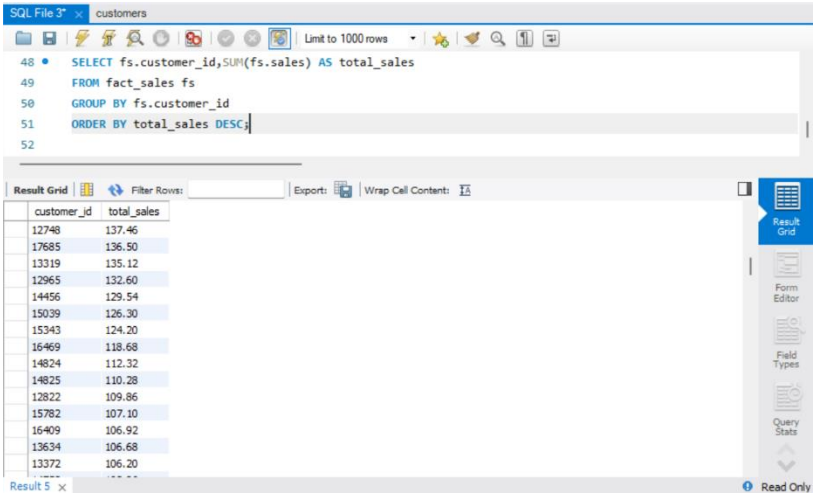
Task 3: SQL for Data Analysis

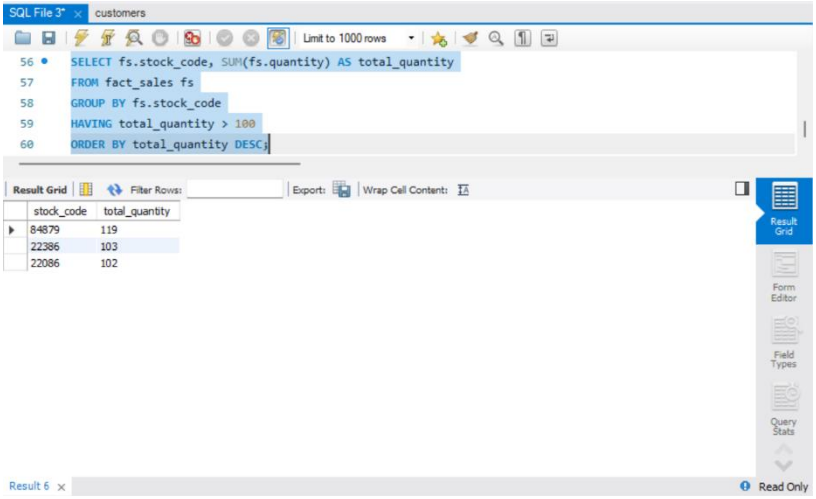
-- 1. Total sales per customer, ordered by total sales:

```
SELECT fs.customer_id,SUM(fs.sales) AS total_sales
FROM fact_sales fs
GROUP BY fs.customer_id
ORDER BY total_sales DESC;
```



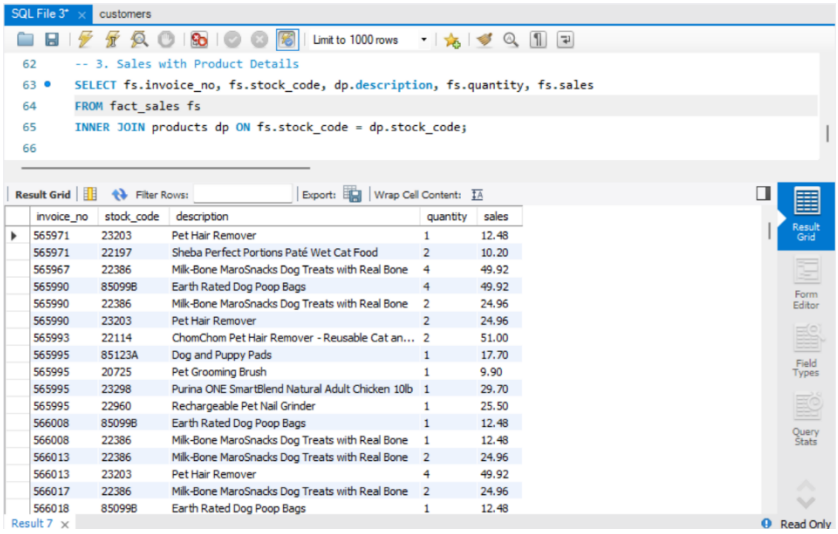
-- 2. Products sold more than 100 times:

```
SELECT fs.stock_code, SUM(fs.quantity) AS total_quantity
FROM fact_sales fs
GROUP BY fs.stock_code
HAVING total_quantity > 100
ORDER BY total_quantity DESC;
```



-- 3. Sales with Product Details

```
SELECT fs.invoice_no, fs.stock_code, dp.description, fs.quantity, fs.sales
FROM fact_sales fs
INNER JOIN products dp ON fs.stock_code = dp.stock_code;
```



```
-- 3. Sales with Product Details
SELECT fs.invoice_no, fs.stock_code, dp.description, fs.quantity, fs.sales
FROM fact_sales fs
INNER JOIN products dp ON fs.stock_code = dp.stock_code;
```

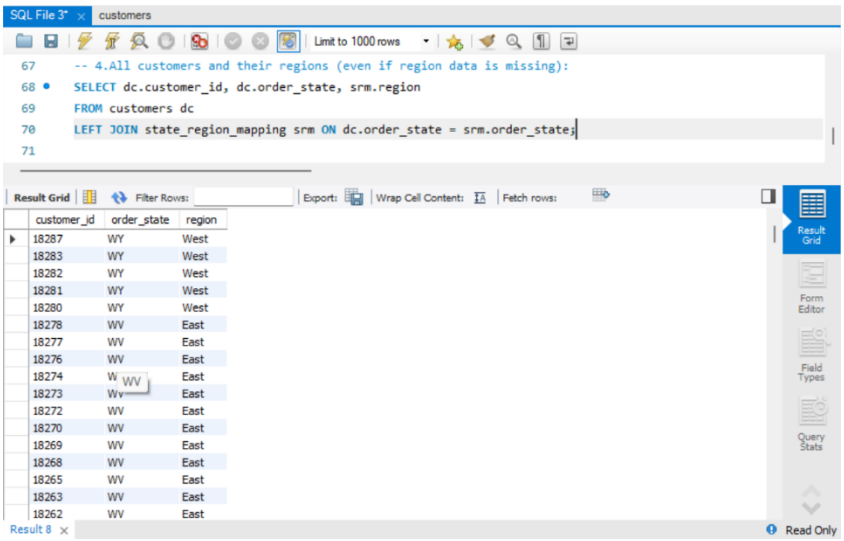
invoice_no	stock_code	description	quantity	sales
565971	23203	Pet Hair Remover	1	12.48
565971	22197	Sheba Perfect Portions Paté Wet Cat Food	2	10.20
565967	22386	Milk-Bone MaroSnacks Dog Treats with Real Bone	4	49.92
565990	85099B	Earth Rated Dog Poop Bags	4	49.92
565990	22386	Milk-Bone MaroSnacks Dog Treats with Real Bone	2	24.96
565990	23203	Pet Hair Remover	2	24.96
565993	22114	ChomChom Pet Hair Remover - Reusable Cat an...	2	51.00
565995	85123A	Dog and Puppy Pads	1	17.70
565995	20725	Pet Grooming Brush	1	9.90
565995	23298	Purina ONE SmartBlend Natural Adult Chicken 10lb	1	29.70
565995	22960	Rechargeable Pet Nail Grinder	1	25.50
566008	85099B	Earth Rated Dog Poop Bags	1	12.48
566008	22386	Milk-Bone MaroSnacks Dog Treats with Real Bone	1	12.48
566013	22386	Milk-Bone MaroSnacks Dog Treats with Real Bone	2	24.96
566013	23203	Pet Hair Remover	4	49.92
566017	22386	Milk-Bone MaroSnacks Dog Treats with Real Bone	2	24.96
566018	85099B	Earth Rated Dog Poop Bags	1	12.48

-- 4. All customers and their regions (even if region data is missing):

SELECT dc.customer_id, dc.order_state, srm.region

FROM customers dc

LEFT JOIN state_region_mapping srm ON dc.order_state = srm.order_state;



```
-- 4.All customers and their regions (even if region data is missing):
SELECT dc.customer_id, dc.order_state, srm.region
FROM customers dc
LEFT JOIN state_region_mapping srm ON dc.order_state = srm.order_state;
```

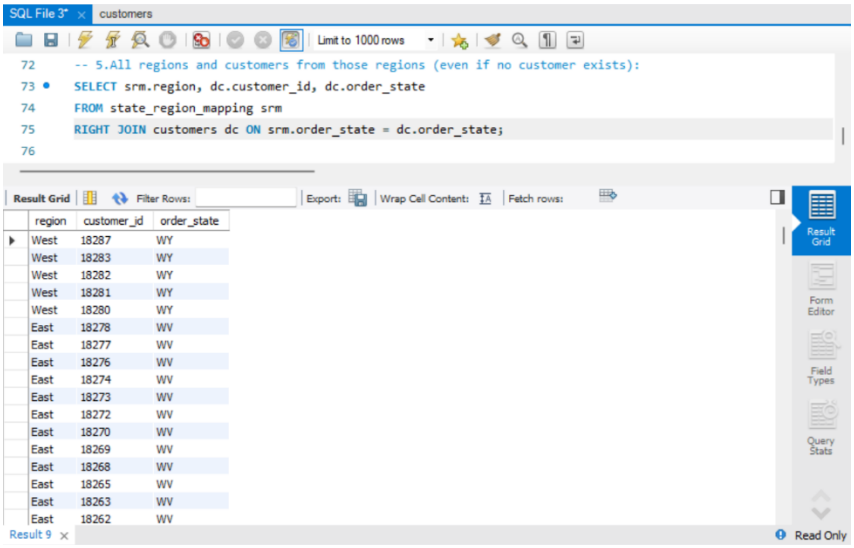
customer_id	order_state	region
18287	WY	West
18283	WY	West
18282	WY	West
18281	WY	West
18280	WY	West
18278	WV	East
18277	WV	East
18276	WV	East
18274	W	East
18273	Wv	East
18272	WV	East
18270	WV	East
18269	WV	East
18268	WV	East
18265	WV	East
18263	WV	East
18262	WV	East

-- 5. All regions and customers from those regions (even if no customer exists):

SELECT srm.region, dc.customer_id, dc.order_state

FROM state_region_mapping srm

RIGHT JOIN customers dc ON srm.order_state = dc.order_state;



-- 6. Top 5 customers by sales using a subquery:

SELECT * FROM (

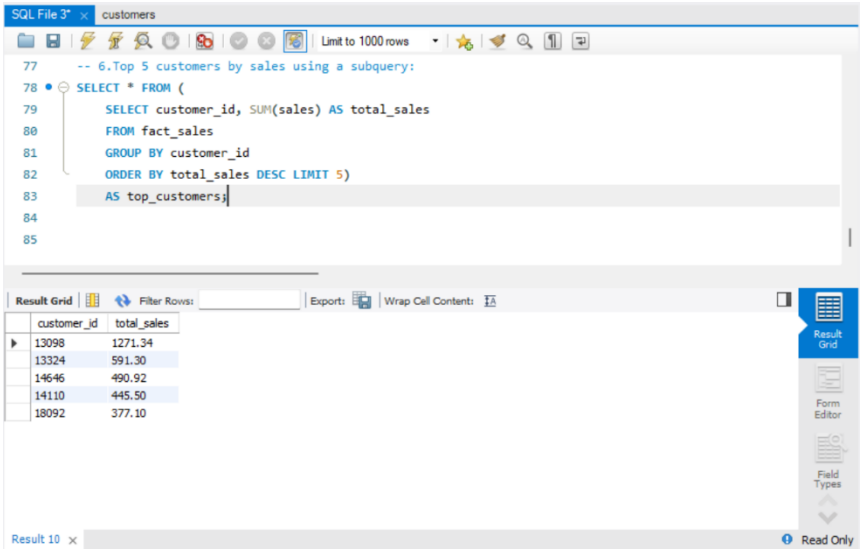
SELECT customer_id, SUM(sales) AS total_sales

FROM fact_sales

GROUP BY customer_id

ORDER BY total_sales DESC LIMIT 5)

AS top_customers;

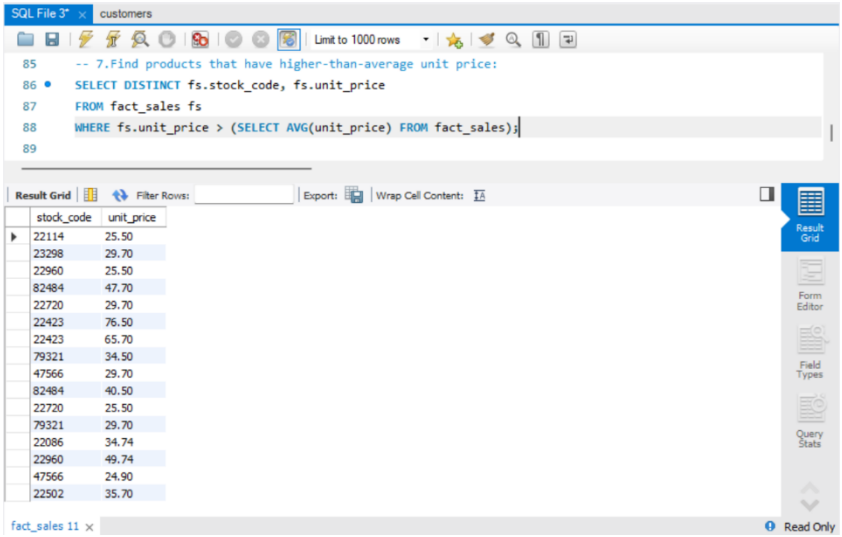


-- 7. Find products that have higher-than-average unit price:

SELECT DISTINCT fs.stock_code, fs.unit_price

FROM fact_sales fs

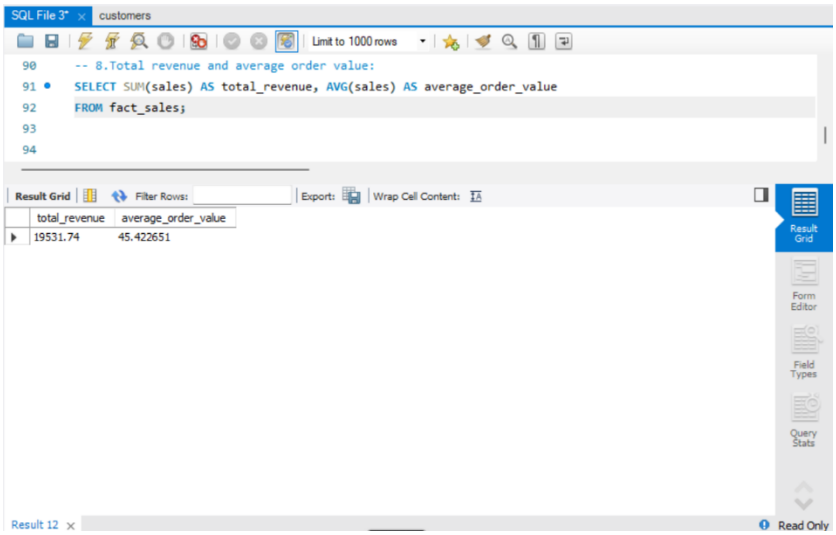
WHERE fs.unit_price > (SELECT AVG(unit_price) FROM fact_sales);



-- 8. Total revenue and average order value:

SELECT SUM(sales) AS total_revenue, AVG(sales) AS average_order_value

FROM fact_sales;



-- 9. Create a view for total sales per customer with region info:

CREATE VIEW customer_sales_region AS

SELECT fs.customer_id, SUM(fs.sales) AS total_sales, srm.region

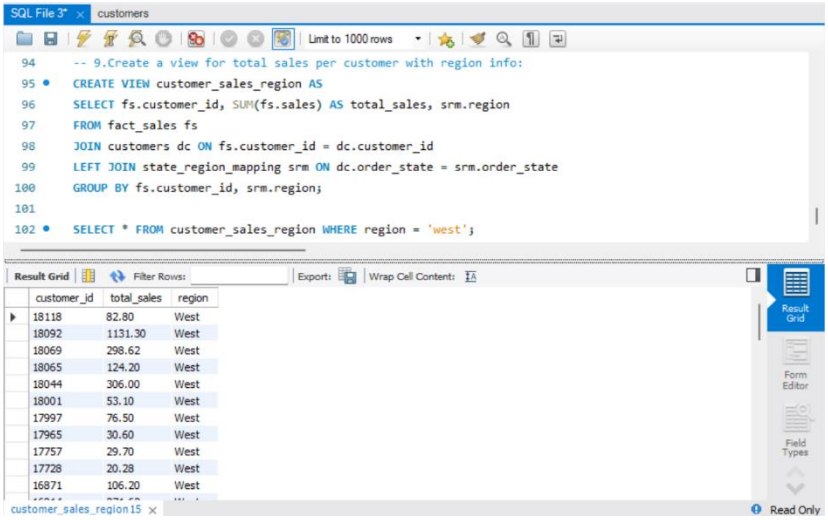
FROM fact_sales fs

JOIN customers dc ON fs.customer_id = dc.customer_id

LEFT JOIN state_region_mapping srm ON dc.order_state = srm.order_state

GROUP BY fs.customer_id, srm.region;

SELECT * FROM customer_sales_region WHERE region = 'west';



-- 10. Average quantity sold per product:

SELECT stock_code, AVG(quantity) AS avg_quantity

FROM fact_sales

GROUP BY stock_code;