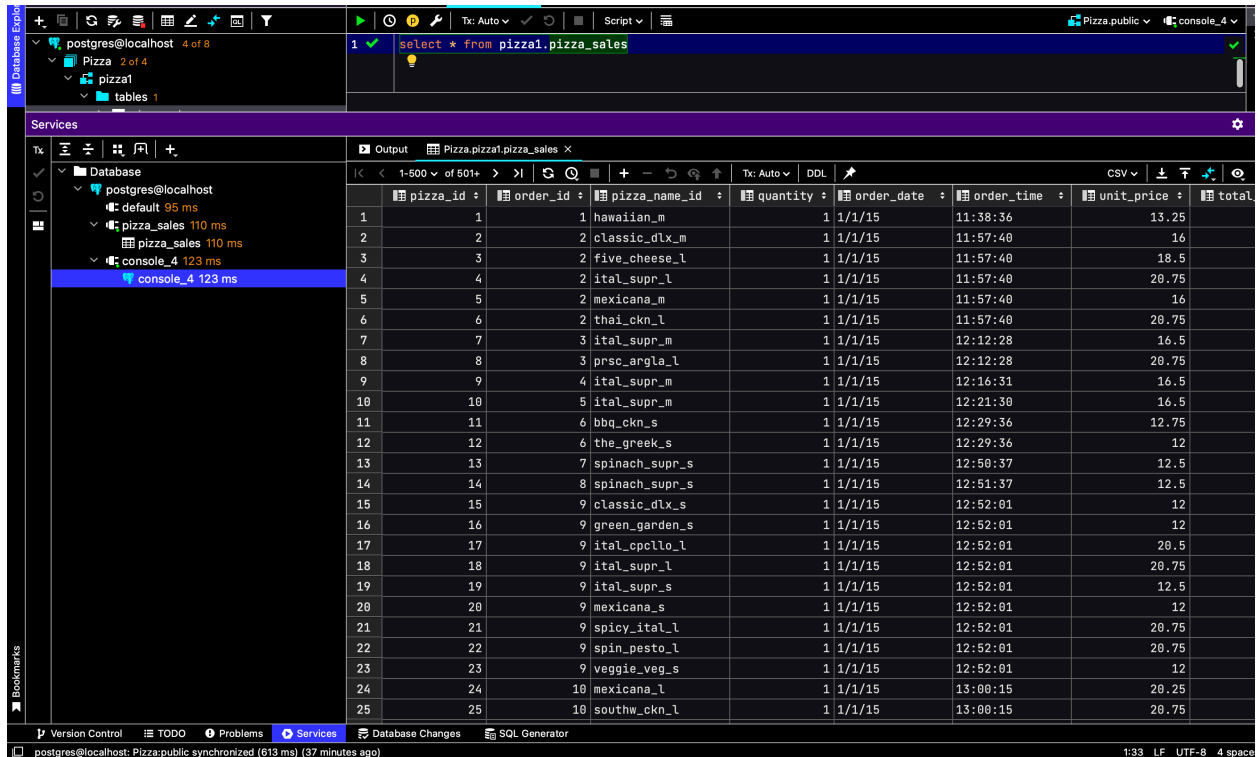


KPI's Requirement:

1.) Pizza Sales:

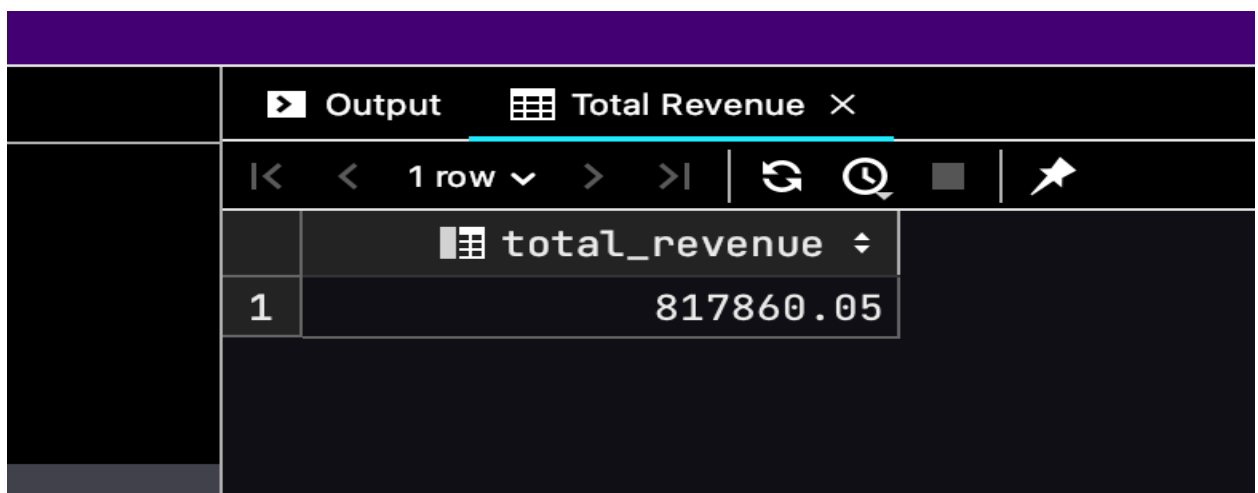


The screenshot shows a database management tool interface. On the left, a tree view shows the database structure: postgres@localhost, Pizza (2 of 4), pizza1, and tables (1). The main area displays a query result for the query `select * from pizza1.pizza_sales`. The result is a table with 8 columns: pizza_id, order_id, pizza_name_id, quantity, order_date, order_time, unit_price, and total. The table contains 25 rows of data, showing various pizza orders and their details.

pizza_id	order_id	pizza_name_id	quantity	order_date	order_time	unit_price	total
1	1	1 hawaiian_m	1	1/1/15	11:38:36	13.25	
2	2	2 classic_dlx_m	1	1/1/15	11:57:40	16	
3	3	2 five_cheese_l	1	1/1/15	11:57:40	18.5	
4	4	2 ital_supr_l	1	1/1/15	11:57:40	20.75	
5	5	2 mexicana_m	1	1/1/15	11:57:40	16	
6	6	2 thai_ckn_l	1	1/1/15	11:57:40	20.75	
7	7	3 ital_supr_m	1	1/1/15	12:12:28	16.5	
8	8	3 prsc_argla_l	1	1/1/15	12:12:28	20.75	
9	9	4 ital_supr_m	1	1/1/15	12:16:31	16.5	
10	10	5 ital_supr_m	1	1/1/15	12:21:30	16.5	
11	11	6 bbq_ckn_s	1	1/1/15	12:29:36	12.75	
12	12	6 the_greek_s	1	1/1/15	12:29:36	12	
13	13	7 spinach_supr_s	1	1/1/15	12:50:37	12.5	
14	14	8 spinach_supr_s	1	1/1/15	12:51:37	12.5	
15	15	9 classic_dlx_s	1	1/1/15	12:52:01	12	
16	16	9 green_garden_s	1	1/1/15	12:52:01	12	
17	17	9 ital_cpollo_l	1	1/1/15	12:52:01	20.5	
18	18	9 ital_supr_l	1	1/1/15	12:52:01	20.75	
19	19	9 ital_supr_s	1	1/1/15	12:52:01	12.5	
20	20	9 mexicana_s	1	1/1/15	12:52:01	12	
21	21	9 spicy_ital_l	1	1/1/15	12:52:01	20.75	
22	22	9 spin_pesto_l	1	1/1/15	12:52:01	20.75	
23	23	9 veggie_veg_s	1	1/1/15	12:52:01	12	
24	24	10 mexicana_l	1	1/1/15	13:00:15	20.25	
25	25	10 southw_ckn_l	1	1/1/15	13:00:15	20.75	

2.) Total Revenue:

```
/* Total Revenue */  
select sum(total_price) as Total Revenue from pizza1.pizza_sales
```



The screenshot shows a database management tool interface. The main area displays the result of the query `select sum(total_price) as Total Revenue from pizza1.pizza_sales`. The result is a table with 1 row and 1 column: total_revenue. The value in the row is 817860.05.

total_revenue
817860.05

3.) Average Order Value:

```
select * from pizzal.pizza_sales
/* Average Order Value */

select sum(total_price) / count(distinct order_id) as Avg_Order_Value from
pizzal.pizza_sales
```

Output Avg_Order_Value:numeric X	
1 row	
	avg_order_value
1	38.3072622950819672

4.) Total Pizzas sold:

```
select * from pizzal.pizza_sales
/* Total Pizzas Sold */
select sum(quantity) as Total Pizza Sold from pizzal.pizza_sales
```

Output Total_Pizza_Sold:numeric X	
1 row	
	total_pizza_sold
1	49574

5.) Total number of orders place:

```
select * from pizza1.pizza_sales

/* Total number of orders place */
select count(distinct order_id) as Total Orders from pizza1.pizza_sales
```

Output		Total_Orders:bigint	×
< < 1 row v > >		↺ ↻ 🔍	★
	total_orders	↕	
1	21350		

6.) Average Pizzas per order:

```
select * from pizza1.pizza_sales;

/* Average Pizzas per order */
select cast(sum(quantity) as decimal(10,2)) / cast(count(distinct order_id)
as decimal(10,2)) as avg_pizza_order
from pizza1.pizza_sales
```

Output		avg_pizza_order:decimal(10,2)	×
< < 1 row v > >		↺ ↻ 🔍	★
	avg_pizza_order	↕	
1	2.3219672131147541		

Problem Statement:

Chart Requirements

1.) Daily Trend for Total Orders:

```
select * from pizzal.pizza_sales;

/* Daily Trend for Total Orders */

SELECT TO_CHAR(TO_DATE(order_date, 'DD/MM/YY'), 'Dy') AS order_day,
        COUNT(DISTINCT order_id) AS total_orders
FROM pizzal.pizza_sales
GROUP BY TO_CHAR(TO_DATE(order_date, 'DD/MM/YY'), 'Dy');
```

Output

Result 67

7 rows

	order_day	total_orders
1	Fri	3538
2	Mon	2794
3	Sat	3158
4	Sun	2624
5	Thu	3239
6	Tue	2973
7	Wed	3024

2.) Monthly Trend for Total Orders:

```
select * from pizzal.pizza_sales;

/* Monthly Trend for Total Orders */

SELECT TO_CHAR(TO_DATE(order_date, 'DD/MM/YY'), 'Mon') AS month,
       COUNT(DISTINCT order_id) AS Total_orders
FROM pizzal.pizza_sales
GROUP BY TO_CHAR(TO_DATE(order_date, 'DD/MM/YY'), 'Mon')
ORDER BY Total_orders desc;
```

Output Result 71 X		
12 rows		
	month	total_orders
1	Jul	1935
2	May	1853
3	Jan	1845
4	Aug	1841
5	Mar	1840
6	Apr	1799
7	Nov	1792
8	Jun	1773
9	Feb	1685
10	Dec	1680
11	Sep	1661
12	Oct	1646

3.) Total revenue in percentage:

```
select * from pizzal.pizza_sales;

/* Total revenue in percentage */

select pizza_category, sum(total_price)* 100 / (select sum(total_price) from
pizzal.pizza_sales) as PCT
from pizzal.pizza_sales
group by pizza_category
```

	pizza_category	pct
1	Supreme	25.4563112600988396
2	Chicken	23.955137556847287
3	Veggie	23.6825909273842145
4	Classic	26.9059602556696589

4.) Percentage of sales by Pizza Size: *For quarter*

```
select * from pizzal.pizza_sales;

/* Percentage of sales by Pizza Size: For quarter*/

SELECT pizza_size,
       CAST(SUM(total_price) AS decimal(10, 2)) AS Total_Sales,
       CAST(SUM(total_price) * 100 / (SELECT SUM(total_price) FROM
pizzal.pizza_sales WHERE EXTRACT(QUARTER FROM TO_DATE(order_date,
'DD/MM/YY')) = 1) AS decimal(10, 2)) AS pct
FROM pizzal.pizza_sales
WHERE EXTRACT(QUARTER FROM TO_DATE(order_date, 'DD/MM/YY')) = 1
GROUP BY pizza_size
ORDER BY pct DESC;
```

	pizza_size	total_sales	pct
1	L	95229.65	46.37
2	M	61159.00	29.78
3	S	45384.25	22.10
4	XL	3289.50	1.60
5	XXL	287.60	0.14

Top 5 Best revenue pizzas:

```
select * from pizzal.pizza_sales;

/* Top 5 Best revenue pizzas */

select pizza_name, sum(total_price) as total_revenue from pizzal.pizza_sales
group by pizza_name
order by total_revenue desc
limit 5
```

	📄 pizza_name	📄 total_revenue
1	The Thai Chicken Pizza	43434.25
2	The Barbecue Chicken Pizza	42768
3	The California Chicken Pizza	41409.5
4	The Classic Deluxe Pizza	38180.5
5	The Spicy Italian Pizza	34831.25

Bottom 5 Worst revenue pizzas:

```
select * from pizzal.pizza_sales;

/* Bottom 5 Worst revenue pizzas */

select pizza_name, sum(total_price) as total_revenue from pizzal.pizza_sales
group by pizza_name
order by total_revenue
limit 5
```

	📄 pizza_name	📄 total_revenue
1	The Brie Carre Pizza	11588.5
2	The Green Garden Pizza	13955.75
3	The Spinach Supreme Pizza	15277.75
4	The Mediterranean Pizza	15360.5
5	The Spinach Pesto Pizza	15596

Top 5 Best Quantity Pizzas:

```
select * from pizzal.pizza_sales;

/* Top 5 Best Quantity pizzas */

select pizza_name, sum(quantity) as total_quantity from pizzal.pizza_sales
group by pizza_name
order by total_quantity desc
limit 5
```

Worst 5 Quantity pizzas:

```
select * from pizzal.pizza_sales;

/* Worst 5 Quantity pizzas */

select pizza_name, sum(quantity) as total_quantity from pizzal.pizza_sales
group by pizza_name
order by total_quantity
limit 5
```

	📄 pizza_name	📄 total_quantity
1	The Brie Carre Pizza	490
2	The Mediterranean Pizza	934
3	The Calabrese Pizza	937
4	The Spinach Supreme Pizza	950
5	The Soppresata Pizza	961

Top 5 according to Order id:

```
select * from pizzal.pizza_sales;

/* Top 5 according to Order id */

SELECT pizza_name, COUNT(DISTINCT order_id) AS total_orders
FROM pizzal.pizza_sales
GROUP BY pizza_name
ORDER BY total_orders DESC
LIMIT 5;
```

	📄 pizza_name	📄 total_orders
1	The Classic Deluxe Pizza	2329
2	The Hawaiian Pizza	2280
3	The Pepperoni Pizza	2278
4	The Barbecue Chicken Pizza	2273
5	The Thai Chicken Pizza	2225

Bottom 5 :

```
select * from pizzal.pizza_sales;

/* Bottom 5 according to Order id */

SELECT pizza_name, COUNT(DISTINCT order_id) AS total_orders
FROM pizzal.pizza_sales
GROUP BY pizza_name
ORDER BY total_orders
LIMIT 5;
```

	📄 pizza_name ⚡	📄 total_orders ⚡
1	The Brie Carre Pizza	480
2	The Mediterranean Pizza	912
3	The Calabrese Pizza	918
4	The Spinach Supreme Pizza	918
5	The Chicken Pesto Pizza	938