THE GREAT PIZZA ANALYTICS CHALLENGE

IDC Pizza Analytics. SQL Case study

WELCOME TO IDC PIZZA ANALYTICS

Delivering insights from real sales, real customers, and real data— one slice at a time.

🍕 GOALS OF THE IDC PIZZA ANALYTICS CHALLENGE

To answer a series of questions using SQL, practicing the topics covered up to Day 15

- DATABASE CREATION & TABLE DESIGN
- FILTERING & OPERATORS (WHERE, IN, BETWEEN, LIKE, AND/OR/NOT)
- AGGREGATIONS (SUM, AVG, COUNT, MIN, MAX, GROUP BY, HAVING)
- JOINS (INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN, SELF JOIN)
- DATA CLEANING (DISTINCT, COALESCE, HANDLING NULLS)

ABOUT THE IDC PIZZA DATASET

The project is built on four relational tables, each capturing a different part of IDC Pizza’s operations:

- order_details – order_details_id, order_id, pizza_id, quantity.
- orders – order_id, date, time.
- pizza_types – pizza_type_id, name, category, ingredients
- pizzas – pizza_id, pizza_type_id, size, price

Together, these tables provide a complete and structured view of IDC Pizza’s menu, pricing, orders, and sales activity — enabling deep analysis using SQL.

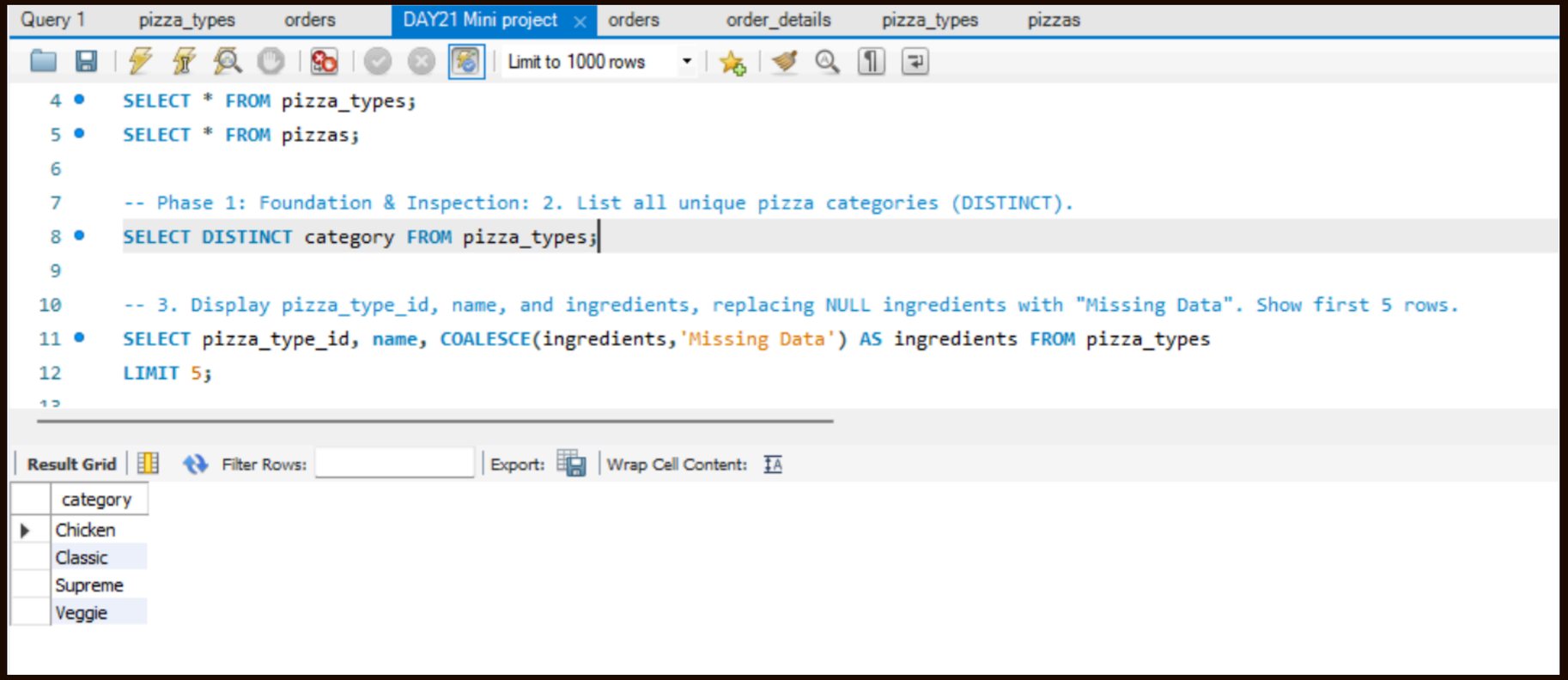
QUESTIONS

Phase 1: Foundation & Inspection

1. Install IDC_Pizza.dump as IDC_Pizza server

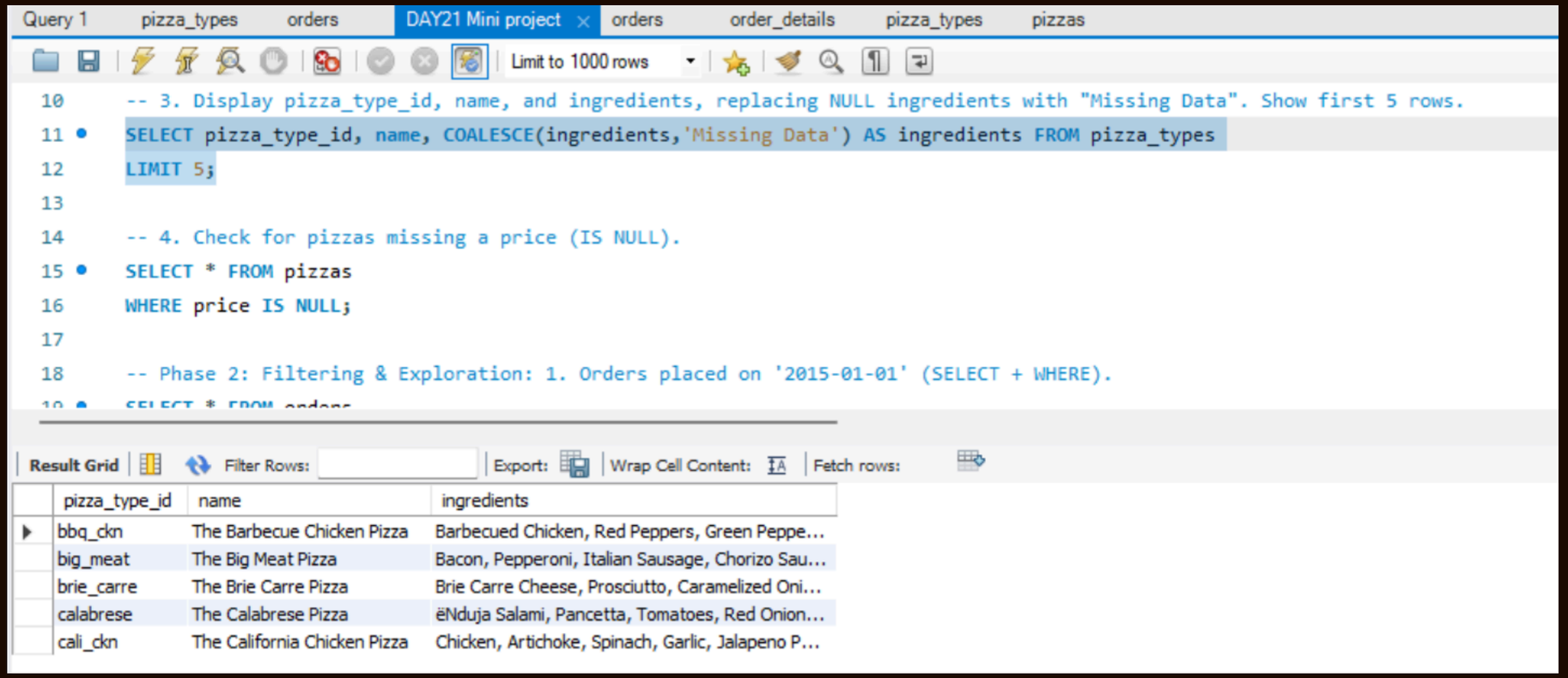
2. List all unique pizza categories (DISTINCT).

```
SELECT DISTINCT category FROM pizza_types;
```


Query 1
pizza_types
orders
DAY21 Mini project
orders
order_details
pizza_types
pizzas
Limit to 1000 rows
4 SELECT * FROM pizza_types;
5 SELECT * FROM pizzas;
6
7 -- Phase 1: Foundation & Inspection: 2. List all unique pizza categories (DISTINCT).
8 SELECT DISTINCT category FROM pizza_types;
9
10 -- 3. Display pizza_type_id, name, and ingredients, replacing NULL ingredients with "Missing Data". Show first 5 rows.
11 SELECT pizza_type_id, name, COALESCE(ingredients,'Missing Data') AS ingredients FROM pizza_types
12 LIMIT 5;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
category
Chicken
Classic
Supreme
Veggie

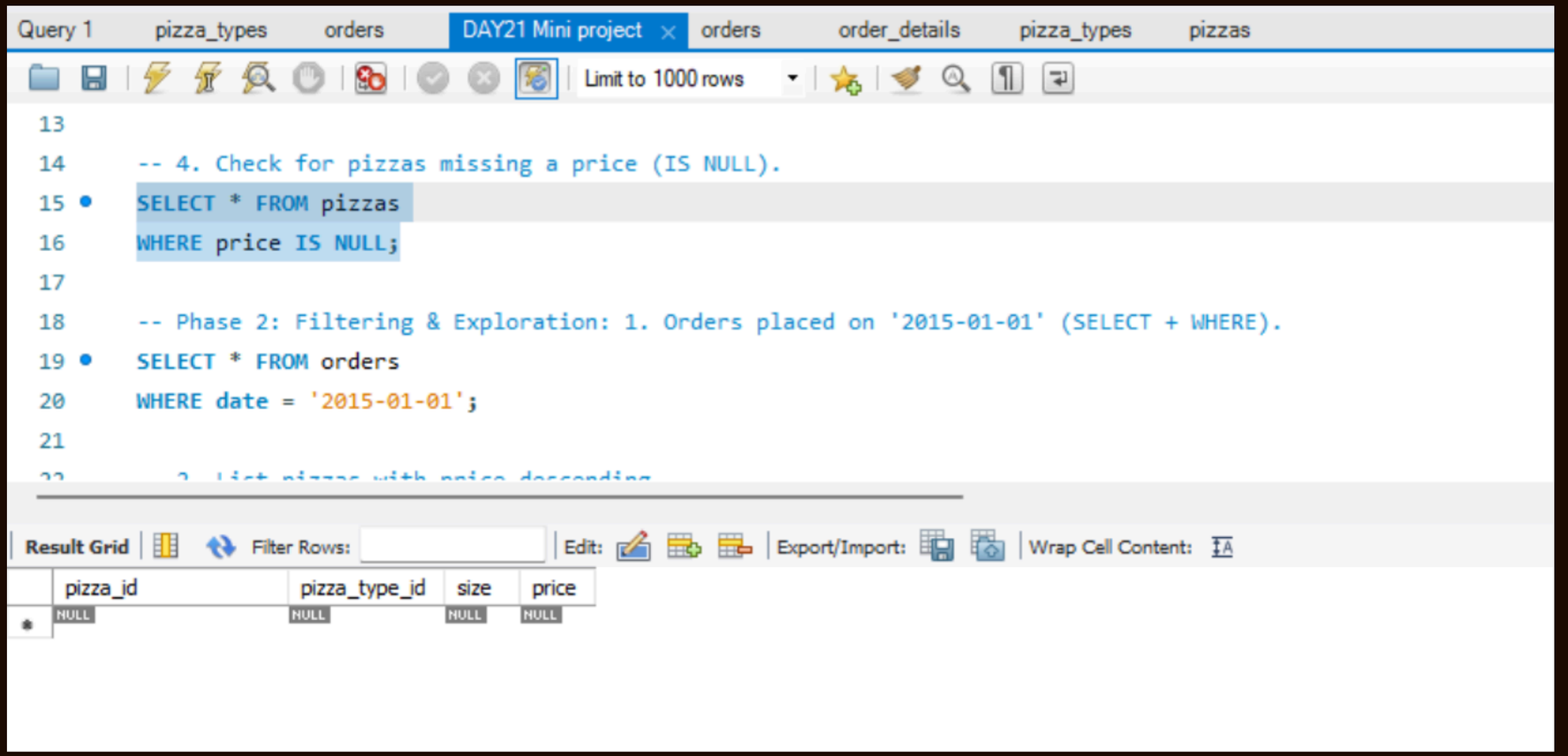
3. Display pizza_type_id,name, and ingredients, replacing NULL ingredients with "Missing Data". Show first 5 rows.

```
SELECT pizza_type_id, name,
COALESCE(ingredients,'Missing Data') AS ingredients
FROM pizza_types
LIMIT 5;
```


Query 1
pizza_types
orders
DAY21 Mini project
orders
order_details
pizza_types
pizzas
Limit to 1000 rows
10 -- 3. Display pizza_type_id, name, and ingredients, replacing NULL ingredients with "Missing Data". Show first 5 rows.
11 SELECT pizza_type_id, name, COALESCE(ingredients,'Missing Data') AS ingredients FROM pizza_types
12 LIMIT 5;
13
14 -- 4. Check for pizzas missing a price (IS NULL).
15 SELECT * FROM pizzas
16 WHERE price IS NULL;
17
18 -- Phase 2: Filtering & Exploration: 1. Orders placed on '2015-01-01' (SELECT + WHERE).
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Fetch rows:
pizza_type_id
name
ingredients
bbq_ckn
The Barbecue Chicken Pizza
Barbecued Chicken, Red Peppers, Green Peppe...
big_meat
The Big Meat Pizza
Bacon, Pepperoni, Italian Sausage, Chorizo Sau...
brie_carre
The Brie Carre Pizza
Brie Carre Cheese, Prosciutto, Caramelized Oni...
calabrese
The Calabrese Pizza
ëNduja Salami, Pancetta, Tomatoes, Red Onion...
cali_ckn
The California Chicken Pizza
Chicken, Artichoke, Spinach, Garlic, Jalapeno P...

4. Check for pizzas missing a price (IS NULL).

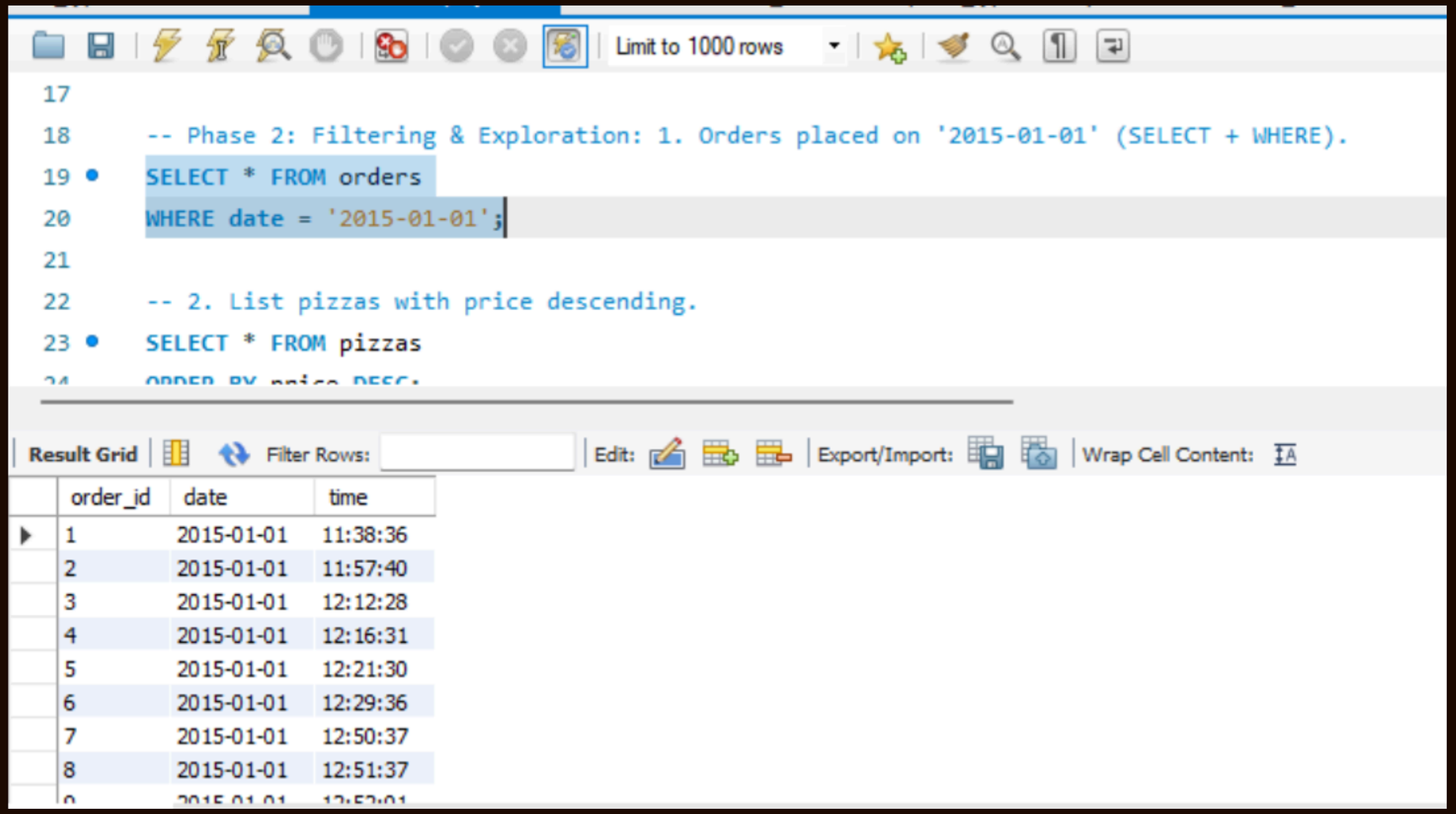
```
SELECT * FROM pizzas

WHERE price IS NULL;
```


Query 1
pizza_types
orders
DAY21 Mini project
orders
order_details
pizza_types
pizzas
Limit to 1000 rows
13
14 -- 4. Check for pizzas missing a price (IS NULL).
15 SELECT * FROM pizzas
16 WHERE price IS NULL;
17
18 -- Phase 2: Filtering & Exploration: 1. Orders placed on '2015-01-01' (SELECT + WHERE).
19 SELECT * FROM orders
20 WHERE date = '2015-01-01';
21
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
pizza_id
pizza_type_id
size
price
NULL
NULL
NULL
NULL

Phase 2: Filtering & Exploration

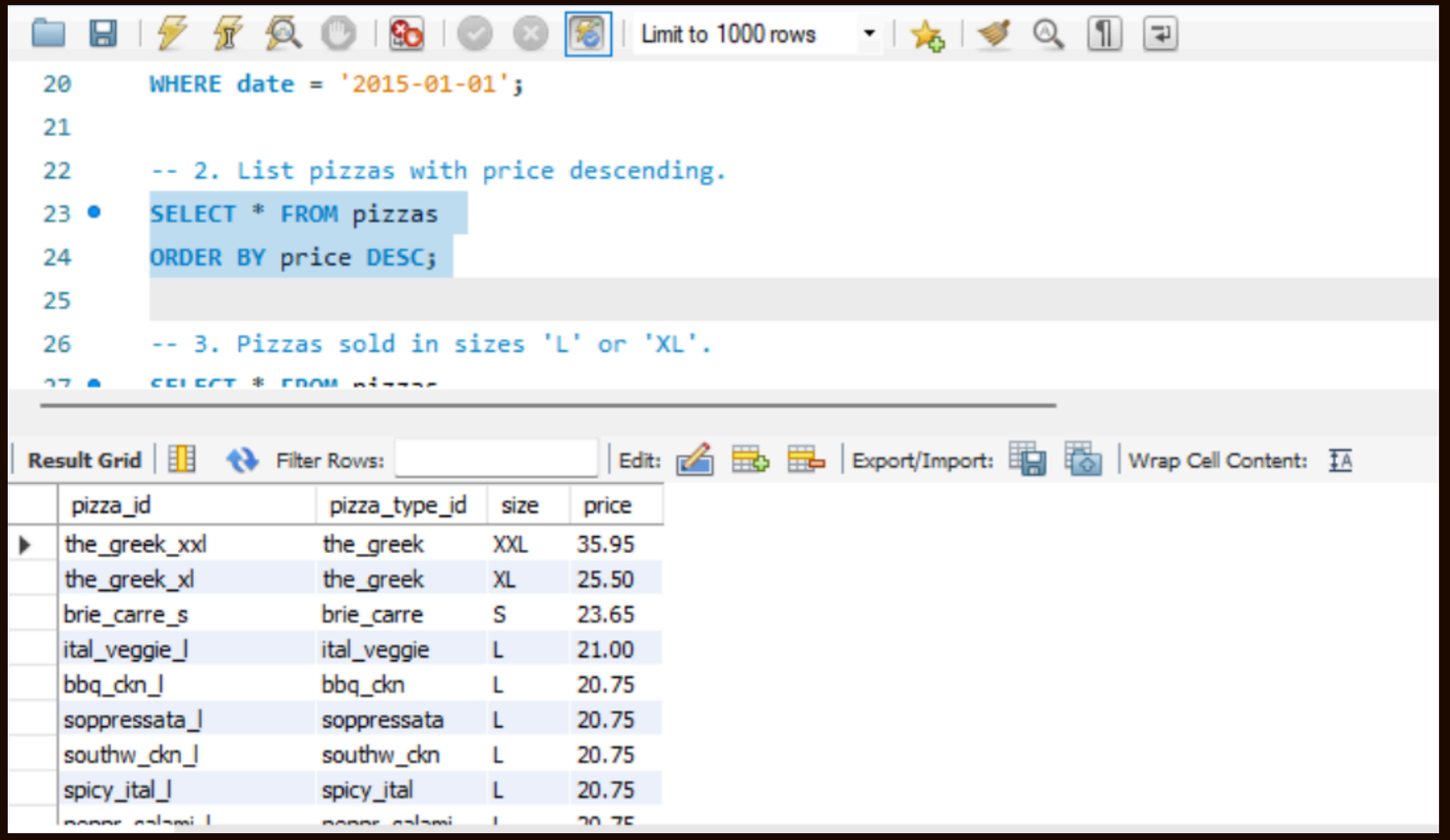
1. Orders placed on '2015-01-01' (SELECT+WHERE).

```
SELECT * FROM orders

WHERE date = '2015-01-01' ;
```


Limit to 1000 rows
17
18 -- Phase 2: Filtering & Exploration: 1. Orders placed on '2015-01-01' (SELECT + WHERE).
19 SELECT * FROM orders
20 WHERE date = '2015-01-01';
21
22 -- 2. List pizzas with price descending.
23 SELECT * FROM pizzas
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
order_id date time
1 2015-01-01 11:38:36
2 2015-01-01 11:57:40
3 2015-01-01 12:12:28
4 2015-01-01 12:16:31
5 2015-01-01 12:21:30
6 2015-01-01 12:29:36
7 2015-01-01 12:50:37
8 2015-01-01 12:51:37

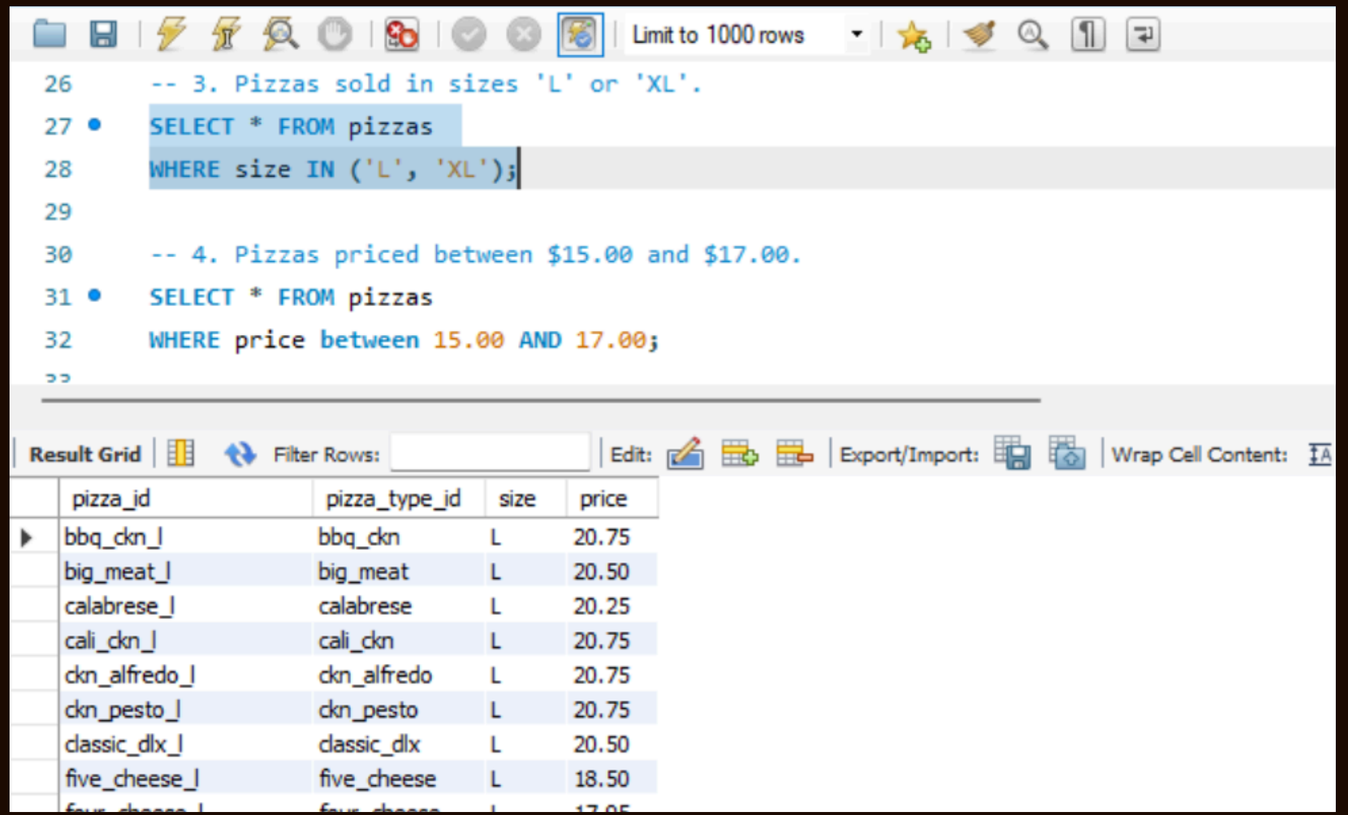
2. List pizzas with price descending.

```
SELECT * FROM pizzas

ORDER BY price DESC;
```


Limit to 1000 rows
20 WHERE date = '2015-01-01';
21
22 -- 2. List pizzas with price descending.
23 SELECT * FROM pizzas
24 ORDER BY price DESC;
25
26 -- 3. Pizzas sold in sizes 'L' or 'XL'.
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
pizza_id pizza_type_id size price
the_greek_xxl the_greek XXL 35.95
the_greek_xl the_greek XL 25.50
brie_carre_s brie_carre S 23.65
ital_veggie_l ital_veggie L 21.00
bbq_ckn_l bbq_ckn L 20.75
soppressata_l soppressata L 20.75
southw_ckn_l southw_ckn L 20.75
spicy_ital_l spicy_ital L 20.75

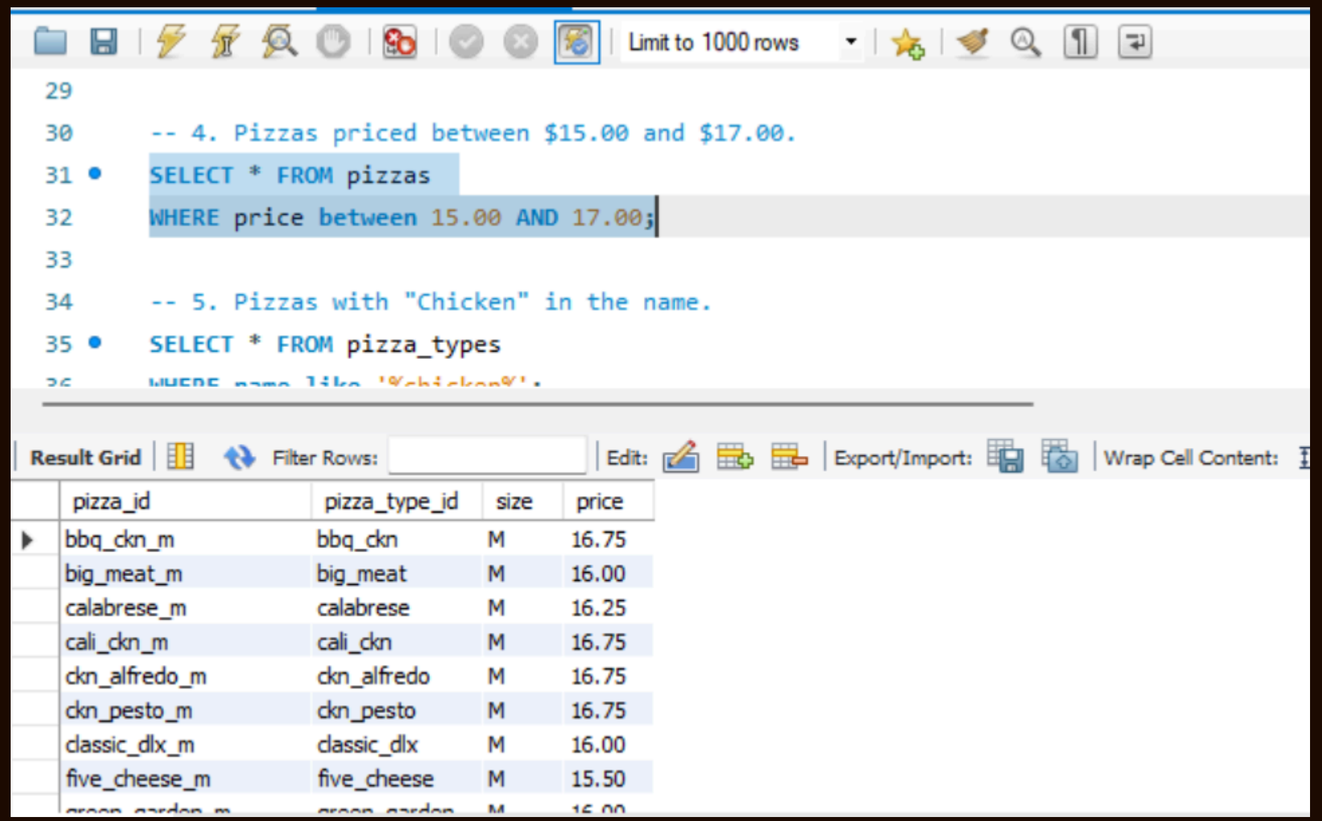
3. Pizzas sold in sizes 'L' or 'XL'.

```
SELECT * FROM pizzas

WHERE size IN ('L', 'XL');
```


Limit to 1000 rows
26 -- 3. Pizzas sold in sizes 'L' or 'XL'.
27 SELECT * FROM pizzas
28 WHERE size IN ('L', 'XL');
29
30 -- 4. Pizzas priced between $15.00 and $17.00.
31 SELECT * FROM pizzas
32 WHERE price between 15.00 AND 17.00;
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
pizza_id pizza_type_id size price
bbq_ckn_l bbq_ckn L 20.75
big_meat_l big_meat L 20.50
calabrese_l calabrese L 20.25
cali_ckn_l cali_ckn L 20.75
ckn_alfredo_l ckn_alfredo L 20.75
ckn_pesto_l ckn_pesto L 20.75
classic_dlx_l classic_dlx L 20.50
five_cheese_l five_cheese L 18.50

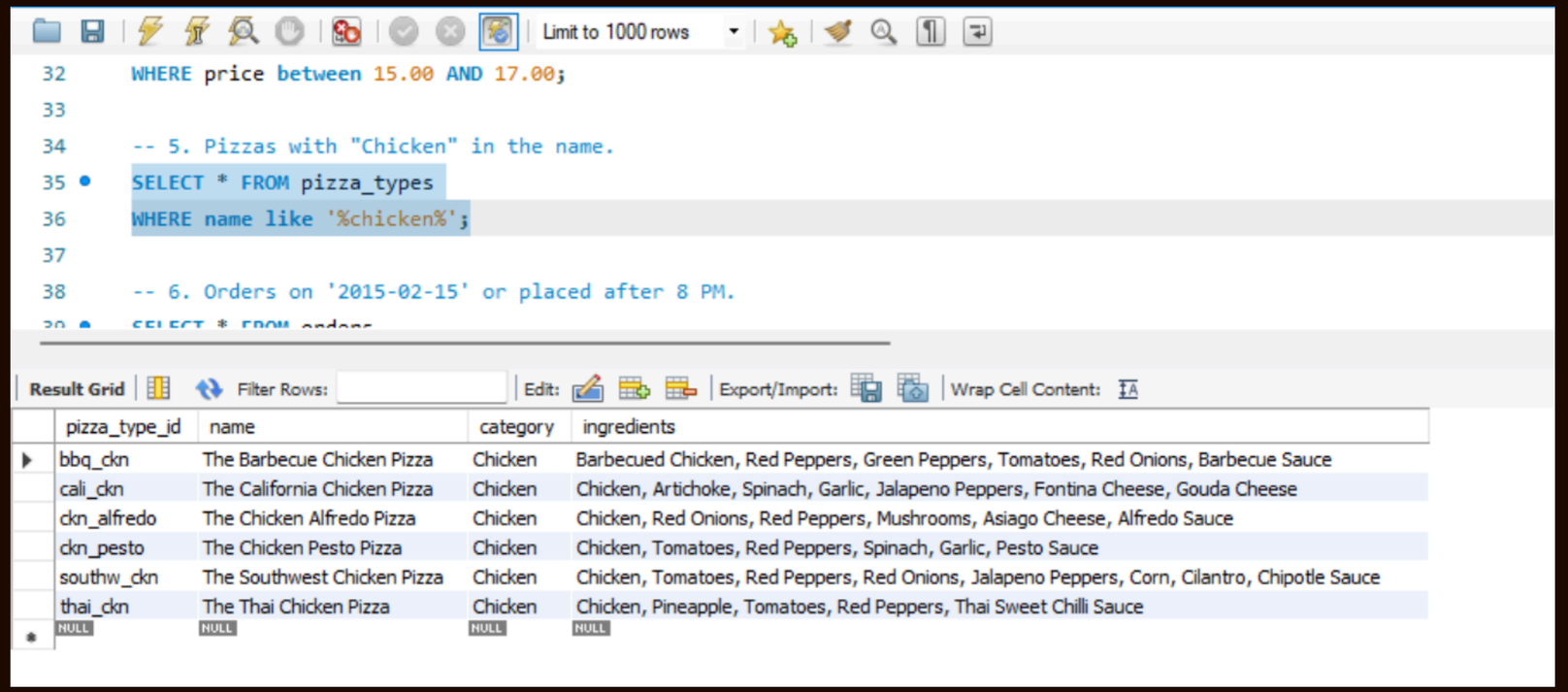
4. Pizzas priced between $15.00 and $17.00.

```
SELECT * FROM pizzas

WHERE price between 15.00 AND 17.00;
```


Limit to 1000 rows
29
30 -- 4. Pizzas priced between $15.00 and $17.00.
31 SELECT * FROM pizzas
32 WHERE price between 15.00 AND 17.00;
33
34 -- 5. Pizzas with "Chicken" in the name.
35 SELECT * FROM pizza_types
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
pizza_id | pizza_type_id | size | price
bbq_ckn_m | bbq_ckn | M | 16.75
big_meat_m | big_meat | M | 16.00
calabrese_m | calabrese | M | 16.25
cali_ckn_m | cali_ckn | M | 16.75
ckn_alfredo_m | ckn_alfredo | M | 16.75
ckn_pesto_m | ckn_pesto | M | 16.75
classic_dlx_m | classic_dlx | M | 16.00
five_cheese_m | five_cheese | M | 15.50

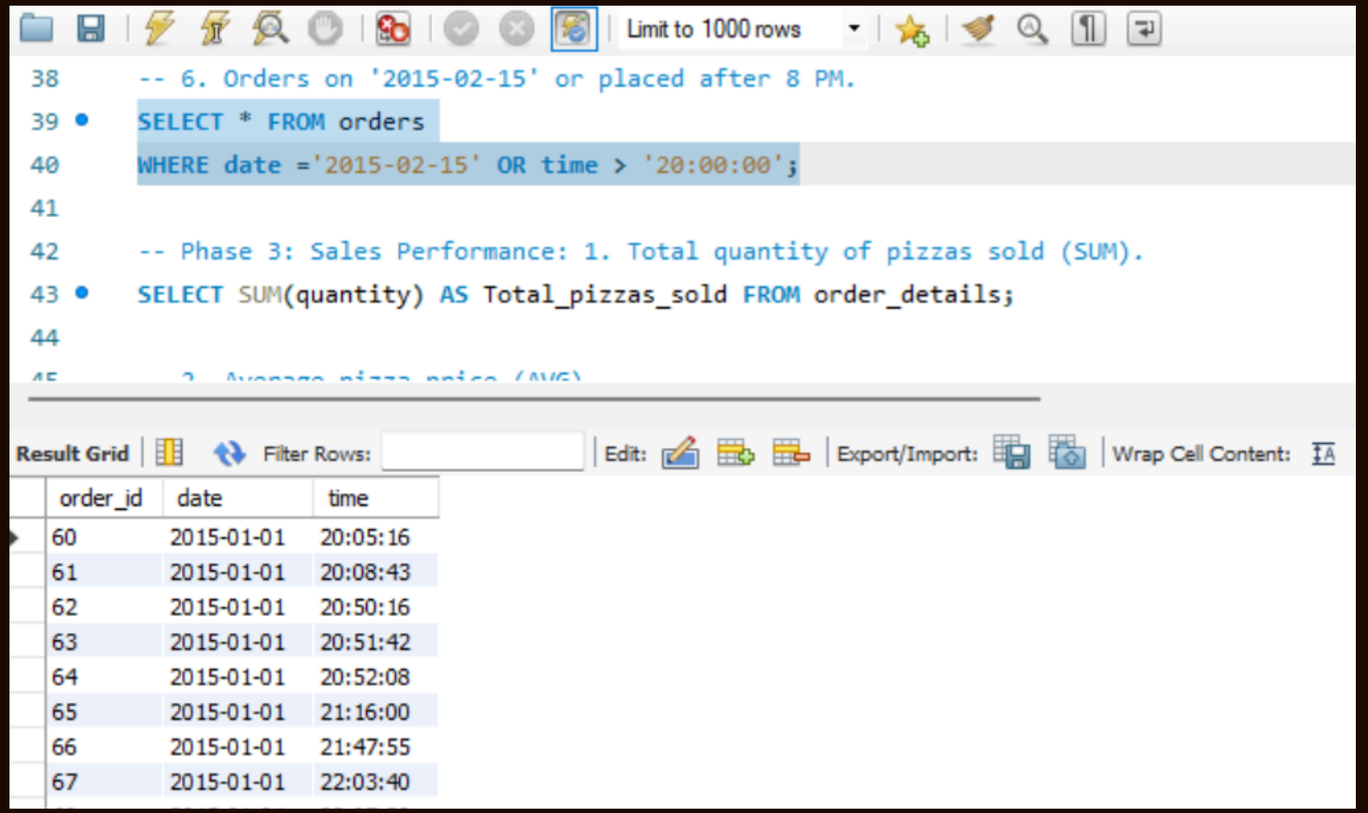
5. Pizzas with "Chicken" in the name.

```
SELECT * FROM pizza_types

WHERE name like '%chicken%' ;
```


Limit to 1000 rows
32 WHERE price between 15.00 AND 17.00;
33
34 -- 5. Pizzas with "Chicken" in the name.
35 SELECT * FROM pizza_types
36 WHERE name like '%chicken%';
37
38 -- 6. Orders on '2015-02-15' or placed after 8 PM.
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
pizza_type_id | name | category | ingredients
bbq_ckn | The Barbecue Chicken Pizza | Chicken | Barbecued Chicken, Red Peppers, Green Peppers, Tomatoes, Red Onions, Barbecue Sauce
cali_ckn | The California Chicken Pizza | Chicken | Chicken, Artichoke, Spinach, Garlic, Jalapeno Peppers, Fontina Cheese, Gouda Cheese
ckn_alfredo | The Chicken Alfredo Pizza | Chicken | Chicken, Red Onions, Red Peppers, Mushrooms, Asiago Cheese, Alfredo Sauce
ckn_pesto | The Chicken Pesto Pizza | Chicken | Chicken, Tomatoes, Red Peppers, Spinach, Garlic, Pesto Sauce
southw_ckn | The Southwest Chicken Pizza | Chicken | Chicken, Tomatoes, Red Peppers, Red Onions, Jalapeno Peppers, Corn, Cilantro, Chipotle Sauce
thai_ckn | The Thai Chicken Pizza | Chicken | Chicken, Pineapple, Tomatoes, Red Peppers, Thai Sweet Chilli Sauce
NULL NULL NULL NULL

6. Orders on '2015-02-15' or placed after 8 PM.

```
SELECT * FROM orders

WHERE date ='2015-02-15' OR time > '20:00:00' ;
```



```
38  -- 6. Orders on '2015-02-15' or placed after 8 PM.
39  SELECT * FROM orders
40  WHERE date ='2015-02-15' OR time > '20:00:00';
41
42  -- Phase 3: Sales Performance: 1. Total quantity of pizzas sold (SUM).
43  SELECT SUM(quantity) AS Total_pizzas_sold FROM order_details;
44
```

Limit to 1000 rows

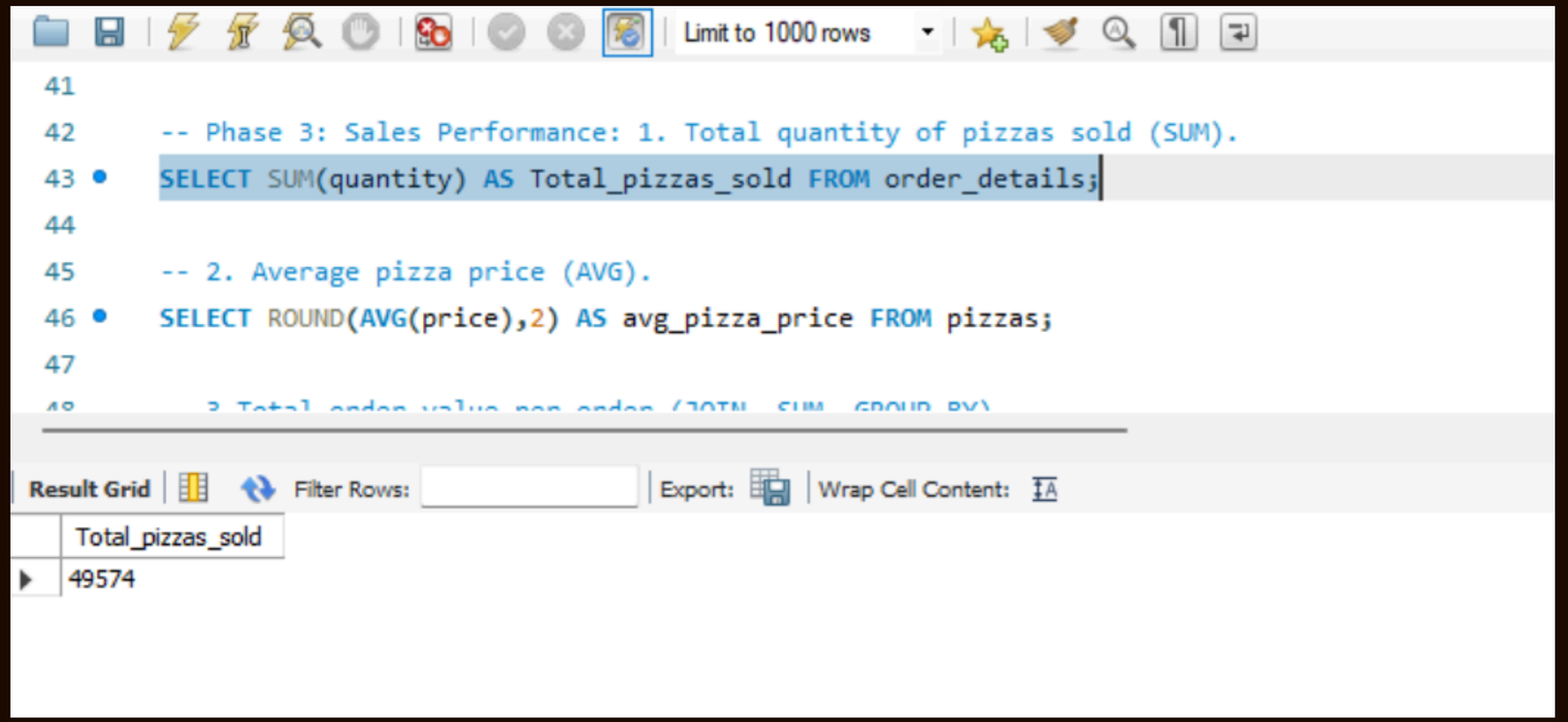
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

order_id	date	time
60	2015-01-01	20:05:16
61	2015-01-01	20:08:43
62	2015-01-01	20:50:16
63	2015-01-01	20:51:42
64	2015-01-01	20:52:08
65	2015-01-01	21:16:00
66	2015-01-01	21:47:55
67	2015-01-01	22:03:40

Phase 3: Sales Performance

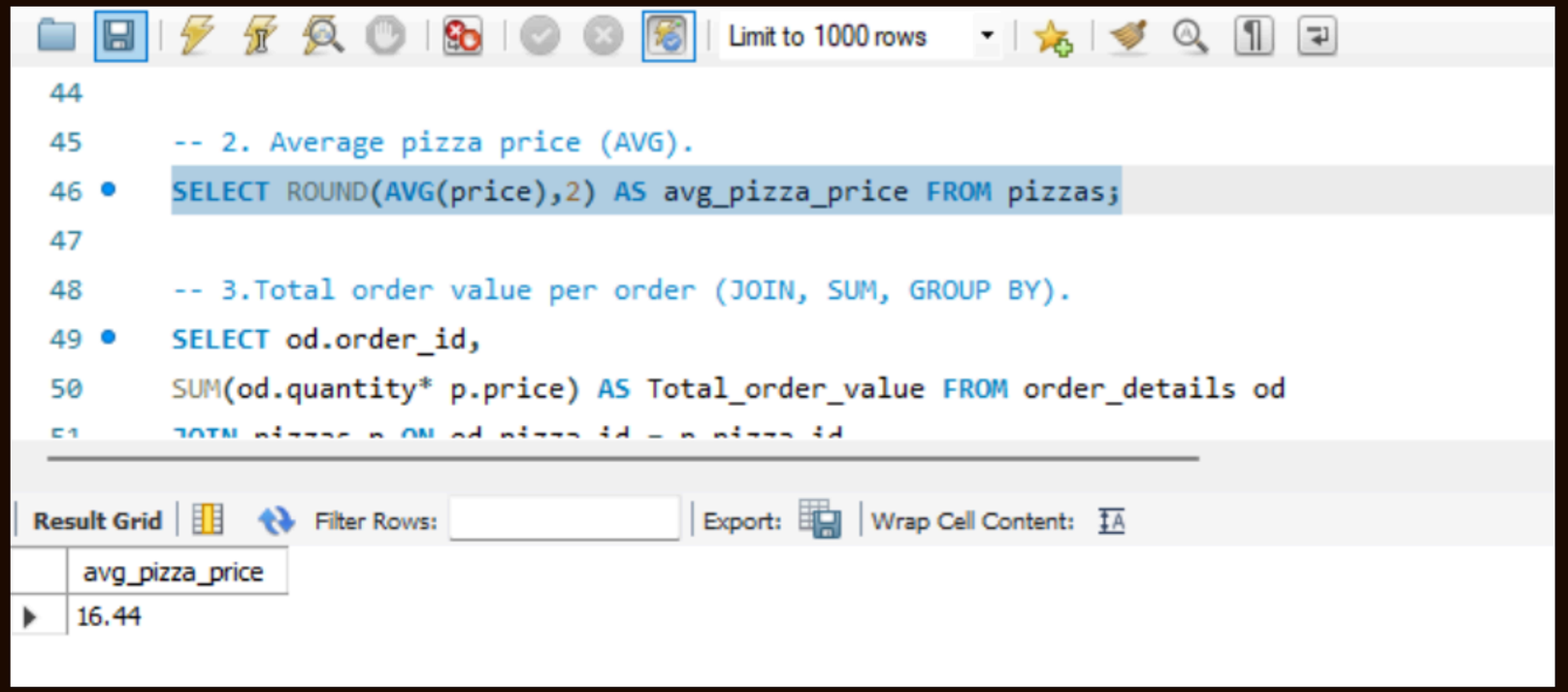
1. Total quantity of pizzas sold (SUM).

```
SELECT SUM(quantity) AS Total_pizzas_sold

FROM order_details;
```


Limit to 1000 rows
41
42 -- Phase 3: Sales Performance: 1. Total quantity of pizzas sold (SUM).
43 SELECT SUM(quantity) AS Total_pizzas_sold FROM order_details;
44
45 -- 2. Average pizza price (AVG).
46 SELECT ROUND(AVG(price),2) AS avg_pizza_price FROM pizzas;
47
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Total_pizzas_sold
49574

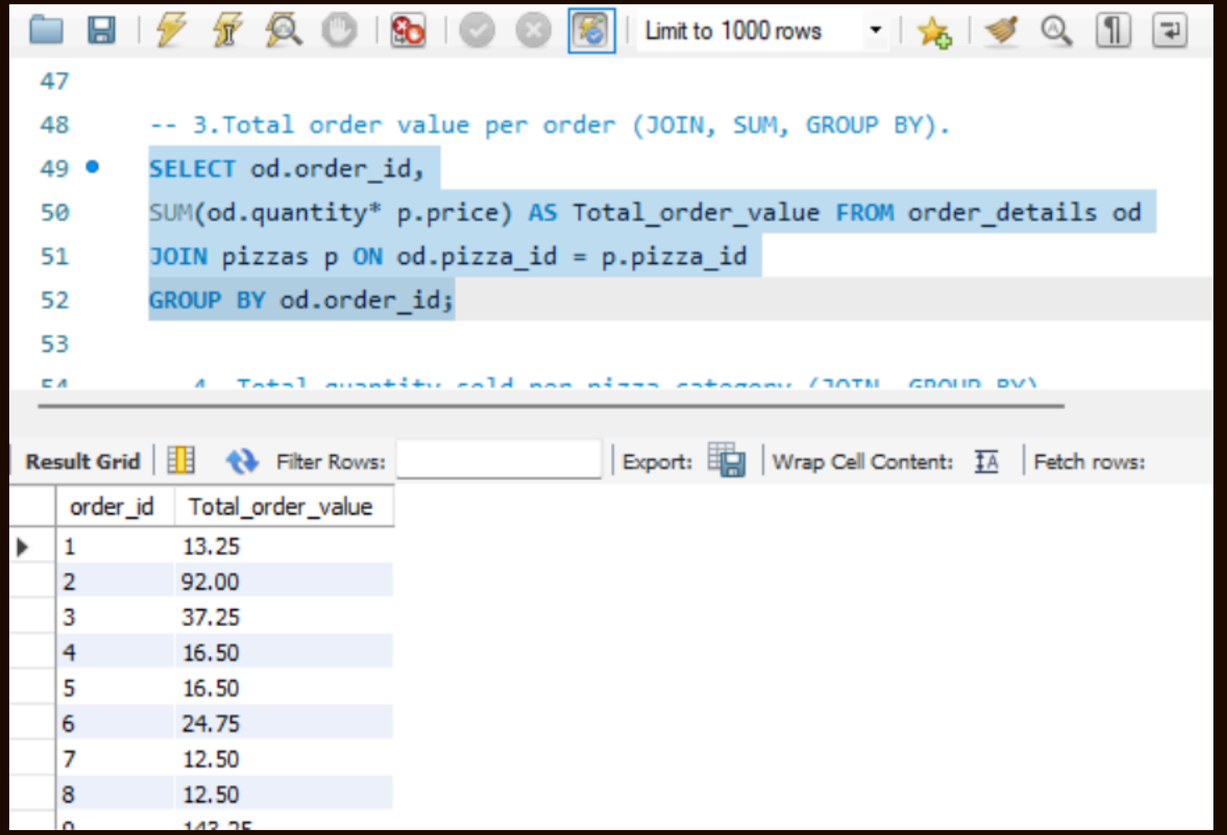
2. Average pizza price (AVG).

```
SELECT ROUND(AVG(price),2) AS avg_pizza_price

FROM pizzas;
```


Limit to 1000 rows
44
45 -- 2. Average pizza price (AVG).
46 SELECT ROUND(AVG(price),2) AS avg_pizza_price FROM pizzas;
47
48 -- 3.Total order value per order (JOIN, SUM, GROUP BY).
49 SELECT od.order_id,
50 SUM(od.quantity* p.price) AS Total_order_value FROM order_details od
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
avg_pizza_price
16.44

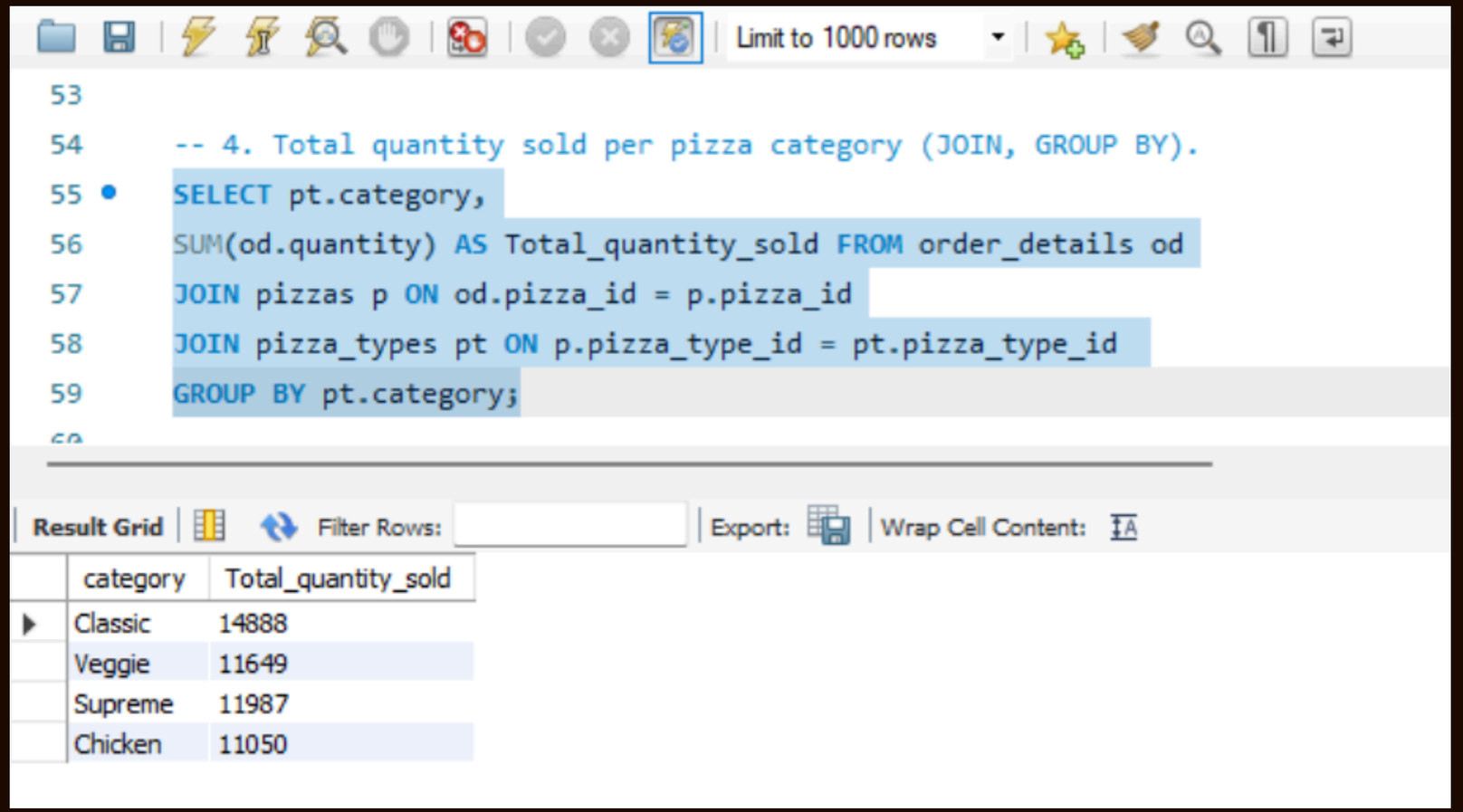
3.Total order value per order (JOIN, SUM, GROUP BY).

```
SELECT od.order_id,
SUM(od.quantity*  p.price)  AS  Total_order_value  FROM
order_details od
JOIN pizzas p ON od.pizza_id = p.pizza_id
GROUP BY od.order_id;
```


Limit to 1000 rows
47
48 -- 3.Total order value per order (JOIN, SUM, GROUP BY).
49 SELECT od.order_id,
50 SUM(od.quantity* p.price) AS Total_order_value FROM order_details od
51 JOIN pizzas p ON od.pizza_id = p.pizza_id
52 GROUP BY od.order_id;
53
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Fetch rows:
order_id
Total_order_value
1 13.25
2 92.00
3 37.25
4 16.50
5 16.50
6 24.75
7 12.50
8 12.50

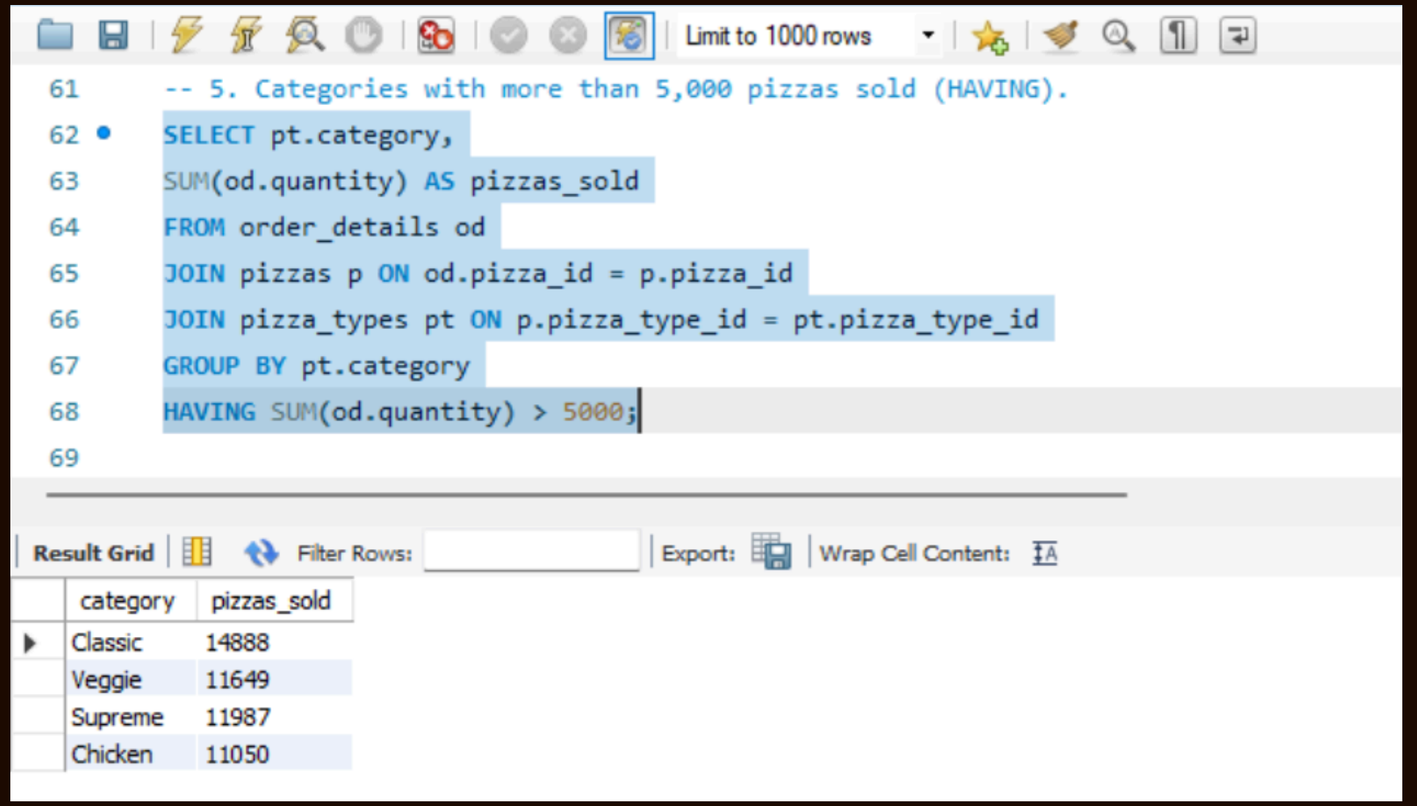
4. Total quantity sold per pizza category (JOIN, GROUP BY).

```
SELECT pt.category,
SUM(od.quantity) AS Total_quantity_sold FROM order_details od
JOIN pizzas p ON od.pizza_id = p.pizza_id
JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.category;
```


Limit to 1000 rows
53
54 -- 4. Total quantity sold per pizza category (JOIN, GROUP BY).
55 SELECT pt.category,
56 SUM(od.quantity) AS Total_quantity_sold FROM order_details od
57 JOIN pizzas p ON od.pizza_id = p.pizza_id
58 JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
59 GROUP BY pt.category;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
category
Total_quantity_sold
Classic 14888
Veggie 11649
Supreme 11987
Chicken 11050

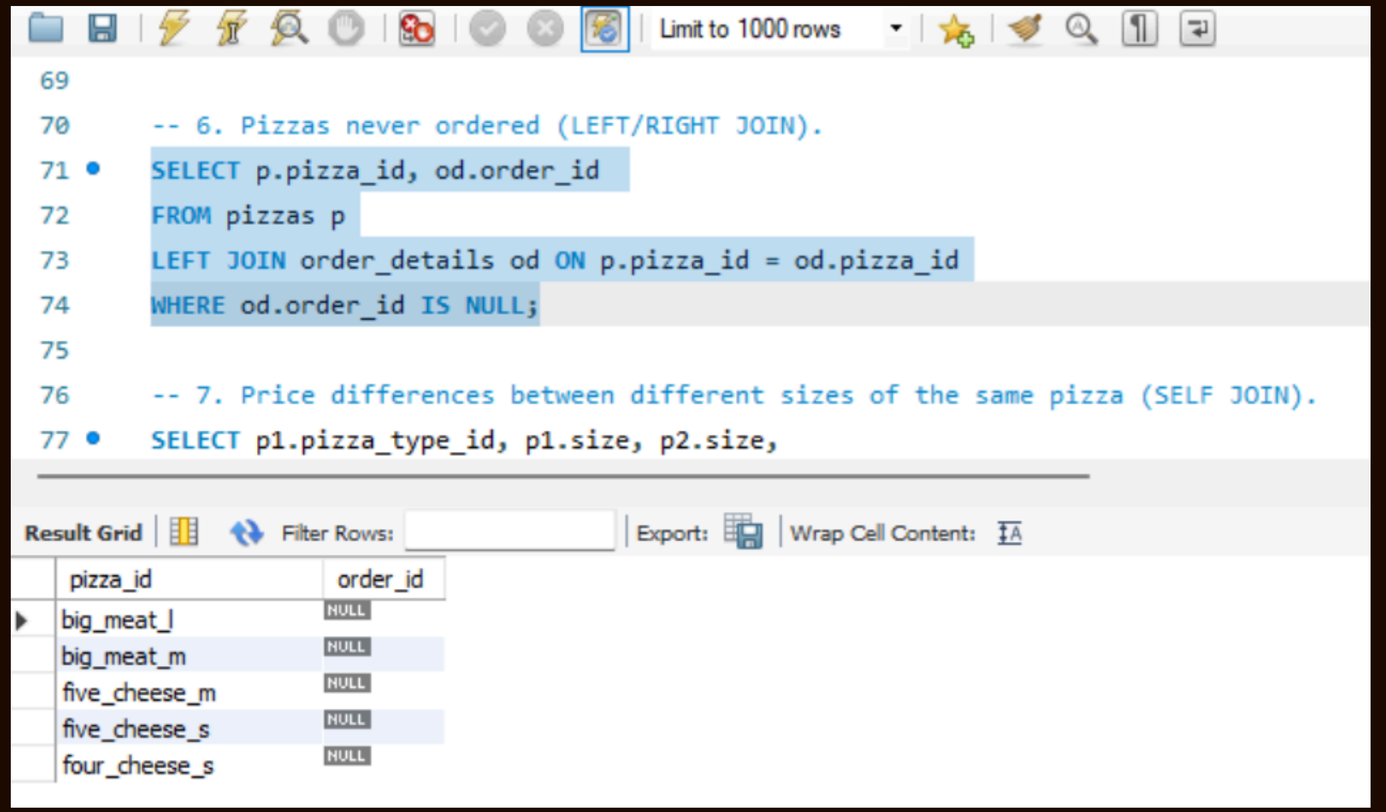
5. Categories with more than 5,000 pizzas sold (HAVING).

```
SELECT pt.category,
SUM(od.quantity) AS pizzas_sold
FROM order_details od
JOIN pizzas p ON od.pizza_id = p.pizza_id
JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.category
HAVING SUM(od.quantity) > 5000;
```


Limit to 1000 rows
61 -- 5. Categories with more than 5,000 pizzas sold (HAVING).
62 SELECT pt.category,
63 SUM(od.quantity) AS pizzas_sold
64 FROM order_details od
65 JOIN pizzas p ON od.pizza_id = p.pizza_id
66 JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
67 GROUP BY pt.category
68 HAVING SUM(od.quantity) > 5000;
69
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
category pizzas_sold
Classic 14888
Veggie 11649
Supreme 11987
Chicken 11050

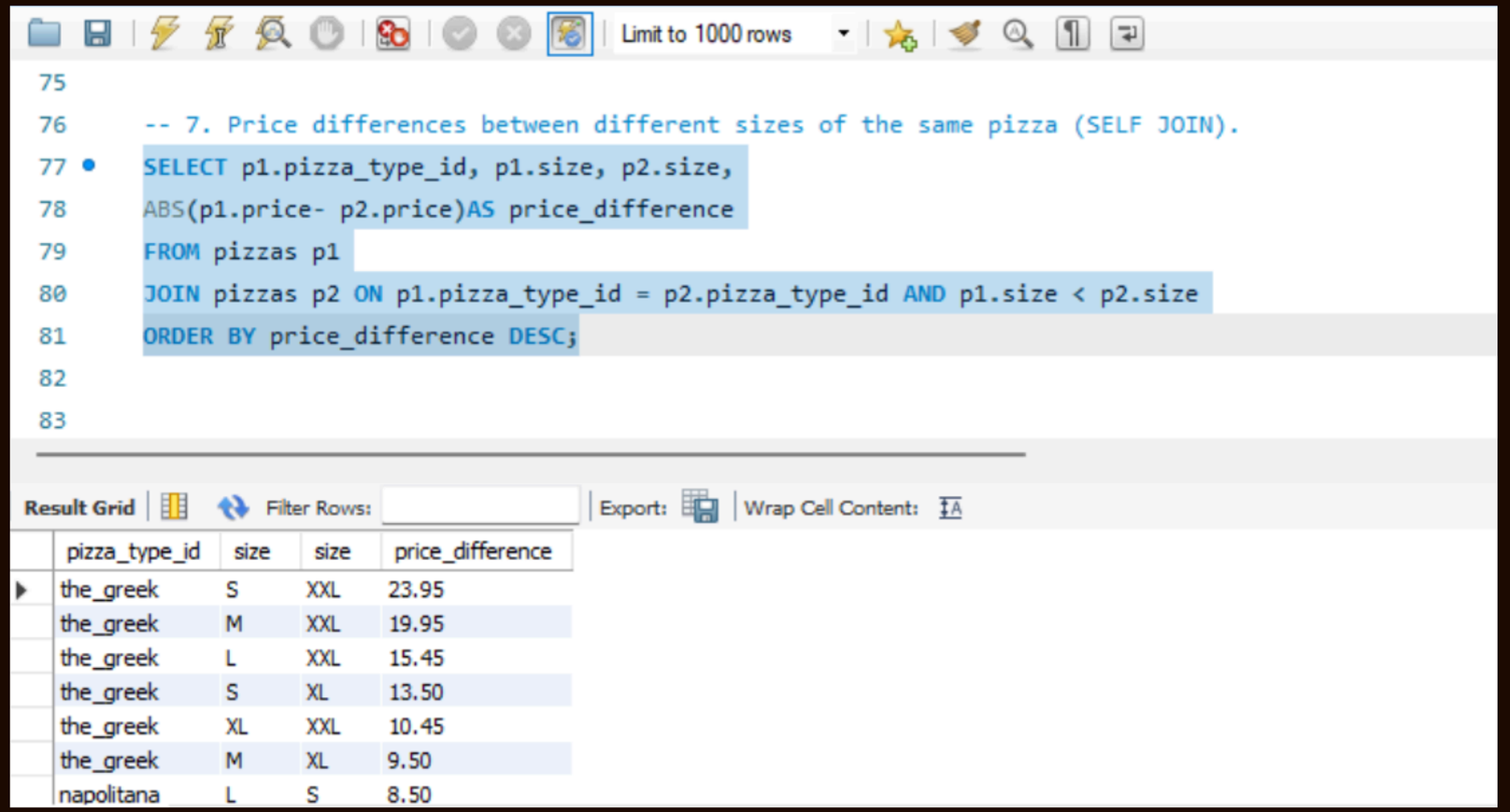
6. Pizzas never ordered (LEFT/RIGHT JOIN).

```
SELECT p.pizza_id, od.order_id
FROM pizzas p
LEFT JOIN order_details od ON p.pizza_id = od.pizza_id
WHERE od.order_id IS NULL;
```


Limit to 1000 rows
69
70 -- 6. Pizzas never ordered (LEFT/RIGHT JOIN).
71 SELECT p.pizza_id, od.order_id
72 FROM pizzas p
73 LEFT JOIN order_details od ON p.pizza_id = od.pizza_id
74 WHERE od.order_id IS NULL;
75
76 -- 7. Price differences between different sizes of the same pizza (SELF JOIN).
77 SELECT p1.pizza_type_id, p1.size, p2.size,
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
pizza_id
order_id
big_meat_l
big_meat_m
five_cheese_m
five_cheese_s
four_cheese_s
NULL

7. Price differences between different sizes of the same pizza (SELF JOIN).

```
SELECT p1.pizza_type_id, p1.size, p2.size,
ABS(p1.price- p2.price) AS price_difference
FROM pizzas p1
JOIN pizzas p2 ON p1.pizza_type_id = p2.pizza_type_id
AND p1.size < p2.size
ORDER BY price_difference DESC;
```


Limit to 1000 rows
75
76 -- 7. Price differences between different sizes of the same pizza (SELF JOIN).
77 SELECT p1.pizza_type_id, p1.size, p2.size,
78 ABS(p1.price- p2.price)AS price_difference
79 FROM pizzas p1
80 JOIN pizzas p2 ON p1.pizza_type_id = p2.pizza_type_id AND p1.size < p2.size
81 ORDER BY price_difference DESC;
82
83
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
pizza_type_id size size price_difference
the_greek S XXL 23.95
the_greek M XXL 19.95
the_greek L XXL 15.45
the_greek S XL 13.50
the_greek XL XXL 10.45
the_greek M XL 9.50
napolitana L S 8.50