

# Pizza Sales Analysis Project On MySQL

## 1<sup>st</sup> Step - Entering data into MySQL

//Creating databsase by name of practice\_pizza\_analysis

```
CREATE SCHEMA practice_pizza_sales_analysis;
```

```
use practice_pizza_sales_analysis;
```

//Creating tables using syntax

```
CREATE TABLE pizzas (
```

```
    pizza_id VARCHAR(14) NOT NULL,
```

```
    pizza_type_id VARCHAR(12) NOT NULL,
```

```
    size VARCHAR(3) NOT NULL,
```

```
    price DECIMAL(38, 2) NOT NULL
```

```
);
```

```
CREATE TABLE orders (
```

```
    order_id DECIMAL(38, 0) NOT NULL,
```

```
    `date` DATE NOT NULL,
```

```
    `time` TIME NOT NULL
```

```
);
```

```
CREATE TABLE order_details (
```

```
    order_details_id DECIMAL(38, 0) NOT NULL,
```

```
    order_id DECIMAL(38, 0) NOT NULL,
```

```
    pizza_id VARCHAR(14) NOT NULL,
```

```
    quantity DECIMAL(38, 0) NOT NULL
```

```
);
```

//Creating table using Create a new table in connected sql server icon

```
CREATE TABLE `practice_pizza_sales_analysis`.`pizza_types` (
```

```
    `pizza_type_id` VARCHAR(300) NOT NULL,
```

```
    `name` VARCHAR(300) NULL,
```

```
    `ingredients` VARCHAR(300) NULL,
```

```
    `category` VARCHAR(300) NULL,
```

PRIMARY KEY (`pizza\_type\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8mb4;

**//Load data from .csv file to mysql server using load query:**

- 1) load data infile 'orders.csv'  
into table orders  
fields terminated by ','  
ignore 1 rows;
- 2) load data infile 'order\_details.csv'  
into table order\_details  
fields terminated by ','  
ignore 1 rows;
- 3) load data infile 'pizzas.csv'  
into table pizzas  
fields terminated by ','  
ignore 1 rows;

**I have Load the data in pizza\_types table using table import wizard box**

**It came with double quotes as in the .csv file So I removed double quotes using following UPDATE queries:**

**1) To remove left quotes**

UPDATE pizza\_types

SET ingredients=SUBSTRING(ingredients, 2, LENGTH(ingredients));

**2) To remove right quotes**

UPDATE pizza\_types

SET ingredients=SUBSTRING(ingredients, 1, LENGTH(ingredients)-1);

**//Assigning Primary keys to each table using ALTER queries as follows:**

```
ALTER TABLE `practice_pizza_sales_analysis`.`order_details`
```

```
ADD PRIMARY KEY (`order_details_id`);
```

```
;
```

```
ALTER TABLE `practice_pizza_sales_analysis`.`orders`
```

```
ADD PRIMARY KEY (`order_id`);
```

```
;
```

```
ALTER TABLE `practice_pizza_sales_analysis`.`pizzas`
```

```
ADD PRIMARY KEY (`pizza_id`);
```

```
;
```

```
ALTER TABLE `practice_pizza_sales_analysis`.`pizza_types`
```

```
ADD PRIMARY KEY (`pizza_type_id`);
```

```
;
```

## 2<sup>nd</sup> Step - Analysing data using Queries

### 1. How many customers do we have each day?

```
with dailycustomers AS (  
  select count(order_id),count(distinct(`date`))  
  from orders)  
select count(order_id)/count(distinct(`date`)) as average  
from orders,dailycustomers;
```

Result Grid	Filter Rows:	Export:
	average	
▶	59.6369	

## 2. Are there any peak hours?

Create view timezoneview as

Select order\_id,`time`,(case

when'00:00:00'<=`time`AND`time`<='00:59:59'then"12 to 1"

when'01:00:00'<=`time`AND`time`<='01:59:59'then"1 to 2"

when'02:00:00'<=`time`AND`time`<='02:59:59'then"2 to 3"

when'03:00:00'<=`time`AND`time`<='03:59:59'then"3 to 4"

when'04:00:00'<=`time`AND`time`<='04:59:59'then"4 to 5"

when'05:00:00'<=`time`AND`time`<='05:59:59'then"5 to 6"

when'06:00:00'<=`time`AND`time`<='06:59:59'then"6 to 7"

when'07:00:00'<=`time`AND`time`<='07:59:59'then"7 to 8"

when'08:00:00'<=`time`AND`time`<='08:59:59'then"8 to 9"

when'09:00:00'<=`time`AND`time`<='09:59:59'then"9 to 10"

when'10:00:00'<=`time`AND`time`<='10:59:59'then"10 to 11"

when'11:00:00'<=`time`AND`time`<='11:59:59'then"11 to 12"

when'12:00:00'<=`time`AND`time`<='12:59:59'then"12 to 13"

when'13:00:00'<=`time`AND`time`<='13:59:59'then"13 to 14"

when'14:00:00'<=`time`AND`time`<='14:59:59'then"14 to 15"

when'15:00:00'<=`time`AND`time`<='15:59:59'then"15 to 16"

when'16:00:00'<=`time`AND`time`<='16:59:59'then"16 to 17"

when'17:00:00'<=`time`AND`time`<='17:59:59'then"17 to 18"

when'18:00:00'<=`time`AND`time`<='18:59:59'then"18 to 19"

when'19:00:00'<=`time`AND`time`<='19:59:59'then"19 to 20"

when'20:00:00'<=`time`AND`time`<='20:59:59'then"20 to 21"

when'21:00:00'<=`time`AND`time`<='21:59:59'then"21 to 22"

when'22:00:00'<=`time`AND`time`<='22:59:59'then"22 to 23"

when'23:00:00'<=`time`AND`time`<='23:59:59'then"23 to 24"

END) as timezone

from orders;

select timezone,count(order\_id) from timezoneview

group by timezone

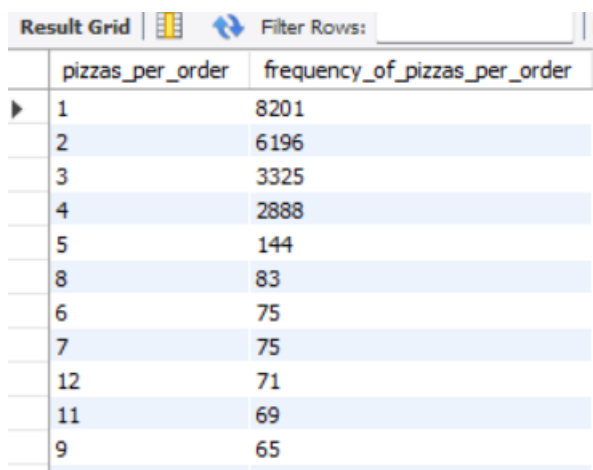
order by 2 desc

limit 6;

Result Grid	Filter Rows:
timezone	count(order_id)
12 to 13	2520
13 to 14	2455
18 to 19	2399
17 to 18	2336
19 to 20	2009
16 to 17	1920

### 3. How many pizzas are typically in an order?

```
create view frequency_pizza as
select order_id,count(quantity) as pizzas_per_order
from order_details
group by order_id;
select pizzas_per_order,count(pizzas_per_order) as
frequency_of_pizzas_per_order
from frequency_pizza
group by pizzas_per_order
order by 2 desc;
select avg(pizzas_per_order) from frequency_pizza;
```

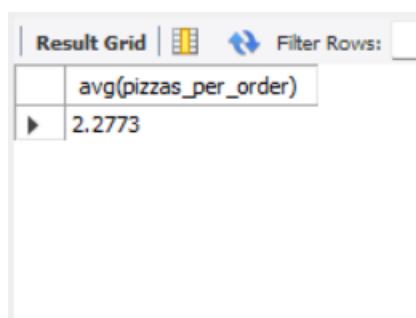


The screenshot shows a 'Result Grid' window with a 'Filter Rows' button. The grid contains two columns: 'pizzas\_per\_order' and 'frequency\_of\_pizzas\_per\_order'. The data is sorted in descending order of frequency. The first row shows 1 pizza per order with a frequency of 8201. The second row shows 2 pizzas per order with a frequency of 6196. The third row shows 3 pizzas per order with a frequency of 3325. The fourth row shows 4 pizzas per order with a frequency of 2888. The fifth row shows 5 pizzas per order with a frequency of 144. The sixth row shows 8 pizzas per order with a frequency of 83. The seventh row shows 6 pizzas per order with a frequency of 75. The eighth row shows 7 pizzas per order with a frequency of 75. The ninth row shows 12 pizzas per order with a frequency of 71. The tenth row shows 11 pizzas per order with a frequency of 69. The eleventh row shows 9 pizzas per order with a frequency of 65.

pizzas_per_order	frequency_of_pizzas_per_order
1	8201
2	6196
3	3325
4	2888
5	144
8	83
6	75
7	75
12	71
11	69
9	65

Here we manually checked that there are very few outliers which won't affect

The average so much so we applied average formula directly



The screenshot shows a 'Result Grid' window with a 'Filter Rows' button. The grid contains one column: 'avg(pizzas\_per\_order)'. The first row shows the average value of 2.2773.

avg(pizzas_per_order)
2.2773

#### 4. Do we have any bestsellers?

```
select sum(sold_pizzas),pizza_types.`name` from pizza_types left join
(select order_details.pizza_id,pizzas.pizza_type_id, count(quantity) as
sold_pizzas
from order_details
left join pizzas on order_details.pizza_id=pizzas.pizza_id
group by order_details.pizza_id
order by 3 desc) as quantity_sold
on pizza_types.pizza_type_id=quantity_sold.pizza_type_id
group by `name`
order by 1 desc
limit 10;
```

Result Grid			Filter Rows:	
	sum(sold_pizzas)	name		
▶	2416	The Classic Deluxe Pizza		
	2372	The Barbecue Chicken Pizza		
	2370	The Hawaiian Pizza		
	2369	The Pepperoni Pizza		
	2315	The Thai Chicken Pizza		
	2302	The California Chicken Pizza		
	1887	The Sicilian Pizza		
	1887	The Spicy Italian Pizza		
	1885	The Southwest Chicken Pizza		
	1850	The Four Cheese Pizza		

## 5. How much money did we make this year?

```
select sum(total_value) from
(select quantity_sold, price, (quantity_sold*price) as total_value from (select
pizza_id,count(quantity) as quantity_sold
from order_details
group by pizza_id)as pizzas_sold left join pizzas
on pizzas_sold.pizza_id=pizzas.pizza_id)as final;
```

Result Grid		Filter Rows:
	sum(total_value)	
▶	801944.70	



## 6. Are there any pizzas we should take of the menu, or any promotions we could leverage?

```
select sum(sold_pizzas),pizza_types.`name` from pizza_types left join
(select order_details.pizza_id,pizzas.pizza_type_id, count(quantity) as
sold_pizzas
from order_details
left join pizzas on order_details.pizza_id=pizzas.pizza_id
group by order_details.pizza_id
order by 3 desc) as quantity_sold
on pizza_types.pizza_type_id=quantity_sold.pizza_type_id
group by `name`
order by 1;
```

Result Grid	Filter Rows:	Export
	sum(sold_pizzas)	name
▶	480	The Brie Carre Pizza
	923	The Mediterranean Pizza
	927	The Calabrese Pizza
	940	The Spinach Supreme Pizza
	957	The Soppressata Pizza