

# SQL & Power BI Pizza Sales Project

## Trouble importing table on pgAdmin

1. Fix Excel:
  - a. Convert comma-separated cells into double-quoted
    - Convert the data into a table
    - Copy the column values to a new column
    - write formula: =""""&\$B&""""
    - Copy the formula to all cells of the column
    - Replace the original column by the new column by pasting the values only
  - b. Convert dates into YYYY-MM-DD format
  - c. For large cell data, change datatype to TEXT, not VARCHAR
2. Copy the below code in the Query Tool after storing the CSV file in the C:/data folder:

```
copy pizza_raw
FROM 'C:/data/pizza_sales.csv'
WITH (FORMAT csv, HEADER true, DELIMITER ',', QUOTE '''', ENCODING 'UTF8');
```

## 3. Trouble with different formats in the date/time data type

- a. Check if there is any empty date:

```
SELECT COUNT(*)
FROM pizza_raw
WHERE order_date IS NULL
OR TRIM(order_date) = '';
```

b. Count how many rows are in DD-MM-YYYY format	Count how many rows are in YYYY-MM-DD format	Check Slash format
<pre>SELECT COUNT(*) FROM pizza_raw WHERE TRIM(order_date) ~ '^\\d{2}-\\d{2}-\\d{4}\$';</pre>	<pre>SELECT COUNT(*) FROM pizza_raw WHERE TRIM(order_date) ~ '^\\d{4}-\\d{2}-\\d{2}-\\d{2}\$';</pre>	<pre>SELECT COUNT(*) FROM pizza_raw WHERE TRIM(order_date) ~ '^\\d{2}/\\d{2}/\\d{4}\$';</pre>

#### 4. Now format the table with correct data types

```

CREATE TABLE pizzas AS
SELECT
pizza_id::INT,
order_id::INT,
pizza_name_id,
quantity::INT,
CASE
    WHEN TRIM(order_date) ~ '^\\d{2}-\\d{2}-\\d{4}$'
        THEN TO_DATE(TRIM(order_date), 'DD-MM-YYYY')
    WHEN TRIM(order_date) ~ '^\\d{4}-\\d{2}-\\d{2}$'
        THEN TO_DATE(TRIM(order_date), 'YYYY-MM-DD')
    ELSE NULL
END AS order_date,
CASE
    WHEN TRIM(order_time) ~ '^\\d{2}:\\d{2}:\\d{2}$'
        THEN TRIM(order_time)::TIME
    ELSE NULL
END AS order_time,
unit_price::NUMERIC,
total_price::NUMERIC,
pizza_size,
pizza_category,
```

```
pizza_ingredients,  
pizza_name
```

```
FROM pizza_raw;
```

## Requirements of clients

### A. KPIs

1. Total revenue: Sum of the total price of all pizza orders

```
SELECT SUM(total_price) AS To  
tal_Revenue FROM pizza_sales;
```

	total_revenue
1	817860.05

2. Average order value: The average amount spent per order = total revenue/total orders

```
SELECT (SUM(total_price)/COUN  
T(DISTINCT order_id)) AS Avg_  
order_value  
FROM pizza_sales;
```

	avg_order_value
1	38.3072622950819672

Here, one person can order more than one pizza; hence, distinct order\_ids are taken.

3. Total pizzas sold: Sum of the number of pizzas sold

```
SELECT SUM(quantity) AS Total  
_Pizza_Sold  
FROM pizza_sales;
```

	total_pizza_sold
1	49574

#### 4. Total Orders

```
SELECT COUNT(DISTINCT order_id) AS Total_Orders  
FROM pizza_sales;
```

	total_orders	bigint
1	21350	

#### 5. Average pizzas per order: total number of pizzas sold/total number of orders

```
SELECT ROUND((CAST(SUM(quantity) AS DECIMAL(10,2))/CAST(COUNT(DISTINCT order_id) AS DECIMAL(10,2))), 2)  
AS Avg_pizza_per_order  
FROM pizza_sales;
```

	avg_pizza_per_order	numeric
1	2.32	

#### A.A. Charts Requirements

1. Daily trend for total orders: bar chart → displays daily trends of total orders over a specific time period → identify patterns in order volumes on daily basis

```
SELECT TO_CHAR(order_date::date, 'Day') AS order_day, COUNT(DISTINCT order_id) AS total_order  
FROM pizza_sales  
GROUP BY order_day;
```

	order_day	total_order
1	Friday	3359
2	Monday	2940
3	Saturday	3126
4	Sunday	2710
5	Thursday	3173
6	Tuesday	2978
7	Wednesday	3064

2. Hourly trend for total orders: Line chart → throughout a day → identify peak hours for high activity

```

SELECT TO_CHAR(order_time::time, 'HH24') AS hourly_order,
       COUNT(DISTINCT order_id) AS total_orders
  FROM pizza_sales
 GROUP BY hourly_order
 ORDER BY hourly_order;

```

hourly_order	total_orders
09	4
10	8
11	1228
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28

3. Percentage of sales by pizza category: Pie chart → Show popularity of different pizza categories + their contribution to sales

Using  
PERCENT\_RANK,

```

SELECT pizza_category, SUM(total_price) AS total_sales,
       PERCENT_RANK() OVER (ORDER BY SUM(total_price))
       * 100 AS ranking_category
  FROM pizza_sales
 GROUP BY pizza_category;

```

pizza_category	total_sales	ranking_category
Veggie	193690.45	0
Chicken	195919.50	33.33333333333333
Supreme	208197.00	66.66666666666666
Classic	220053.10	100

Using percent ratio,

```

SELECT pizza_category, SUM(total_price)*100/ (SELECT SUM(total_price) FROM pizza_sales) AS ranking_category
  FROM pizza_sales
 GROUP BY pizza_category;

```

pizza_category	ranking_category
Supreme	25.4563112600988396
Chicken	23.9551375568472870
Veggie	23.6825909273842145
Classic	26.9059602356696589

4. Percentage of sales by pizza size: Pie chart → show customer preference on pizza sizes and their impact on sales

Using  
PERCENT\_RANK,

```
SELECT pizza_size, SUM(total_price)
AS total_sales,
PERCENT_RANK() OVER (ORDER BY SUM(total_price))
* 100 AS ranking_size
FROM pizza_sales
GROUP BY pizza_size;
```

	pizza_size	total_sales	ranking_size
1	XXL	1006.60	0
2	XL	14076.0	25
3	S	178076.50	50
4	M	249382.25	75
5	L	375318.70	100

Using percent ratio,

```
SELECT pizza_size, SUM(total_price)*100/ (SELECT SUM(total_price) FROM pizza_sales) AS ranking_size
FROM pizza_sales
GROUP BY pizza_size
ORDER BY pizza_size DESC;
```

	pizza_size	ranking_size
1	XXL	0.12307729176892794800
2	XL	1.7210768517180904
3	S	21.7734684558806852
4	M	30.4920444518594593
5	L	45.8903329487728371

5. Total pizzas sold by pizza category: Funnel chart → Compare sales performance of each pizza category

```
SELECT pizza_category, SUM(quantity) AS total_pizzas_sold
FROM pizza_sales
GROUP BY pizza_category
ORDER BY pizza_category;
```

	pizza_category	total_pizzas_sold
1	Chicken	11050
2	Classic	14888
3	Supreme	11987
4	Veggie	11649

6. Top 5 best sellers for total pizzas sold: Bar chart → Identify the most popular pizza options

```

SELECT pizza_name, SUM(quantity) AS total_quantity_sold
FROM pizza_sales
GROUP BY pizza_name
ORDER BY total_quantity_sold
DESC
LIMIT 5;

```

	pizza_name text	total_quantity_sold bigint
1	"The Classic Deluxe Pizza"	2453
2	"The Barbecue Chicken Piz..."	2432
3	"The Hawaiian Pizza"	2422
4	"The Pepperoni Pizza"	2418
5	"The Thai Chicken Pizza"	2371

7. Top 5 worst sellers for total pizzas sold: Bar chart → Identify the least popular pizza options

```

SELECT pizza_name, SUM(quantity) AS total_quantity_sold
FROM pizza_sales
GROUP BY pizza_name
ORDER BY total_quantity_sold
LIMIT 5;

```

	pizza_name text	total_quantity_sold bigint
1	"The Brie Carre Pizza"	490
2	"The Mediterranean Pizza"	934
3	"The Calabrese Pizza"	937
4	"The Spinach Supreme Piz..."	950
5	"The Soppressata Pizza"	961

## Excel Part

### Data Cleaning:

1. **Change pizza\_size from one character (L) to full form (Large):**
  - a. Select column pizza\_size
  - b. Select ctrl + H to open find & replace option
  - c. Change