
    SUM(total_net_sales) OVER (PARTITION BY ship_year ORDER BY ship_year) AS overall_net_sales
FROM customer_count
),
percentage_of_totals AS (
SELECT
    ship_year,
    total_net_sales_quintile,
    number_of_customers,
    overall_number_of_customers,
    (number_of_customers / overall_number_of_customers) * 100 AS percentage_of_total_customers,
    total_net_sales,
    overall_net_sales,
    (total_net_sales / overall_net_sales) * 100 AS percentage_of_total_net_sales
FROM overall_totals
)
SELECT
    ship_year,
    total_net_sales_quintile,
    number_of_customers,
    overall_number_of_customers,
    ROUND(percentage_of_total_customers,2) AS percentage_of_total_customers,
    total_net_sales,
    overall_net_sales,
    ROUND(percentage_of_total_net_sales,2) AS percentage_of_total_net_sales
FROM percentage_of_totals
WHERE ship_year = 2017;
```

••• ↑↓	s. ••• ↑↓	total_net_sales_quintile ••• ↑↓	number_of_custo... ••• ↑↓	percentage_of_total_customers ••• ↑↓	total_net_... ••• ↑↓	percentage_of_total_net_sales ••• ↑↓
0	2017	1	170	25.41	1912766.9156	57.3
1	2017	2	120	17.94	680488.0147	20.38
2	2017	3	122	18.24	415597.6521	12.45
3	2017	4	125	18.68	234469.8423	7.02
4	2017	5	132	19.73	94927.5	2.84

Rows: 5

⤢ Expand

SALES BY DISCOUNT LEVEL



```
WITH order_data AS (
    SELECT
        DATEPART(YEAR, o.shipped_date) AS ship_year,
        o.order_id,
        oi.product_id,
        oi.quantity,
        oi.list_price,
        oi.discount,
        c.state
    FROM sales.orders AS o
    INNER JOIN sales.order_items AS oi
    ON o.order_id = oi.order_id
    INNER JOIN sales.customers AS c
    ON o.customer_id = c.customer_id
    WHERE YEAR(o.shipped_date) BETWEEN 2016 AND 2017 AND o.shipped_date IS NOT NULL
),
sales_by_discount_range AS (
    SELECT
        order_data.ship_year AS ship_year,
        discount,
        CAST(COUNT(DISTINCT order_data.order_id) AS DECIMAL) AS total_orders,
        CAST(SUM(order_data.quantity) AS DECIMAL) AS total_quantity,
        SUM(order_data.quantity * order_data.list_price) AS total_gross_sales,
        SUM(order_data.quantity * order_data.list_price) / SUM(order_data.quantity) AS gross_average_selling_price,
        SUM(order_data.quantity * order_data.list_price * order_data.discount) AS total_discounts,
        SUM(order_data.quantity * order_data.list_price * (1 - order_data.discount)) AS total_net_sales,
        (SUM(order_data.quantity * order_data.list_price) - SUM(order_data.quantity * order_data.list_price * order_data.discount)) / SUM(order_data.quantity) AS
net_average_selling_price,
    CASE
        WHEN discount = 0 THEN 'No Discount'
        WHEN discount BETWEEN 0.01 AND 0.05 THEN 'Low Discount'
        WHEN discount BETWEEN 0.051 AND 0.07 THEN 'Medium-Low Discount'
        WHEN discount BETWEEN 0.071 AND 0.10 THEN 'Medium-High Discount'
        ELSE 'High Discount'
        END AS discount_range
    FROM order_data
    GROUP BY order_data.ship_year, order_data.discount, CASE
        WHEN discount = 0 THEN 'No Discount'
        WHEN discount BETWEEN 0.01 AND 0.05 THEN 'Low Discount'
        WHEN discount BETWEEN 0.051 AND 0.07 THEN 'Medium-Low Discount'
        WHEN discount BETWEEN 0.071 AND 0.10 THEN 'Medium-High Discount'
        ELSE 'High Discount'
        END
)
SELECT
    ship_year,
    discount,
    discount_range,
    total_gross_sales,
    total_discounts,
    total_net_sales,
    (total_net_sales / total_gross_sales) * 100 AS total_net_sales_percentage_of_total_gross_sales
```

```sql
FROM sales_by_discount_range
WHERE ship_year = 2017
ORDER BY discount;
```

	s.	discount_range	total_gross_s...	total_disc...	total_net_...	total_net_sales_percentage_of_tota...
3	2017	High Discount	915134.28	183026.856	732107.424	80
2	2017	Medium-High Discount	895801.93	89580.193	806221.737	90
1	2017	Medium-Low Discount	964302.94	67501.2058	896801.7342	93
0	2017	Low Discount	950651.61	47532.5805	903119.0295	95

Rows: 4

Expand