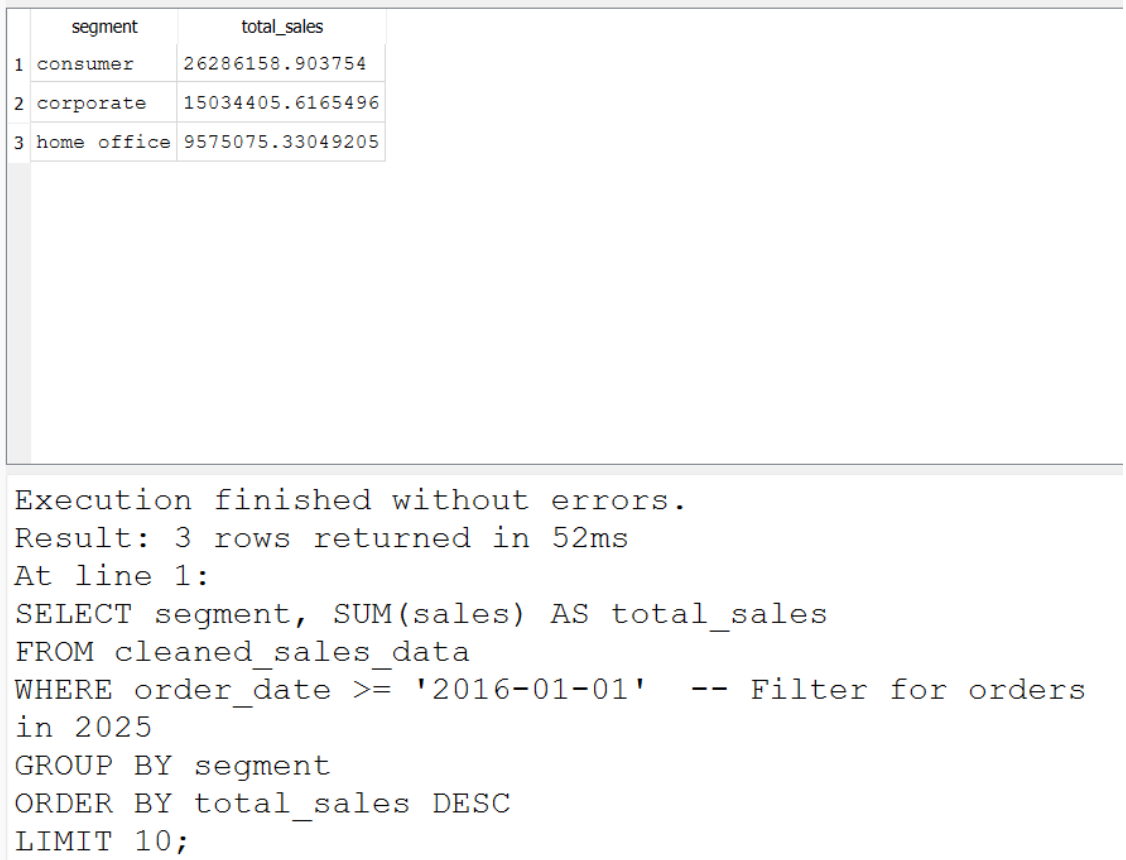
1. Query using SELECT, WHERE, ORDER BY, and GROUP BY

```
SELECT segment, SUM(sales) AS total_sales
FROM cleaned_sales_data
WHERE order_date >= '2016-01-01'  -- Filter for orders in 2025
GROUP BY segment
ORDER BY total_sales DESC
LIMIT 10;
```



	segment	total_sales
1	consumer	26286158.903754
2	corporate	15034405.6165496
3	home office	9575075.33049205

```
Execution finished without errors.
Result: 3 rows returned in 52ms
At line 1:
SELECT segment, SUM(sales) AS total_sales
FROM cleaned_sales_data
WHERE order_date >= '2016-01-01'  -- Filter for orders
in 2025
GROUP BY segment
ORDER BY total_sales DESC
LIMIT 10;
```

2. **INNER JOIN to get orders with product details (sales > 500)**

```
SELECT o1.order_id, o1.sales, o1.product_name
FROM cleaned_sales_data o1
INNER JOIN cleaned_sales_data o2 ON o1.product_id = o2.product_id
WHERE o1.sales > 500
LIMIT 10;
```

	order_id	sales	product_name
1	CA-2017-152156	3604.243977	Bush Somerset Collection Bookcase
2	CA-2017-152156	3604.243977	Bush Somerset Collection Bookcase
3	CA-2017-152156	3604.243977	Bush Somerset Collection Bookcase
4	CA-2017-152156	3604.243977	Bush Somerset Collection Bookcase
5	CA-2017-152156	1664.369269	Hon Deluxe Fabric Upholstered …
6	CA-2017-152156	1664.369269	Hon Deluxe Fabric Upholstered …
7	CA-2017-152156	1664.369269	Hon Deluxe Fabric Upholstered …
8	CA-2017-152156	1664.369269	Hon Deluxe Fabric Upholstered …
9	CA-2017-152156	1664.369269	Hon Deluxe Fabric Upholstered …
10	CA-2017-152156	1664.369269	Hon Deluxe Fabric Upholstered …

```
Execution finished without errors.
Result: 10 rows returned in 46ms
At line 1:
SELECT o1.order_id, o1.sales, o1.product_name
FROM cleaned_sales_data o1
INNER JOIN cleaned_sales_data o2 ON o1.product_id =
o2.product_id
WHERE o1.sales > 500
LIMIT 10;
```

3. **Query with Subqueries**

SELECT order_id, customer_name, sales

FROM cleaned_sales_data

WHERE sales > (SELECT AVG(sales) FROM cleaned_sales_data)

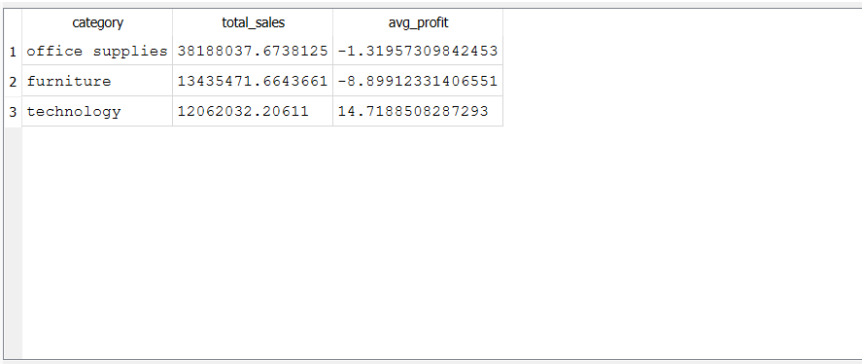
LIMIT 10;

	order_id	customer_name	sales
1	CA-2015-115812	Brosina Hoffman	6578.957164
2	CA-2015-115812	Brosina Hoffman	19584.93743
3	CA-2015-115812	Brosina Hoffman	7418.467726
4	CA-2015-115812	Brosina Hoffman	11209.08286
5	CA-2015-115812	Brosina Hoffman	14534.26161
6	CA-2018-114412	Andrew Allen	11459.14745
7	CA-2017-161389	Irene Maddox	21321.118
8	US-2016-118983	Harold Pawlan	12904.31631
9	CA-2015-167164	Alejandro Grove	9833.289446
10	US-2018-156909	Sandra Flanagan	7793.331876

```
Execution finished without errors.
Result: 10 rows returned in 12ms
At line 1:
SELECT order_id, customer_name, sales
FROM cleaned_sales_data
WHERE sales > (SELECT AVG(sales) FROM cleaned_sales_data)
LIMIT 10;
```

4. **Query with Aggregate Functions (SUM, AVG)**

```
SELECT category, SUM(sales) AS total_sales, AVG(profit) AS avg_profit
FROM cleaned_sales_data
GROUP BY category
ORDER BY total_sales DESC
LIMIT 10;
```



	category	total_sales	avg_profit
1	office supplies	38188037.6738125	-1.31957309842453
2	furniture	13435471.6643661	-8.89912331406551
3	technology	12062032.20611	14.7188508287293

```
Execution finished without errors.
Result: 3 rows returned in 25ms
At line 1:
SELECT category, SUM(sales) AS total_sales, AVG(profit)
AS avg_profit
FROM cleaned_sales_data
GROUP BY category
ORDER BY total_sales DESC
LIMIT 10;
```

5. **Create View for Analysis & Optimize Queries with Indexes**

```
-- Create a view for total sales by region
CREATE VIEW region_sales AS
SELECT region, SUM(sales) AS total_sales
FROM cleaned_sales_data
GROUP BY region;


-- Optimizing queries with an index on the 'order_date' column
CREATE INDEX idx_order_date ON cleaned_sales_data(order_date);


-- Querying the view created above
SELECT * FROM region_sales
ORDER BY total_sales DESC
```

LIMIT 10;

	region	total_sales
1	West	19966670.4083475
2	East	18356772.587054
3	Central	14741915.5748196
4	South	10620182.9740676

```
Execution finished without errors.
Result: 4 rows returned in 15ms
At line 10:
-- Querying the view created above
SELECT * FROM region_sales
ORDER BY total_sales DESC
LIMIT 10;
```