Project: Pizza Sales Analysis using SQL

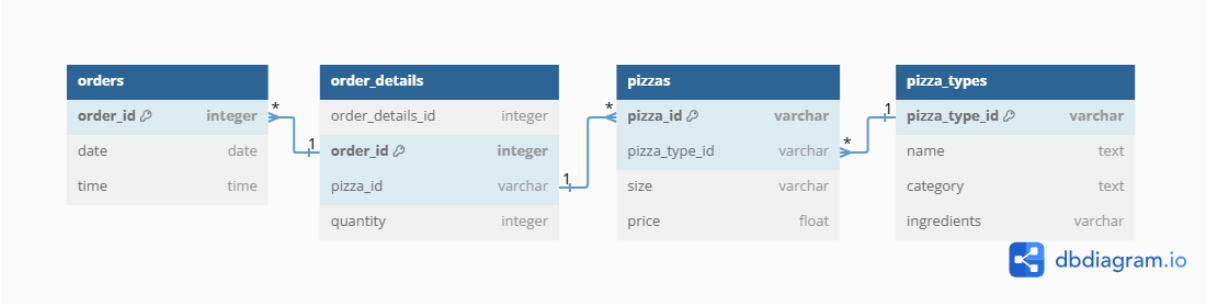
Introduction

This project aims to investigate the sales data of a pizza restaurant using SQL. By analyzing various aspects of the sales data, such as sales volume, customer preferences, and product performance, we can gain valuable insights into the restaurant's business operations and identify areas for improvement.

Objectives

- Analyze sales trends: Identify patterns in sales data over time, including peak sales periods and seasonal variations.
- Evaluate product popularity: Determine the most and least popular pizza items and their corresponding sales volumes.
- Examine customer behavior: Analyze customer purchasing habits, such as average order size, frequency of visits, and preferred payment methods.
- Identify revenue drivers: Identify the key factors contributing to the restaurant's revenue, including high-margin products and successful marketing campaigns.
- Provide recommendations: Offer data-driven recommendations to improve sales, enhance customer satisfaction, and optimize business operations.

Tables and Relationships



Questions

Basic level:

- Retrieve the total number of orders placed.

```sql
1.Retrieve the total number of orders placed.
SELECT
    COUNT(ORDER_ID) AS TOTAL_ORDERS
FROM
    ORDERS;
```

Data Output | Messages | Notifications

	total_orders bigint
1	21350

- Calculate the total revenue generated from pizza sales.

```sql
--Calculate the total revenue generated from pizza sales.
SELECT
    ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE), 2) AS REVENUE
FROM
    ORDER_DETAILS
    JOIN PIZZAS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID;
```

Data Output | Messages | Notifications

	revenue numeric
1	817860.05

- Identify the highest-priced pizza.

```sql
--Identify the highest-priced pizza.
SELECT PIZZA_ID, MAX(PRICE)
FROM PIZZAS
GROUP BY PIZZA_ID
ORDER BY MAX(PRICE) DESC
LIMIT 1;
```

Data Output | Messages | Notifications

	pizza_id character varying	max numeric
1	the_greek_xxl	35.95

- Identify the most common pizza size ordered.

```sql
--A4. Identify the most common pizza size ordered.
SELECT PIZZAS.SIZE, COUNT(ORDER_DETAILS.ORDER_ID) AS FREQ_SIZE
FROM ORDER_DETAILS
JOIN PIZZAS
ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
GROUP BY PIZZAS.SIZE
ORDER BY FREQ_SIZE DESC
LIMIT 1;
```

Data Output | Messages | Notifications

	size character varying	freq_size bigint
1	L	18526

- List the top 5 most ordered pizza types along with their quantities.

```sql
--A5. List the top 5 most ordered pizza types along with their quantities.
SELECT PIZZA_TYPES.NAME, SUM(ORDER_DETAILS.QUANTITY) AS QUANTITY
FROM ORDER_DETAILS
JOIN PIZZAS
ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
JOIN PIZZA_TYPES
ON PIZZAS.PIZZA_TYPE_ID = PIZZA_TYPES.PIZZA_TYPE_ID
GROUP BY PIZZA_TYPES.NAME
ORDER BY QUANTITY DESC
LIMIT 5;
```

Data Output | Messages | Notifications

	name character varying	quantity numeric
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

Intermediate:

- Join the necessary tables to find the total quantity of each pizza category ordered.

```sql
--B1. Join the necessary tables to find the total quantity of each pizza category ordered.
SELECT PIZZA_TYPES.CATEGORY, SUM(ORDER_DETAILS.QUANTITY) AS QUANTITY
FROM ORDER_DETAILS
JOIN PIZZAS
ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
JOIN PIZZA_TYPES
ON PIZZAS.PIZZA_TYPE_ID = PIZZA_TYPES.PIZZA_TYPE_ID
GROUP BY PIZZA_TYPES.CATEGORY
ORDER BY QUANTITY DESC;
```

Data Output | Messages | Notifications

	category character varying	quantity numeric
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
4	Chicken	11050

- Determine the distribution of orders by hour of the day.

```sql
--B2. Determine the distribution of orders by hour of the day.
SELECT EXTRACT(HOUR
FROM TIME) AS HOUR,
COUNT(ORDER_ID) AS ORDER_FREQ
FROM ORDERS
GROUP BY HOUR
ORDER BY HOUR;
```

Data Output | Messages | Notifications

	hour numeric	order_freq bigint
1	9	1
2	10	8
3	11	1231
4	12	2520
5	13	2455

- Join relevant tables to find the category-wise distribution of pizzas.

```sql
--B3. Join relevant tables to find the category-wise distribution of pizzas.
SELECT CATEGORY, COUNT(PIZZA_TYPE_ID) AS FREQ
FROM PIZZA_TYPES
GROUP BY CATEGORY
ORDER BY FREQ;
```

Data Output | Messages | Notifications

	category character varying	freq bigint
1	Chicken	6
2	Classic	8
3	Supreme	9
4	Veggie	9

- Group the orders by date and calculate the average number of pizzas ordered per day.

```sql
--B4. Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT ORDERS.DATE AS DAY, SUM(ORDER_DETAILS.QUANTITY) AS ORDER
FROM ORDERS
JOIN ORDER_DETAILS ON ORDERS.ORDER_ID = ORDER_DETAILS.ORDER_ID
GROUP BY DAY
ORDER BY DAY;

SELECT ROUND(AVG(QUANTITY), 0) AS AVG_ORDER
FROM
(SELECT ORDERS.DATE AS DAY, SUM(ORDER_DETAILS.QUANTITY) AS QUANTITY
```

Data Output | Messages | Notifications

	day date	order numeric
1	2015-01-01	162
2	2015-01-02	165
3	2015-01-03	158
4	2015-01-04	106
5	2015-01-05	125

```sql
SELECT ROUND(AVG(QUANTITY), 0) AS AVG_ORDER
FROM
(SELECT ORDERS.DATE AS DAY, SUM(ORDER_DETAILS.QUANTITY) AS QUANTITY
FROM ORDERS JOIN ORDER_DETAILS
ON ORDERS.ORDER_ID = ORDER_DETAILS.ORDER_ID
GROUP BY DAY
ORDER BY DAY) AS ORDER_BY_DAY;
```

Data Output | Messages | Notifications

	avg_order numeric
1	138

- Determine the top 3 most ordered pizza types based on revenue.

```
204
205  --B5. Determine the top 3 most ordered pizza types based on revenue.
206  SELECT PIZZA_TYPES.NAME AS NAME,
207  SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE) AS REVENUE
208  FROM
209  ORDER_DETAILS
210  JOIN PIZZAS
211  ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
212  JOIN PIZZA_TYPES
213  ON PIZZAS.PIZZA_TYPE_ID = PIZZA_TYPES.PIZZA_TYPE_ID
214  GROUP BY NAME
215  ORDER BY REVENUE DESC
216  LIMIT 3;
217
```

Data Output | Messages | Notifications

	name character varying	revenue numeric
1	The Thai Chicken Pizza	43434.25
2	The Barbecue Chicken Pizza	42768.00
3	The California Chicken Pizza	41409.50

Advanced:

- Calculate the percentage contribution of each pizza type to total revenue.

Query | Query History

```
185  --C1. Calculate the percentage contribution of each pizza type to total revenue.
186  SELECT PIZZA_TYPES.CATEGORY AS CAT,
187      ROUND((SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE)
188      / (SELECT ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE), 2)
189      AS TOTAL_SALES
190      FROM PIZZA_TYPES
191      JOIN PIZZAS
192      ON PIZZA_TYPES.PIZZA_TYPE_ID = PIZZAS.PIZZA_TYPE_ID
193      JOIN ORDER_DETAILS
194      ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID)) * 100, 2) AS REV
195  FROM
196      PIZZAS
197      JOIN PIZZA_TYPES ON PIZZAS.PIZZA_TYPE_ID = PIZZA_TYPES.PIZZA_TYPE_ID
198      JOIN ORDER_DETAILS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
199      GROUP BY CAT
200      ORDER BY REV DESC;
```

Data Output | Messages | Notifications

	cat character varying	rev numeric
1	Classic	26.91
2	Supreme	25.46
3	Chicken	23.96

- Analyze the cumulative revenue generated over time.

```sql
-- C2. Analyze the cumulative revenue generated over time.
SELECT DATE, REV, SUM(REV) OVER (ORDER BY DATE) AS CUMSUM
FROM
(SELECT ORDERS.DATE AS DATE,
SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE) AS REV
FROM ORDER_DETAILS
JOIN ORDERS ON ORDERS.ORDER_ID = ORDER_DETAILS.ORDER_ID
JOIN PIZZAS ON PIZZAS.PIZZA_ID = ORDER_DETAILS.PIZZA_ID
GROUP BY DATE
ORDER BY DATE) AS A;
```

Data Output | Messages | Notifications

	date date	rev numeric	cumsum numeric
1	2015-01-01	2713.85	2713.85
2	2015-01-02	2731.90	5445.75
3	2015-01-03	2662.40	8108.15

- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```sql
--C3. Determine the top 3 most ordered pizza types based on revenue for each pizza category.
SELECT CAT, NAME, REV, RN
FROM
(SELECT CAT, NAME, REV,
RANK() OVER (PARTITION BY CAT ORDER BY REV DESC) AS RN
FROM
(SELECT PIZZA_TYPES.CATEGORY AS CAT, PIZZA_TYPES.NAME AS NAME,
SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE) AS REV
FROM ORDER_DETAILS
JOIN PIZZAS ON PIZZAS.PIZZA_ID = ORDER_DETAILS.PIZZA_ID
JOIN PIZZA_TYPES ON PIZZAS.PIZZA_TYPE_ID = PIZZA_TYPES.PIZZA_TYPE_ID
GROUP BY CAT, NAME
ORDER BY REV) AS A
) AS B
WHERE RN <= 3;
```

Data Output | Messages | Notifications

	cat character varying	name character varying	rev numeric	rn bigint
1	Chicken	The Thai Chicken Pizza	43434.25	1
2	Chicken	The Barbecue Chicken Pizza	42768.00	2
3	Chicken	The California Chicken Pizza	41409.50	3