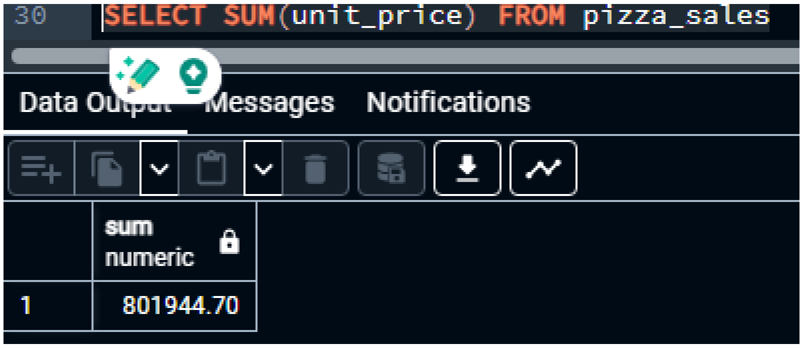
The Pizza Store Sales Power BI Dashboard

We have a dataset from a pizza store. We will analyze this dataset using SQL on Postgres Server. And after drawing insights we will use Microsoft Power BI to visualize our findings.

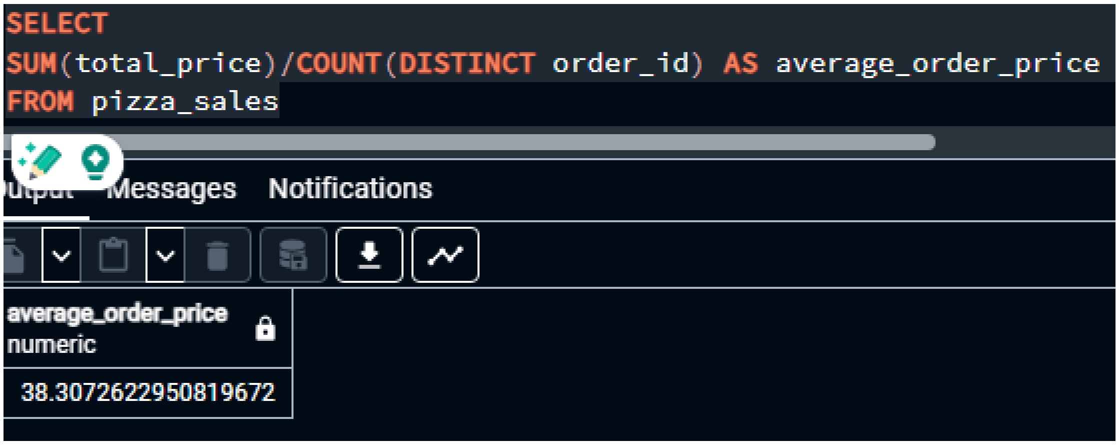
1. Total Revenue: The sum of the total price of all pizza orders.



```sql
SELECT SUM(unit_price) FROM pizza_sales
```

```
  sum
  numeric
1 801944.70
```

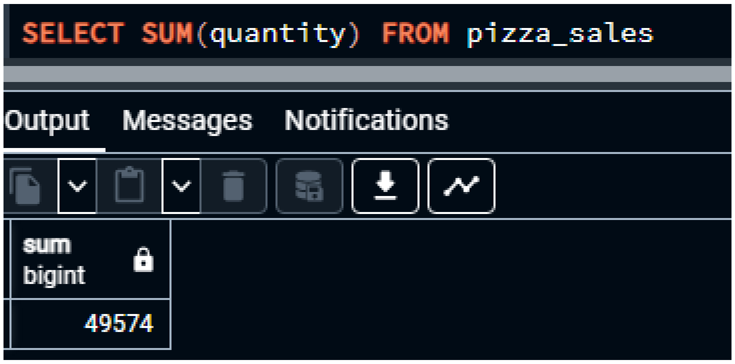
2. Average Order Value: The average amount spent per order, calculated by dividing the total revenue by the total number of orders.



```sql
SELECT
SUM(total_price)/COUNT(DISTINCT order_id) AS average_order_price
FROM pizza_sales
```

```
average_order_price
numeric
38.3072622950819672
```

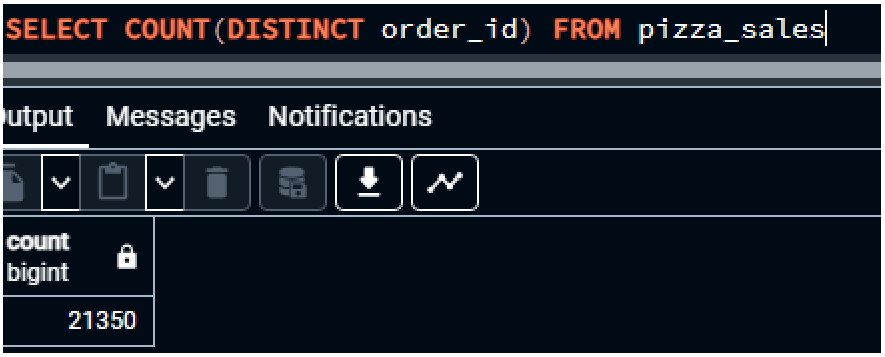
3. Total Pizzas Sold: The sum of the quantities of all pizzas sold.



```
SELECT SUM(quantity) FROM pizza_sales
```

```
sum
bigint
49574
```

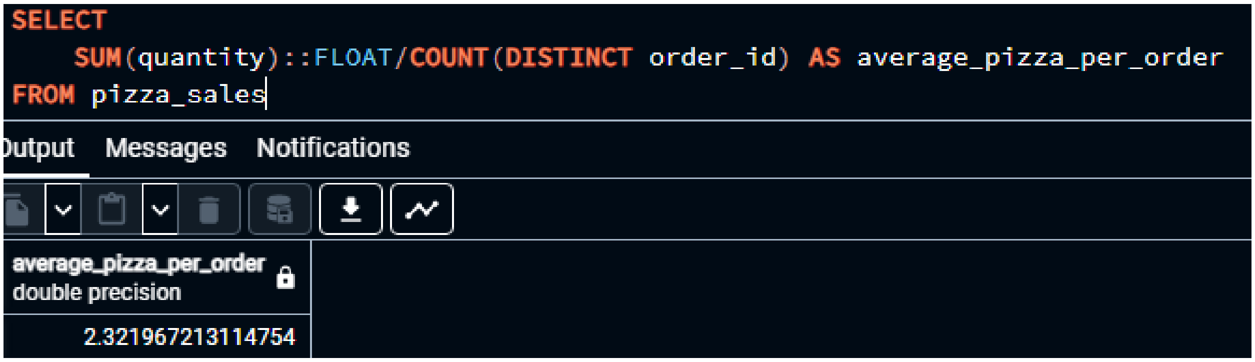
4. Total Orders: The total number of orders placed.



```
SELECT COUNT(DISTINCT order_id) FROM pizza_sales
```

```
count
bigint
21350
```

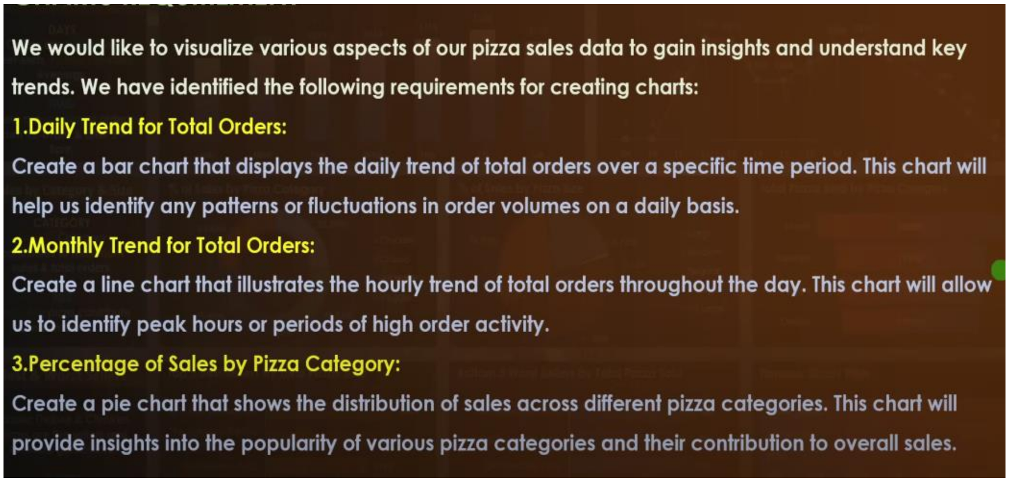
5. Average Pizzas Per Order: The average number of pizzas sold per order, calculated by dividing the total number of pizzas sold by the total number of orders.



```
SELECT
    SUM(quantity)::FLOAT/COUNT(DISTINCT order_id) AS average_pizza_per_order
FROM pizza_sales
```

```
average_pizza_per_order
double precision
2.321967213114754
```

Let's fire some queries to get values for drawing Charts:



We would like to visualize various aspects of our pizza sales data to gain insights and understand key trends. We have identified the following requirements for creating charts:

1.Daily Trend for Total Orders:

Create a bar chart that displays the daily trend of total orders over a specific time period. This chart will help us identify any patterns or fluctuations in order volumes on a daily basis.

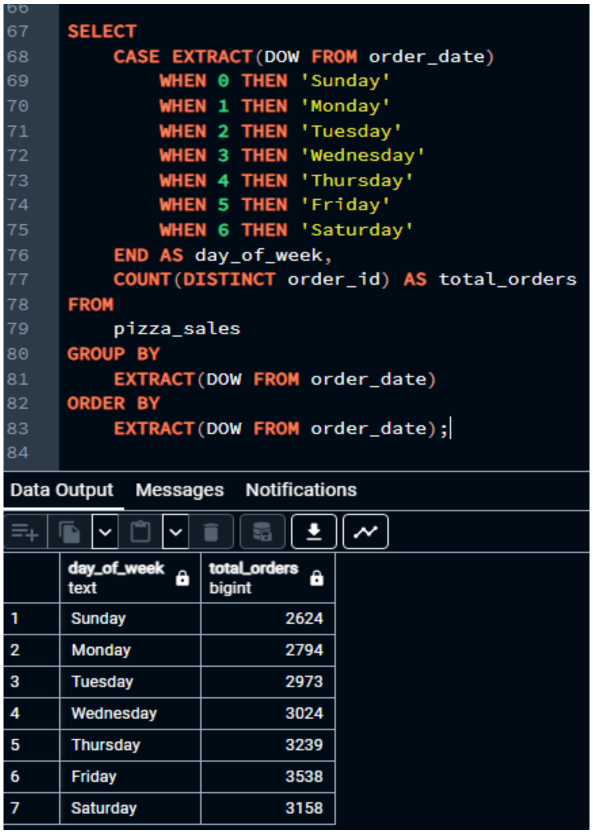
2.Monthly Trend for Total Orders:

Create a line chart that illustrates the hourly trend of total orders throughout the day. This chart will allow us to identify peak hours or periods of high order activity.

3.Percentage of Sales by Pizza Category:

Create a pie chart that shows the distribution of sales across different pizza categories. This chart will provide insights into the popularity of various pizza categories and their contribution to overall sales.

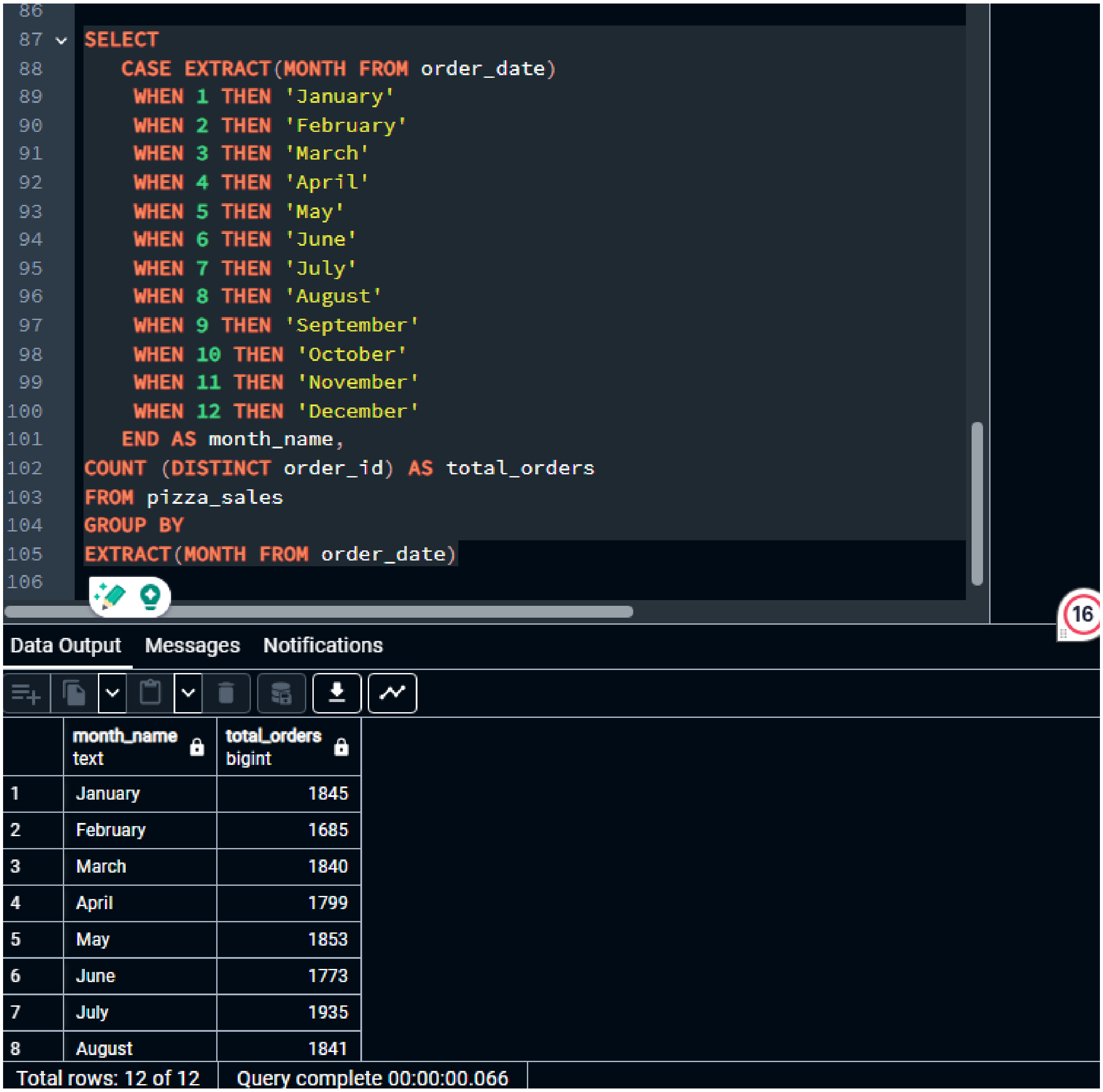
6.Daily trend for total orders-



```sql
SELECT
    CASE EXTRACT(DOW FROM order_date)
        WHEN 0 THEN 'Sunday'
        WHEN 1 THEN 'Monday'
        WHEN 2 THEN 'Tuesday'
        WHEN 3 THEN 'Wednesday'
        WHEN 4 THEN 'Thursday'
        WHEN 5 THEN 'Friday'
        WHEN 6 THEN 'Saturday'
    END AS day_of_week,
    COUNT(DISTINCT order_id) AS total_orders
FROM
    pizza_sales
GROUP BY
    EXTRACT(DOW FROM order_date)
ORDER BY
    EXTRACT(DOW FROM order_date);
```

	day_of_week text	total_orders bigint
1	Sunday	2624
2	Monday	2794
3	Tuesday	2973
4	Wednesday	3024
5	Thursday	3239
6	Friday	3538
7	Saturday	3158

7.Now let's see how the number of orders vary for every month in a year-

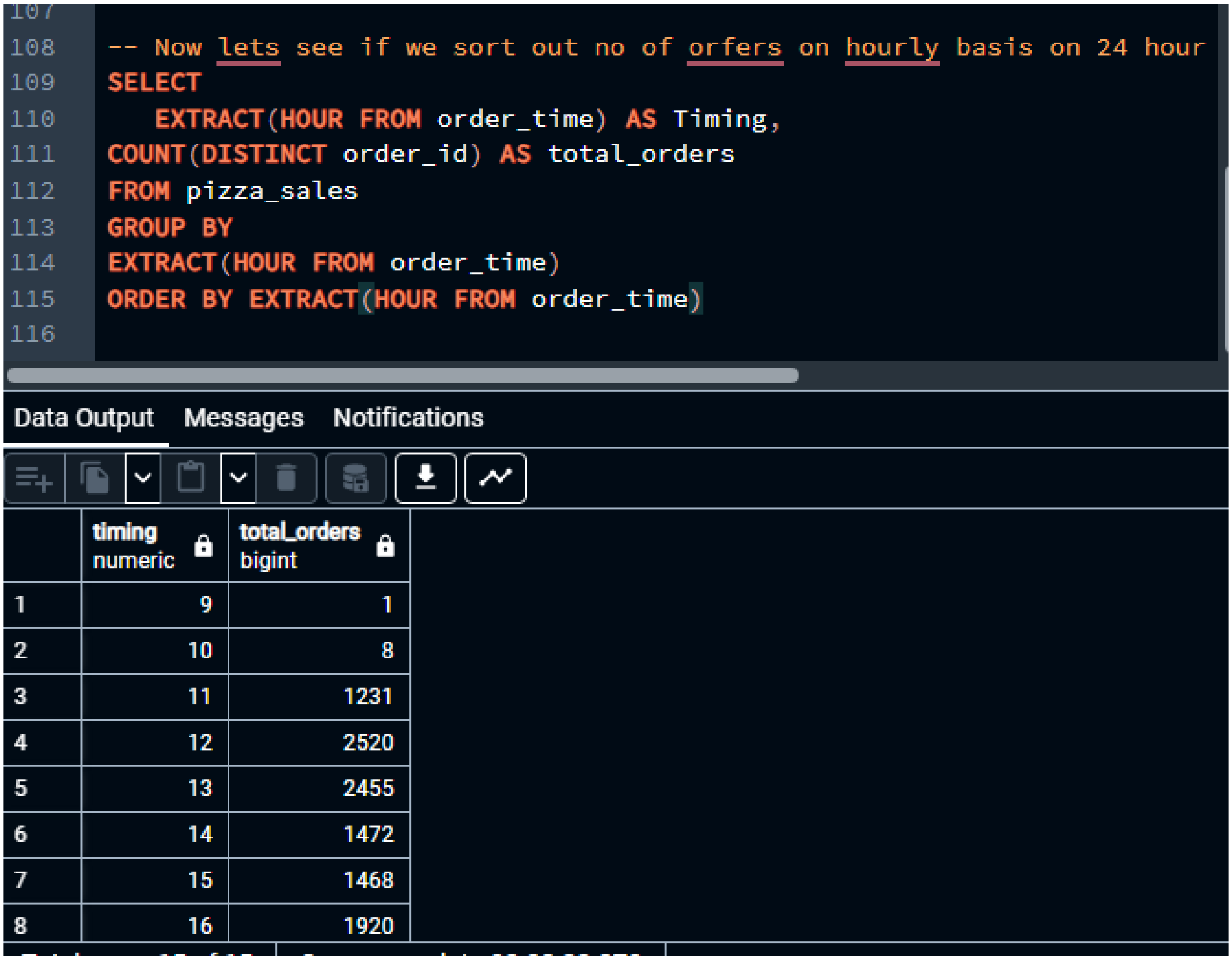


```sql
SELECT
   CASE EXTRACT(MONTH FROM order_date)
    WHEN 1 THEN 'January'
    WHEN 2 THEN 'February'
    WHEN 3 THEN 'March'
    WHEN 4 THEN 'April'
    WHEN 5 THEN 'May'
    WHEN 6 THEN 'June'
    WHEN 7 THEN 'July'
    WHEN 8 THEN 'August'
    WHEN 9 THEN 'September'
    WHEN 10 THEN 'October'
    WHEN 11 THEN 'November'
    WHEN 12 THEN 'December'
   END AS month_name,
COUNT (DISTINCT order_id) AS total_orders
FROM pizza_sales
GROUP BY
EXTRACT(MONTH FROM order_date)
```

	month_name text	total_orders bigint
1	January	1845
2	February	1685
3	March	1840
4	April	1799
5	May	1853
6	June	1773
7	July	1935
8	August	1841

Total rows: 12 of 12   Query complete 00:00:00.066

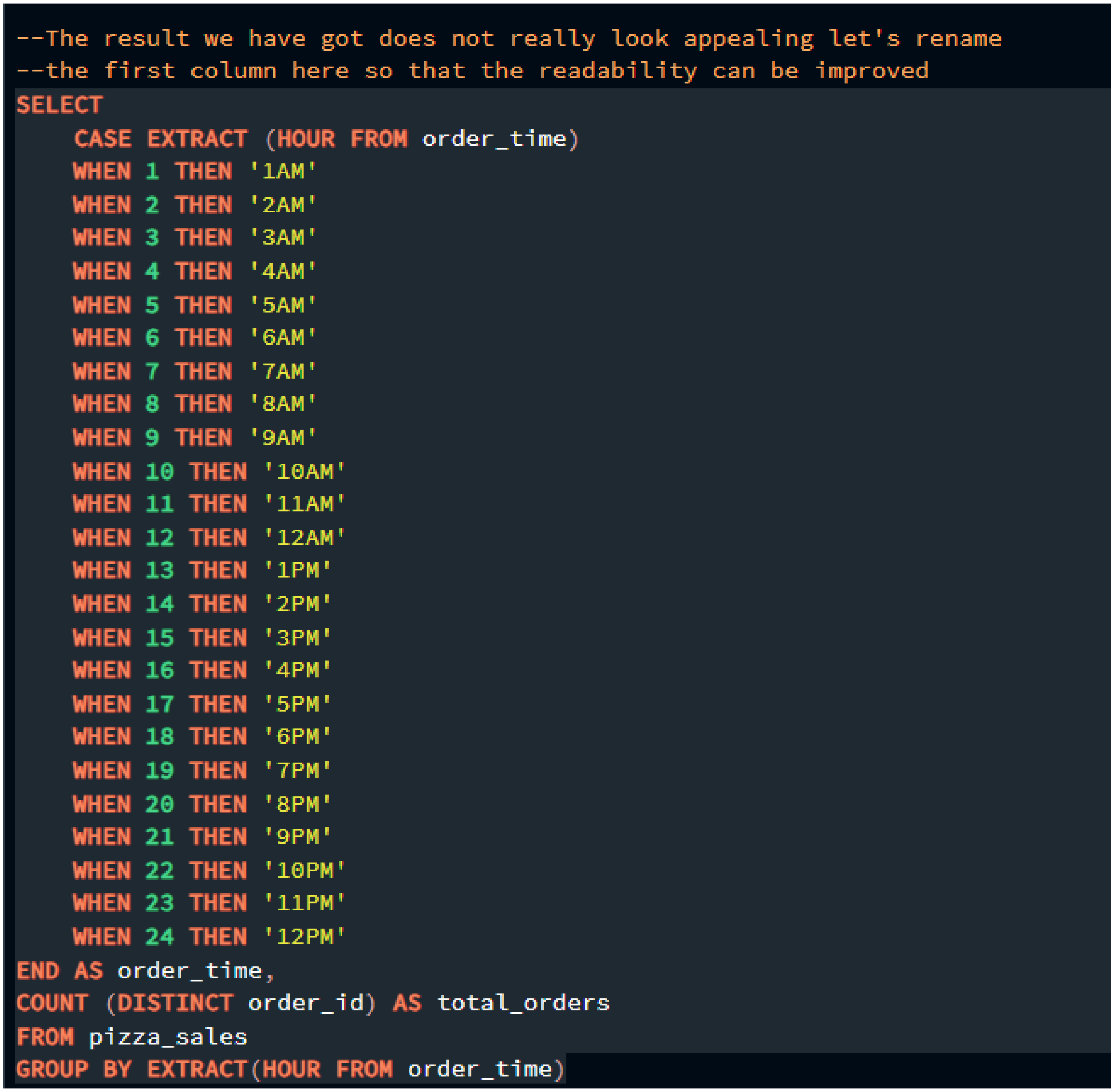
8.Now let's see on a 24-hour cycle how many orders are there.



```
-- Now lets see if we sort out no of orfers on hourly basis on 24 hour
SELECT
   EXTRACT(HOUR FROM order_time) AS Timing,
COUNT(DISTINCT order_id) AS total_orders
FROM pizza_sales
GROUP BY
EXTRACT(HOUR FROM order_time)
ORDER BY EXTRACT(HOUR FROM order_time)
```

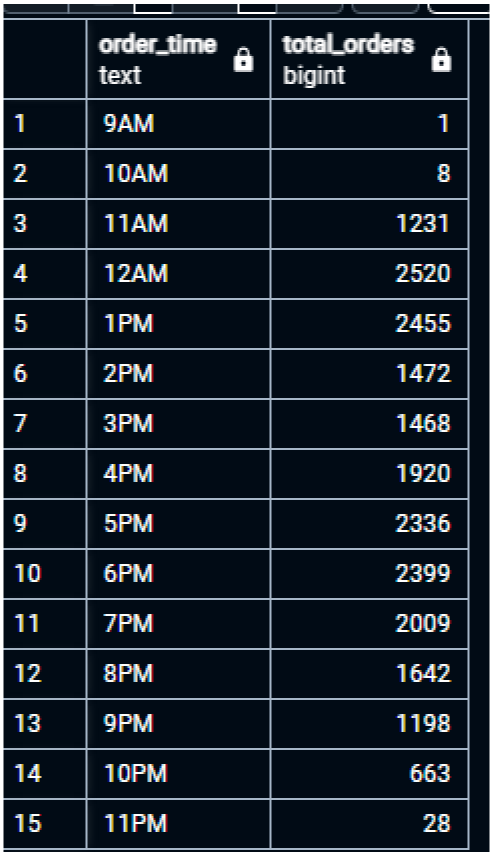
	timing numeric	total_orders bigint
1	9	1
2	10	8
3	11	1231
4	12	2520
5	13	2455
6	14	1472
7	15	1468
8	16	1920

9. For the sake of readability lets change the first column by firing these queries



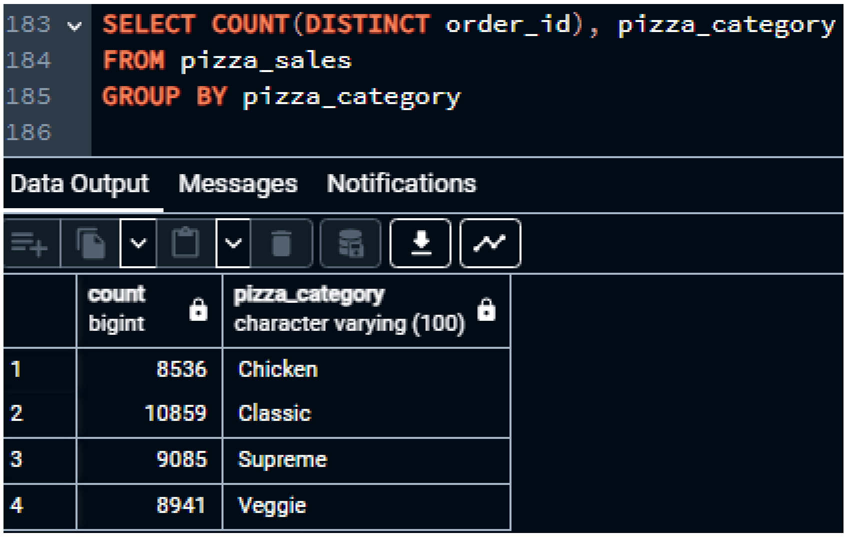
```sql
--The result we have got does not really look appealing let's rename
--the first column here so that the readability can be improved
SELECT
    CASE EXTRACT (HOUR FROM order_time)
    WHEN 1 THEN '1AM'
    WHEN 2 THEN '2AM'
    WHEN 3 THEN '3AM'
    WHEN 4 THEN '4AM'
    WHEN 5 THEN '5AM'
    WHEN 6 THEN '6AM'
    WHEN 7 THEN '7AM'
    WHEN 8 THEN '8AM'
    WHEN 9 THEN '9AM'
    WHEN 10 THEN '10AM'
    WHEN 11 THEN '11AM'
    WHEN 12 THEN '12AM'
    WHEN 13 THEN '1PM'
    WHEN 14 THEN '2PM'
    WHEN 15 THEN '3PM'
    WHEN 16 THEN '4PM'
    WHEN 17 THEN '5PM'
    WHEN 18 THEN '6PM'
    WHEN 19 THEN '7PM'
    WHEN 20 THEN '8PM'
    WHEN 21 THEN '9PM'
    WHEN 22 THEN '10PM'
    WHEN 23 THEN '11PM'
    WHEN 24 THEN '12PM'
END AS order_time,
COUNT (DISTINCT order_id) AS total_orders
FROM pizza_sales
GROUP BY EXTRACT(HOUR FROM order_time)
```

and this is the result we are getting-



	order_time text	total_orders bigint
1	9AM	1
2	10AM	8
3	11AM	1231
4	12AM	2520
5	1PM	2455
6	2PM	1472
7	3PM	1468
8	4PM	1920
9	5PM	2336
10	6PM	2399
11	7PM	2009
12	8PM	1642
13	9PM	1198
14	10PM	663
15	11PM	28

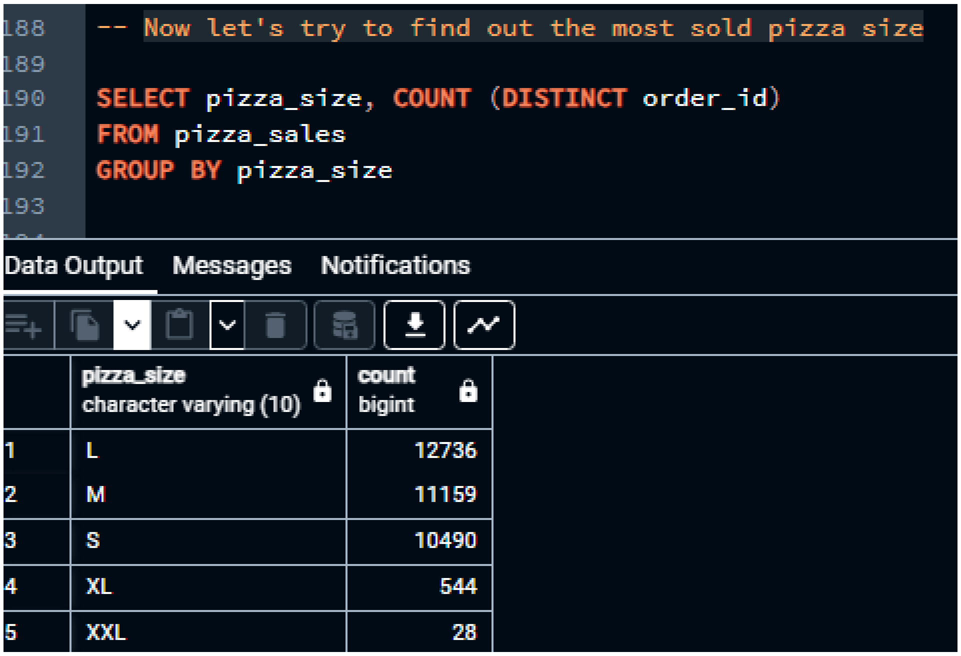
10.Now let's find out what is the distribution of pizza sales category wise



```
SELECT COUNT(DISTINCT order_id), pizza_category
FROM pizza_sales
GROUP BY pizza_category
```

	count bigint	pizza_category character varying (100)
1	8536	Chicken
2	10859	Classic
3	9085	Supreme
4	8941	Veggie

- 11.Now let's try to find out the most sold pizza size

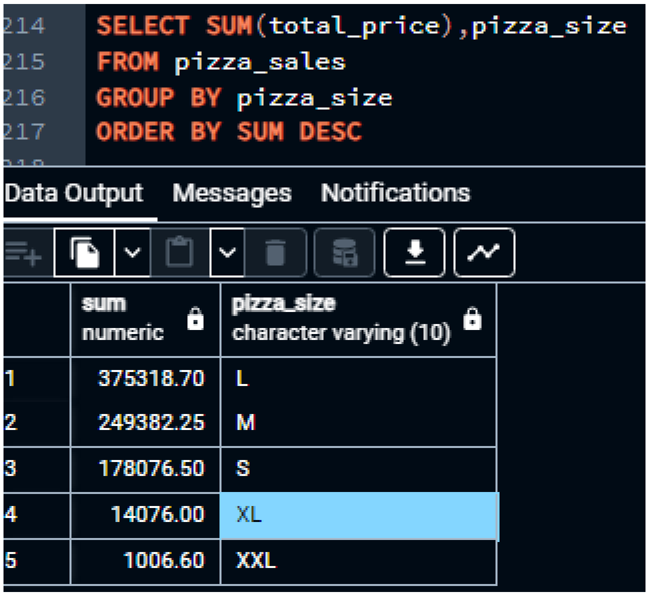


```sql
-- Now let's try to find out the most sold pizza size

SELECT pizza_size, COUNT (DISTINCT order_id)
FROM pizza_sales
GROUP BY pizza_size
```

	pizza_size character varying (10)	count bigint
1	L	12736
2	M	11159
3	S	10490
4	XL	544
5	XXL	28

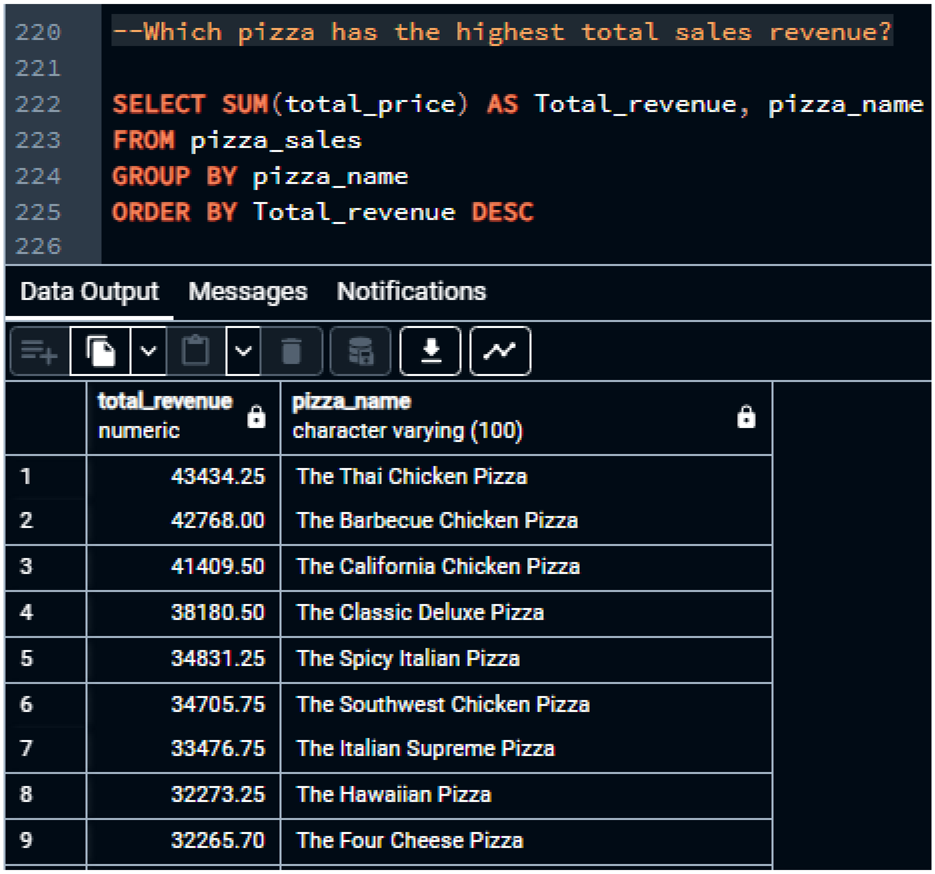
12. What is the total revenue generated by each pizza size (e.g., small, medium, large)?



```sql
SELECT SUM(total_price),pizza_size
FROM pizza_sales
GROUP BY pizza_size
ORDER BY SUM DESC
```

	sum numeric	pizza_size character varying (10)
1	375318.70	L
2	249382.25	M
3	178076.50	S
4	14076.00	XL
5	1006.60	XXL

13. Which pizza has the highest total sales revenue?



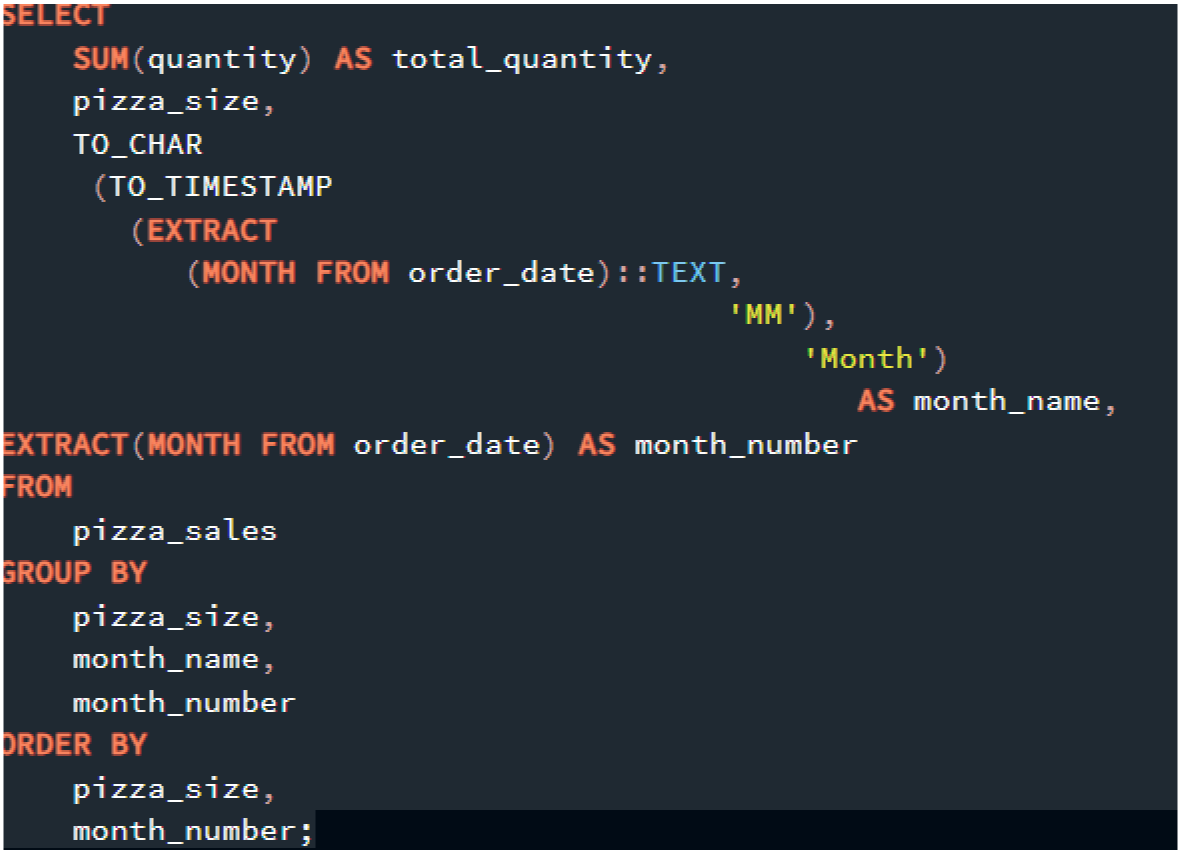
```sql
--Which pizza has the highest total sales revenue?

SELECT SUM(total_price) AS Total_revenue, pizza_name
FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_revenue DESC

```

	total_revenue numeric	pizza_name character varying (100)
1	43434.25	The Thai Chicken Pizza
2	42768.00	The Barbecue Chicken Pizza
3	41409.50	The California Chicken Pizza
4	38180.50	The Classic Deluxe Pizza
5	34831.25	The Spicy Italian Pizza
6	34705.75	The Southwest Chicken Pizza
7	33476.75	The Italian Supreme Pizza
8	32273.25	The Hawaiian Pizza
9	32265.70	The Four Cheese Pizza

14. Sales (quantity wise) of different pizza sizes across 12 months



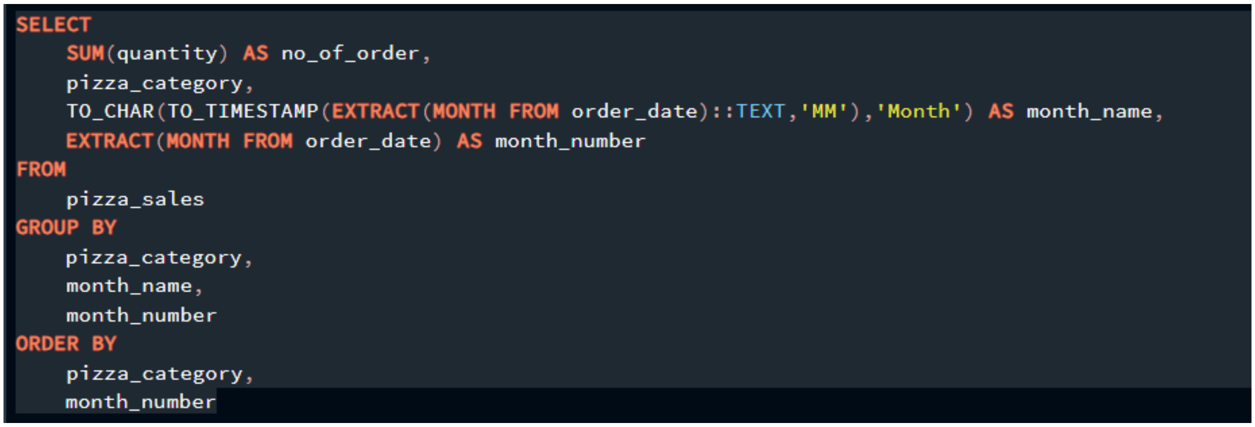
```sql
SELECT
    SUM(quantity) AS total_quantity,
    pizza_size,
    TO_CHAR
     (TO_TIMESTAMP
       (EXTRACT
          (MONTH FROM order_date)::TEXT,
                                       'MM'),
                                           'Month')
                                              AS month_name,
EXTRACT(MONTH FROM order_date) AS month_number
FROM
    pizza_sales
GROUP BY
    pizza_size,
    month_name,
    month_number
ORDER BY
    pizza_size,
    month_number;
```

```
"total_quantity"    "pizza_size"  "month_name"     "month_number"
1640  "L"   "January  "   1
1521  "L"   "February "   2
1651  "L"   "March    "   3
1541  "L"   "April    "   4
1668  "L"   "May      "   5
1590  "L"   "June     "   6
1697  "L"   "July     "   7
1527  "L"   "August   "   8
1511  "L"   "September"   9
1485  "L"   "October  "   10
1623  "L"   "November "   11
1502  "L"   "December "   12
1311  "M"   "January  "   1
1210  "M"   "February "   2
1313  "M"   "March    "   3
1390  "M"   "April    "   4
```

1338	"M"	"May "	5
1335	"M"	"June "	6
1396	"M"	"July "	7
1331	"M"	"August "	8
1188	"M"	"September"	9
1253	"M"	"October "	10
1337	"M"	"November "	11
1233	"M"	"December "	12
1229	"S"	"January "	1
1189	"S"	"February "	2
1253	"S"	"March "	3
1151	"S"	"April "	4
1265	"S"	"May "	5
1131	"S"	"June "	6
1249	"S"	"July "	7
1264	"S"	"August "	8
1152	"S"	"September"	9
1101	"S"	"October "	10
1258	"S"	"November "	11
1161	"S"	"December "	12
50	"XL"	"January "	1
38	"XL"	"February "	2
41	"XL"	"March "	3
65	"XL"	"April "	4
54	"XL"	"May "	5
49	"XL"	"June "	6
50	"XL"	"July "	7
44	"XL"	"August "	8
36	"XL"	"September"	9
42	"XL"	"October "	10
45	"XL"	"November "	11
38	"XL"	"December "	12
2	"XXL"	"January "	1
3	"XXL"	"February "	2
3	"XXL"	"March "	3
4	"XXL"	"April "	4
3	"XXL"	"May "	5
2	"XXL"	"June "	6

2	"XXL"	"August "	8
3	"XXL"	"September"	9
2	"XXL"	"October "	10
3	"XXL"	"November "	11
1	"XXL"	"December "	12

15.sales (total quantity) of different pizza category across 12 months



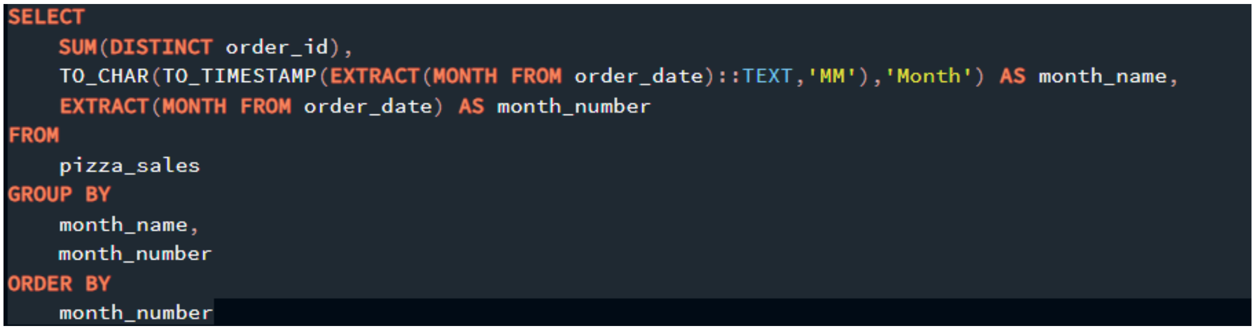
```
SELECT
    SUM(quantity) AS no_of_order,
    pizza_category,
    TO_CHAR(TO_TIMESTAMP(EXTRACT(MONTH FROM order_date)::TEXT,'MM'),'Month') AS month_name,
    EXTRACT(MONTH FROM order_date) AS month_number
FROM
    pizza_sales
GROUP BY
    pizza_category,
    month_name,
    month_number
ORDER BY
    pizza_category,
    month_number
```

"no_of_order"	"pizza_category"	"month_name"	"month_number"
913	"Chicken"	"January "	1
875	"Chicken"	"February "	2
994	"Chicken"	"March "	3
924	"Chicken"	"April "	4
939	"Chicken"	"May "	5
910	"Chicken"	"June "	6
963	"Chicken"	"July "	7
934	"Chicken"	"August "	8
900	"Chicken"	"September"	9
832	"Chicken"	"October "	10

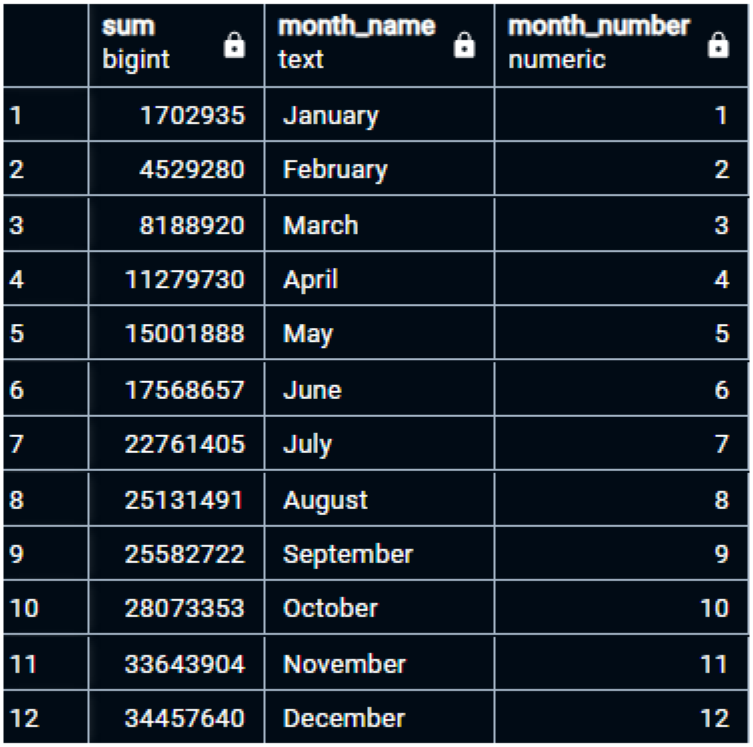
981	"Chicken"	"November "	11
885	"Chicken"	"December "	12
1257	"Classic"	"January "	1
1178	"Classic"	"February "	2
1236	"Classic"	"March "	3
1253	"Classic"	"April "	4
1324	"Classic"	"May "	5
1199	"Classic"	"June "	6
1331	"Classic"	"July "	7
1283	"Classic"	"August "	8
1202	"Classic"	"September"	9
1181	"Classic"	"October "	10
1262	"Classic"	"November "	11
1182	"Classic"	"December "	12
1044	"Supreme"	"January "	1
964	"Supreme"	"February "	2
991	"Supreme"	"March "	3
1013	"Supreme"	"April "	4
1045	"Supreme"	"May "	5
1040	"Supreme"	"June "	6
1041	"Supreme"	"July "	7
991	"Supreme"	"August "	8
877	"Supreme"	"September"	9
998	"Supreme"	"October "	10
1050	"Supreme"	"November "	11
933	"Supreme"	"December "	12

```
1018  "Veggie"    "January  "   1
944   "Veggie"    "February "   2
1040  "Veggie"    "March    "   3
961   "Veggie"    "April    "   4
1020  "Veggie"    "May      "   5
958   "Veggie"    "June     "   6
1057  "Veggie"    "July     "   7
960   "Veggie"    "August   "   8
911   "Veggie"    "September"   9
872   "Veggie"    "October  "   10
973   "Veggie"    "November "   11
935   "Veggie"    "December "   12
```

16. Total no of orders (not quantity) across 12 months

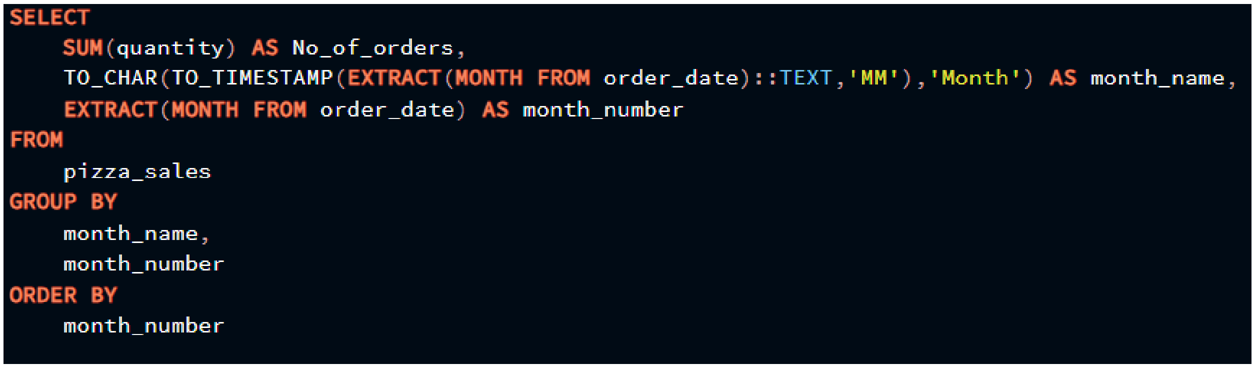


```
SELECT
    SUM(DISTINCT order_id),
    TO_CHAR(TO_TIMESTAMP(EXTRACT(MONTH FROM order_date)::TEXT,'MM'),'Month') AS month_name,
    EXTRACT(MONTH FROM order_date) AS month_number
FROM
    pizza_sales
GROUP BY
    month_name,
    month_number
ORDER BY
    month_number
```

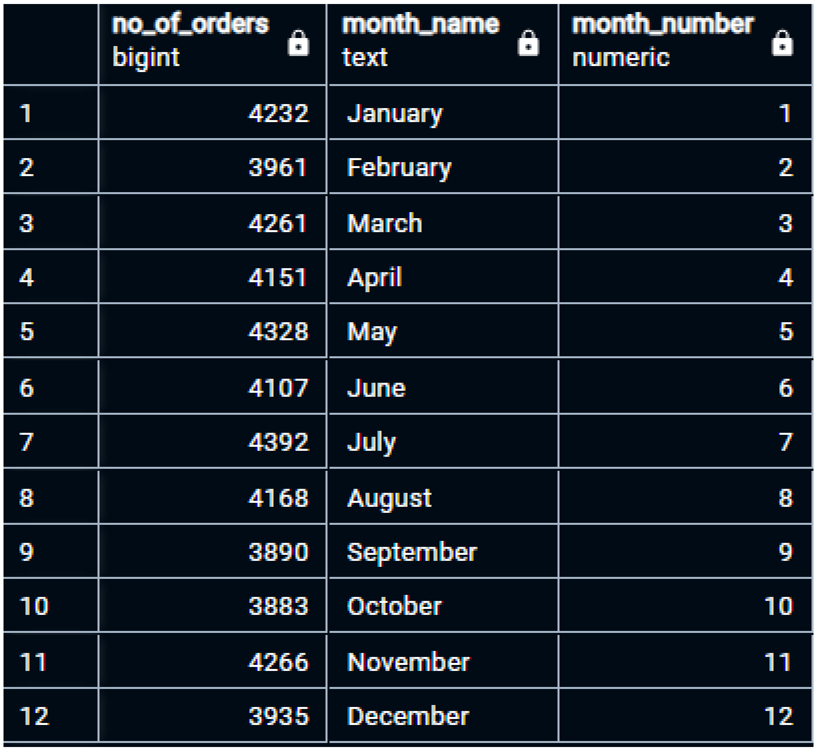


	sum bigint 🔒	month_name text 🔒	month_number numeric 🔒
1	1702935	January	1
2	4529280	February	2
3	8188920	March	3
4	11279730	April	4
5	15001888	May	5
6	17568657	June	6
7	22761405	July	7
8	25131491	August	8
9	25582722	September	9
10	28073353	October	10
11	33643904	November	11
12	34457640	December	12

17. Total no of orders quantity wise across 12 months

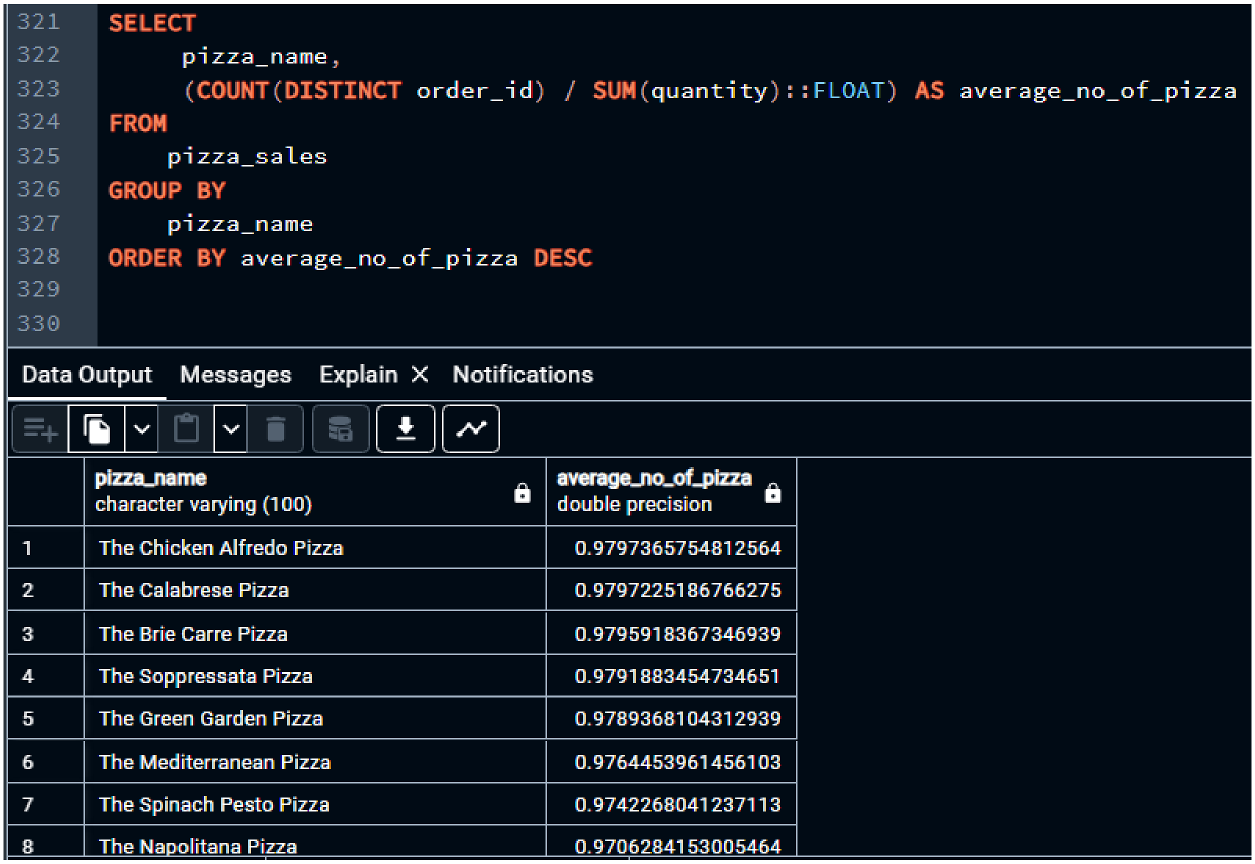


```
SELECT
    SUM(quantity) AS No_of_orders,
    TO_CHAR(TO_TIMESTAMP(EXTRACT(MONTH FROM order_date)::TEXT,'MM'),'Month') AS month_name,
    EXTRACT(MONTH FROM order_date) AS month_number
FROM
    pizza_sales
GROUP BY
    month_name,
    month_number
ORDER BY
    month_number
```



	no_of_orders bigint	month_name text	month_number numeric
1	4232	January	1
2	3961	February	2
3	4261	March	3
4	4151	April	4
5	4328	May	5
6	4107	June	6
7	4392	July	7
8	4168	August	8
9	3890	September	9
10	3883	October	10
11	4266	November	11
12	3935	December	12

18.Which pizza_name has the highest average quantity ordered per order?



```sql
SELECT
     pizza_name,
     (COUNT(DISTINCT order_id) / SUM(quantity)::FLOAT) AS average_no_of_pizza
FROM
    pizza_sales
GROUP BY
    pizza_name
ORDER BY average_no_of_pizza DESC
```

	pizza_name character varying (100)	average_no_of_pizza double precision
1	The Chicken Alfredo Pizza	0.9797365754812564
2	The Calabrese Pizza	0.9797225186766275
3	The Brie Carre Pizza	0.9795918367346939
4	The Soppressata Pizza	0.9791883454734651
5	The Green Garden Pizza	0.9789368104312939
6	The Mediterranean Pizza	0.9764453961456103
7	The Spinach Pesto Pizza	0.9742268041237113
8	The Napolitana Pizza	0.9706284153005464

19. Weekend sales vs. weekday sales (Total revenue)

-- --Weekend sales vs. weekday sales (No. Of orders)