

Objective:

The goal of this data analysis task was to extract meaningful insights from the pizza sales dataset, providing key performance indicators (KPIs) and trends that can help in understanding and optimizing the business.

Dataset:

The dataset, named `pizza_sales`, contains information about pizza orders. Key columns include `order_id`, `order_date`, `order_time`, `pizza_name`, `pizza_category`, `pizza_size`, `quantity`, and `total_price`.

Link:

[Tableau Visualization](#)

Queries:

Below.

PIZZA SALES SQL QUERIES

A. KPI's

1. Total Revenue:

```
SELECT SUM(total_price) AS Total_Revenue FROM pizza_sales
```

The screenshot shows the MySQL Workbench interface with a single query window titled "Data Cleaning Portfolio Project Queries". The query is:

```
1 SELECT SUM(total_price) AS Total_Revenue FROM pizza_sales;
```

The results grid shows one row with the value 817860.049999993. The action output pane shows the execution details:

Time	Action	Response	Duration / Fetch Time
17:56:26	select sum(total_price) from projects.pizza_sales	48620 row(s) returned	0.00050 sec / 0.000016...
17:56:26	select pizza_name, count(distinct(order_id)) from projects.pizza_sales group by pizza_name	5 row(s) returned	0.084 sec / 0.000016...
17:56:26	select * from projects.pizza_sales	48620 row(s) returned	0.00050 sec / 0.081...
19:33:30	SELECT SUM(total_price) AS Total_Revenue FROM pizza_sales	1 row(s) returned	0.138 sec / 0.0003...

At the bottom, it says "Query Completed".

2. Average Order Value

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

The screenshot shows the MySQL Workbench interface with a single query window titled "Data Cleaning Portfolio Project Queries". The query is:

```
1 SELECT (SUM(total_price) / COUNT(DISTINCT order_id)) AS Avg_order_Value FROM pizza_sales
```

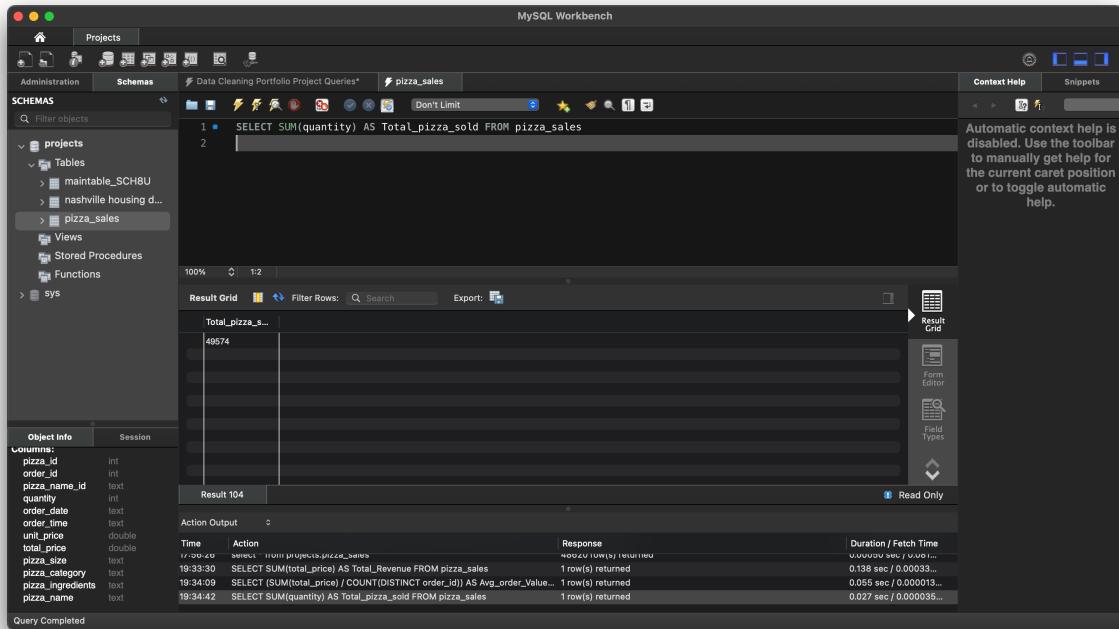
The results grid shows one row with the value 38.307262295081635. The action output pane shows the execution details:

Time	Action	Response	Duration / Fetch Time
17:56:26	select sum(total_price), count(distinct(order_id)) from projects.pizza_sales group by pizza_name	3 row(s) returned	0.00050 sec / 0.000016...
17:56:26	select * from projects.pizza_sales	48620 row(s) returned	0.084 sec / 0.000016...
19:33:30	SELECT SUM(total_price) AS Total_Revenue FROM pizza_sales	1 row(s) returned	0.138 sec / 0.0003...
19:34:09	SELECT (SUM(total_price) / COUNT(DISTINCT order_id)) AS Avg_order_Value	1 row(s) returned	0.055 sec / 0.000013...

At the bottom, it says "Query Completed".

3. Total Pizzas Sold

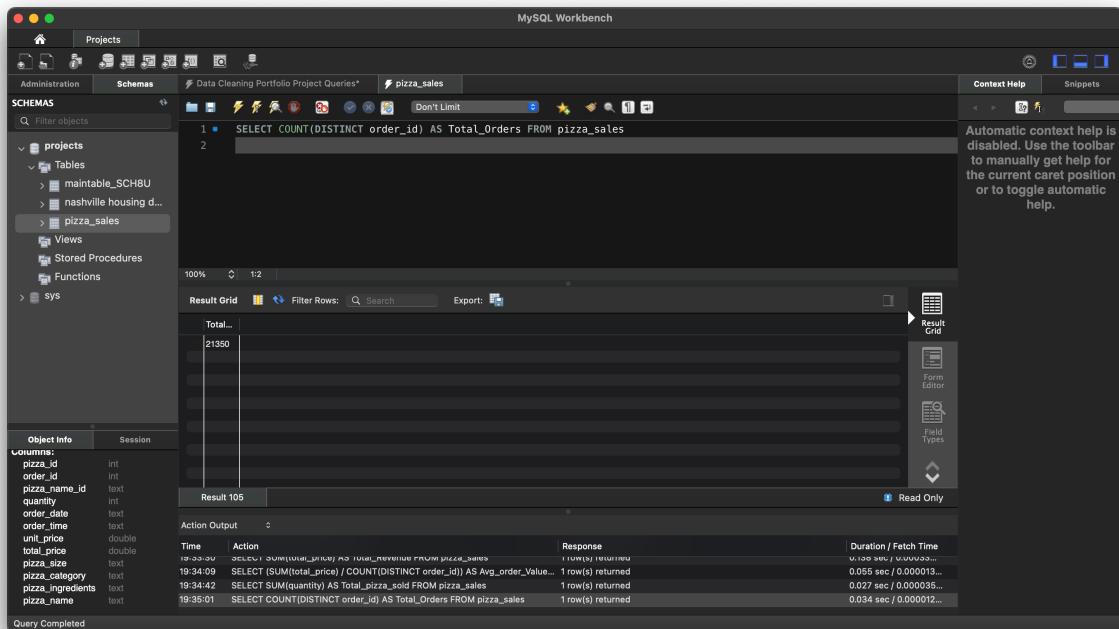
```
SELECT SUM(quantity) AS Total_pizza_sold FROM pizza_sales
```



The screenshot shows the MySQL Workbench interface. The left sidebar displays the schema structure under the 'pizza_sales' table, including columns like pizza_id, order_id, pizza_name_id, quantity, etc. The main query editor window contains the SQL query: `SELECT SUM(quantity) AS Total_pizza_sold FROM pizza_sales`. The results pane shows a single row with the value 49574. The status bar at the bottom right indicates 'Query Completed'.

4. Total Orders

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



The screenshot shows the MySQL Workbench interface. The left sidebar displays the schema structure under the 'pizza_sales' table. The main query editor window contains the SQL query: `SELECT COUNT(DISTINCT order_id) AS Total_Orders FROM pizza_sales`. The results pane shows a single row with the value 21350. The status bar at the bottom right indicates 'Query Completed'.

5. Average Pizzas Per Order

```
SELECT CAST(CAST(SUM(quantity) AS DECIMAL(10,2)) /
```

```

CAST(COUNT(DISTINCT order_id) AS DECIMAL(10,2)) AS DECIMAL(10,2))

AS Avg_Pizzas_per_order

FROM pizza_sales

```

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** Data Cleaning Portfolio Project Queries*
- Table:** pizza_sales
- Query:**

```

1 SELECT CAST(CAST(SUM(quantity) AS DECIMAL(10,2)) /
2   CAST(COUNT(DISTINCT order_id) AS DECIMAL(10,2)) AS DECIMAL(10,2))
3   AS Avg_Pizzas_per_order
4   FROM pizza_sales

```

- Result Grid:** Shows a single row with the value 2.32.
- Action Output:** Displays the execution log with the following entries:

Time	Action	Response	Duration / Fetch Time
19:34:03	SELECT CAST(CAST(SUM(quantity) AS DECIMAL(10,2)) / COUNT(DISTINCT order_id) AS DECIMAL(10,2)) AS Avg_Pizzas_per_order;	1 row(s) returned	0.042 sec / 0.000013...
19:34:42	SELECT SUM(quantity) AS Total_pizza_sold FROM pizza_sales	1 row(s) returned	0.027 sec / 0.000035...
19:35:01	SELECT COUNT(DISTINCT order_id) AS Total_Orders FROM pizza_sales	1 row(s) returned	0.034 sec / 0.000012...
19:35:30	SELECT CAST(CAST(SUM(quantity) AS DECIMAL(10,2)) / COUNT(DISTINCT order_id) AS DECIMAL(10,2)) AS Avg_Pizzas_per_order;	1 row(s) returned	0.042 sec / 0.000013...

B. Hourly Trend for Total Pizzas Sold

```
SELECT HOUR(order_time) as order_hours,
```

```
    SUM(quantity) as total_pizzas_sold
```

```
FROM pizza_sales
```

```
GROUP BY order_hours
```

```
ORDER BY order_hours;
```

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** Data Cleaning Portfolio Project Queries*
- Table:** pizza_sales
- Query:**

```

1 SELECT HOUR(order_time) as order_hours,
2   SUM(quantity) as total_pizzas_sold
3   FROM pizza_sales
4   GROUP BY order_hours
5   ORDER BY order_hours;

```

- Result Grid:** Shows a table with columns order_hours and total_pizzas_sold, containing data from hour 11 to 23.
- Action Output:** Displays the execution log with the following entries:

Time	Action	Response	Duration / Fetch Time
19:36:09	SELECT CAST(CAST(SUM(quantity) AS DECIMAL(10,2)) / COUNT(DISTINCT order_id) AS DECIMAL(10,2)) AS Avg_Pizzas_per_order;	1 row(s) returned	0.042 sec / 0.000013...
19:36:17	SELECT DATEPART(HOUR, order_time) as order_hours, SUM(quantity) as total_pizzas_sold FROM pizza_sales GROUP BY order_hours;	15 row(s) returned	Error Code: 1305. FUNCTION projects.DATEPART does not exist
19:36:20	SELECT DATEPART(HOUR, order_time) as order_hours, SUM(quantity) as total_pizzas_sold FROM pizza_sales GROUP BY order_hours;	15 row(s) returned	Error Code: 1305. FUNCTION projects.DATEPART does not exist
19:37:05	SELECT HOUR(order_time) as order_hours, SUM(quantity) as total_pizzas_sold FROM pizza_sales GROUP BY order_hours;	15 row(s) returned	0.0012 sec / 0.000011...

C. Weekly Trend for Orders

```
SELECT
    WEEK(STR_TO_DATE(order_date, '%d-%m-%Y'), 3) AS WeekNumber,
    YEAR(STR_TO_DATE(order_date, '%d-%m-%Y')) AS Year,
    COUNT(DISTINCT order_id) AS Total_orders
FROM
    pizza_sales
GROUP BY
    WeekNumber,
    Year
ORDER BY
    Year, WeekNumber;
```

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
SELECT
    WEEK(STR_TO_DATE(order_date, '%d-%m-%Y'), 3) AS WeekNumber,
    YEAR(STR_TO_DATE(order_date, '%d-%m-%Y')) AS Year,
    COUNT(DISTINCT order_id) AS Total_orders
FROM
    pizza_sales
GROUP BY
    WeekNumber,
    Year
ORDER BY
    Year, WeekNumber;
```

The results grid displays the following data:

WeekNumber	Year	Total_Orders
1	2015	400
2	2015	415
3	2015	436
4	2015	422
5	2015	423
6	2015	393
7	2015	409
8	2015	420
9	2015	404
10	2015	416
11	2015	427
12	2015	433
13	2015	408
14	2015	408
15	2015	408

The session details pane shows the following columns for the pizza_sales table:

Columns:
pizza_id
order_id
pizza_name_id
order_time
order_date
order_time
unit_price
total_price
pizza_size
pizza_category
pizza_ingredients
pizza_name

The status bar at the bottom left says "Query Completed".

D. % of Sales by Pizza Category

```
SELECT pizza_category, CAST(SUM(total_price) AS DECIMAL(10,2)) as total_revenue,
CAST(SUM(total_price) * 100 / (SELECT SUM(total_price) from pizza_sales) AS DECIMAL(10,2)) AS PCT
FROM pizza_sales
GROUP BY pizza_category
```

MySQL Workbench Screenshot showing a query results grid for the pizza_sales table. The grid displays four rows of data:

pizza_category	total_revenue	PCT
Classic	220053.10	26.91
Veggie	193890.45	23.68
Supreme	208197.00	25.46
Chicken	195819.50	23.98

The Action Output pane shows the execution log for the query:

```

Time Action Response Duration / Fetch Time
19:38:10 SELECT WEEK(STR_TO_DATE(order_date, '%d-%m-%Y'), 3) AS WeekNum... 0.07 sec / 0.000023...
19:38:33 SELECT WEEK(STR_TO_DATE(order_date, '%d-%m-%Y'), 3) AS WeekNum... 0.07 sec / 0.000015...
19:38:37 SELECT WEEK(STR_TO_DATE(order_date, '%d-%m-%Y'), 3) AS WeekNum... 0.127 sec / 0.000010...
19:39:03 SELECT pizza_category, CAST(SUM(total_price) AS DECIMAL(10,2)) as total_... 4 row(s) returned

```

E. % of Sales by Pizza Size

```

SELECT pizza_size, CAST(SUM(total_price) AS DECIMAL(10,2)) as total_revenue,
CAST(
SUM(total_price) * 100 / (SELECT SUM(total_price) from pizza_sales) AS DECIMAL(10,2)) AS PCT
FROM pizza_sales
GROUP BY pizza_size
ORDER BY pizza_size

```

MySQL Workbench Screenshot showing a query results grid for the pizza_sales table. The grid displays five rows of data:

pizza_size	total_revenue	PCT
L	375318.70	46.89
M	249392.25	30.49
S	178076.50	21.77
XL	14076.00	1.72
XXL	1008.60	0.12

The Action Output pane shows the execution log for the query:

```

Time Action Response Duration / Fetch Time
19:39:23 SELECT WEEK(STR_TO_DATE(order_date, '%d-%m-%Y'), 3) AS WeekNum... 0.07 sec / 0.000023...
19:39:37 SELECT WEEK(STR_TO_DATE(order_date, '%d-%m-%Y'), 3) AS WeekNum... 0.07 sec / 0.000015...
19:39:03 SELECT pizza_size, CAST(SUM(total_price) AS DECIMAL(10,2)) as total_... 4 row(s) returned
19:39:32 SELECT pizza_size, CAST(SUM(total_price) AS DECIMAL(10,2)) as total_re... 5 row(s) returned

```

F. Total Pizzas Sold by Pizza Category

```
SELECT pizza_category, SUM(quantity) as Total_Quantity_Sold  
FROM pizza_sales  
GROUP BY pizza_category  
ORDER BY Total_Quantity_Sold DESC;
```

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** The current schema is "nashville_housing".
- Tables:** The table "pizza_sales" is selected.
- Query Editor:** The SQL query is displayed:

```
1  SELECT pizza_category, SUM(quantity) as Total_Quantity_Sold  
2  FROM pizza_sales  
3  GROUP BY pizza_category  
4  ORDER BY Total_Quantity_Sold DESC;
```
- Result Grid:** The results show the total quantity sold for each category:

pizza_category	Total_Quantity_Sold
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050
- Action Output:** The log shows the execution details:

Time	Action	Response	Duration / Fetch Time
19:41:07	SELECT pizza_category, SUM(quantity) as Total_Quantity_Sold FROM pizza_sales	4 row(s) returned	0.030 sec / 0.000000...
19:42:47	SELECT pizza_category, SUM(quantity) as Total_Quantity_Sold FROM pizza_sales	4 row(s) returned	0.042 sec / 0.000009...
19:42:54	SELECT pizza_category, SUM(quantity) as Total_Quantity_Sold FROM pizza_sales	4 row(s) returned	0.035 sec / 0.000010...
19:43:21	SELECT pizza_category, SUM(quantity) as Total_Quantity_Sold FROM pizza_sales	4 row(s) returned	0.074 sec / 0.000007...

G. Top 5 Pizzas by Revenue

```
SELECT pizza_name, SUM(total_price) AS Total_Revenue  
FROM pizza_sales  
GROUP BY pizza_name  
ORDER BY Total_Revenue DESC  
LIMIT 5
```

The screenshot shows the MySQL Workbench interface with a dark theme. In the top-left pane, the 'Schemas' section is open, showing a tree structure with 'Projects' selected, followed by 'Tables' which includes 'maintable_SCH8U' and 'nashville housing d...', and finally 'pizza_sales'. The main query editor window contains the following SQL code:

```

1 • SELECT pizza_name, SUM(total_price) AS Total_Revenue
2   FROM pizza_sales
3   GROUP BY pizza_name
4   ORDER BY Total_Revenue DESC
5   LIMIT 5

```

The results grid displays the following data:

pizza_name	Total_Revenue
The Thai Chicken Pizza	45434.25
The Barbecue Chicken Pizza	42768.5
The Hawaiian Pizza	42590.5
The Classic Deluxe Pizza	36180.5
The Spicy Italian Pizza	34831.25

The status bar at the bottom left says 'Query Completed'.

H. Bottom 5 Pizzas by Revenue

```

SELECT pizza_name, SUM(total_price) AS Total_Revenue
FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_Revenue ASC
LIMIT 5

```

I. Top 5 Pizzas by Quantity

The screenshot shows the MySQL Workbench interface with a dark theme. In the top-left pane, the 'Schemas' section is open, showing a tree structure with 'Projects' selected, followed by 'Tables' which includes 'maintable_SCH8U' and 'nashville housing d...', and finally 'pizza_sales'. The main query editor window contains the following SQL code:

```

1 • SELECT pizza_name, SUM(total_price) AS Total_Revenue
2   FROM pizza_sales
3   GROUP BY pizza_name
4   ORDER BY Total_Revenue ASC
5   LIMIT 5

```

The results grid displays the following data:

pizza_name	Total_Revenue
The Brie Carré Pizza	11588499999999999
The Green Goddess Pizza	11588499999999999
The Pepper Supreme Pizza	152777.75
The Mediterranean Pizza	15360.5
The Spinach Pesto Pizza	15596

The status bar at the bottom left says 'Query Completed'.

The screenshot shows the MySQL Workbench interface with two separate queries running in different panes.

Top Query:

```

1 SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold
2 FROM pizza_sales
3 GROUP BY pizza_name
4 ORDER BY Total_Pizza_Sold ASC
5 LIMIT 5
    
```

Result Grid:

pizza_name	Total_Pizza_Sold
The Brie Carna Pizza	400
The Mediterranean Pizza	934
The Calabrese Pizza	937
The Spinach Supreme Pizza	950
The Sopressata Pizza	961

Action Output:

Time	Action	Response	Duration / Fetch Time
19:44:29	SELECT pizza_name, SUM(total_price) AS total_revenue FROM pizza_sales...	5 row(s) returned	0.076 sec / 0.000009...
19:45:28	SELECT pizza_name, SUM(total_price) AS Total_Revenue FROM pizza_sales...	5 row(s) returned	0.074 sec / 0.000012...
19:47:22	SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold FROM pizza_sales...	5 row(s) returned	0.085 sec / 0.000000...
19:48:00	SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold FROM pizza_sales...	5 row(s) returned	0.085 sec / 0.000000...

Bottom Query:

```

1 SELECT pizza_name, SUM(total_price) AS total_revenue FROM pizza_sales...
2 GROUP BY pizza_name
3 ORDER BY total_revenue DESC
4 LIMIT 5
    
```

Result Grid:

pizza_name	Total_Pizza_Sold
The Classic Deluxe Pizza	2453
The Hawaiian Pizza	2040
The Margherita Pizza	2042
The Pepperoni Pizza	2018
The Thai Chicken Pizza	2371

Action Output:

Time	Action	Response	Duration / Fetch Time
19:44:16	SELECT pizza_name, SUM(total_price) AS total_revenue FROM pizza_sales...	5 row(s) returned	0.076 sec / 0.000009...
19:44:25	SELECT pizza_name, SUM(total_price) AS Total_Revenue FROM pizza_sales...	5 row(s) returned	0.076 sec / 0.000009...
19:45:28	SELECT pizza_name, SUM(total_price) AS Total_Revenue FROM pizza_sales...	5 row(s) returned	0.074 sec / 0.000012...
19:47:22	SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold FROM pizza_sales...	5 row(s) returned	0.085 sec / 0.000000...

```

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

J. Bottom 5 Pizzas by Quantity

```

SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold
FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_Pizza_Sold ASC
LIMIT 5
    
```

K. Top 5 Pizzas by Total Orders

```
SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_Orders
FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_Orders DESC
LIMIT 5
```

The screenshot shows the MySQL Workbench interface. In the center, a query editor window displays the following SQL code:

```
1 • SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_Orders
2   FROM pizza_sales
3   GROUP BY pizza_name
4   ORDER BY Total_Orders DESC
5   LIMIT 5;
```

Below the query editor is a result grid showing the output of the query:

pizza_name	Total...
The Classic Deluxe Pizza	2329
The Hawaiian Pizza	2280
The Pepperoni Pizza	2278
The Sausage Chicken Pizza	2070
The Tzatziki Chicken Pizza	2225

At the bottom of the interface, the action output pane shows the execution log:

Time	Action	Response	Duration / Fetch Time
19:40:20	SELECT pizza_name, SUM(order_price) AS Total_Revenue FROM pizza_sales...	5 row(s) returned	0.004 sec / 0.000012...
19:47:22	SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold FROM pizza_sales...	5 row(s) returned	0.085 sec / 0.00000...
19:48:00	SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold FROM pizza_sales...	5 row(s) returned	0.085 sec / 0.00000...
19:49:11	SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_Orders FROM pizz...	5 row(s) returned	0.101 sec / 0.000008...

L. Bottom 5 Pizzas by Total Orders

```
SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_Orders
FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_Orders ASC
LIMIT 5
```

MySQL Workbench

Projects (projects)

Administration Schemas Data Cleaning Portfolio Project Queries* pizza_sales Context Help Snippets

Filter objects

SCHEMAS

projects

Tables

Views

Stored Procedures

Functions

sys

1 SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_Orders
2 FROM pizza_sales
3 GROUP BY pizza_name
4 ORDER BY Total_Orders ASC
5 LIMIT 5

Don't Limit

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid Filter Rows: Search Export:

pizza_name	Total...
The Brie Carre Pizza	480
The Mediterranean Pizza	912
The Spinach Supreme Pizza	918
The Calabrese Pizza	918
The Chicken Pesto Pizza	938

Object Info Session

Columns:

Object Info	Session
pizza_id	int
order_id	int
pizza_name_id	text
quantity	int
order_date	text
order_time	text
unit_price	double
total_price	double
pizza_size	text
pizza_category	text
pizza_ingredients	text
pizza_name	text

Action Output

Time	Action	Response	Duration / Fetch Time
19:47:22	SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold FROM pizza_sales...	5 row(s) returned	0.000 sec / 0.00000...
19:48:00	SELECT pizza_name, SUM(quantity) AS Total_Pizza_Sold FROM pizza_sales...	5 row(s) returned	0.085 sec / 0.00000...
19:49:11	SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_Orders FROM pizza...	5 row(s) returned	0.101 sec / 0.000008...
19:49:52	SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_Orders FROM pizza...	5 row(s) returned	0.092 sec / 0.000007...

Result 127 Read Only

Query Completed