SQL Case Study on Roll’S Sales

SQL Data Analysis Project

By Vyankatesh Shinde
Date- 19/07/2024
Contact - Vdshinde962@gmail.com

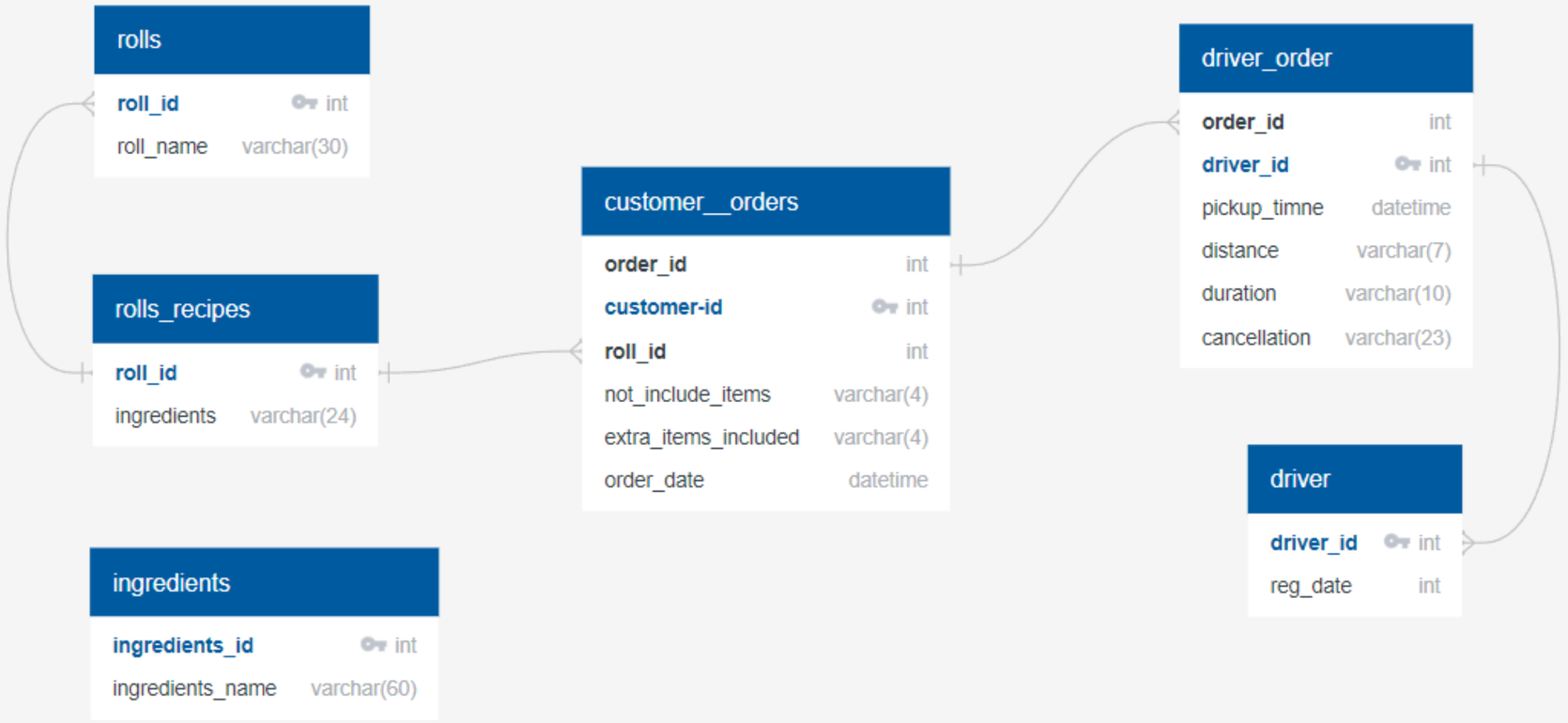
Introduction

Roll’s sales analysis for business decision making

1. Understanding customer preferences is a key for successful veg & non veg roll sales
2. Managing inventory effectively crucial to avoid shortages during peak sales periods
3. Analyzing delivery service to find out steps to be taken for faster and effective deliveries

Schemas Used

Database schemas used for analysis of roll’s sales



Data Cleaning

customer_orders & driver_order table had inconsistent data so it needed to be cleaned before we start analytics.

Driver_order Table with inconsistent data in cancellation, distance, duration, pickup_time column

order_id	driver_id	pickup_time	distance	duration	cancellation
1	1	2021-01-01 18:15:34	20km	32 minutes	
2	1	2021-01-01 19:10:54	20km	27 minutes	
3	1	2021-01-03 00:12:37	13.4km	20 mins	NaN
4	2	2021-01-04 13:53:03	23.4	40	NaN
5	3	2021-01-08 21:10:57	10	15	NaN
6	3	NULL	NULL	NULL	Cancellation
7	2	2020-01-08 21:30:45	25km	25mins	NULL
8	2	2020-01-10 00:15:02	23.4 km	15 minute	NULL
9	2	NULL	NULL	NULL	Customer Cancellation
10	1	2020-01-11 18:50:20	10km	10minutes	NULL

Customer_orders table with inconsistent data in not_include_items, extra_items_included columns.

order_id	customer_id	roll_id	not_include_items	extra_items_included	order_date
1	101	1			2021-01-01 18:05:02
2	101	1			2021-01-01 19:00:52
3	102	1			2021-01-02 23:51:23
3	102	2		NaN	2021-01-02 23:51:23
4	103	1	4		2021-01-04 13:23:46
4	103	1	4		2021-01-04 13:23:46
4	103	2	4		2021-01-04 13:23:46
5	104	1	NULL	1	2021-01-08 21:00:29
6	101	2	NULL	NULL	2021-01-08 21:03:13
7	105	2	NULL	1	2021-01-08 21:20:29
8	102	1	NULL	NULL	2021-01-09 23:54:33

Data Cleaning

Copying all the data of customer_orders to the new table for cleaning operations.

```sql
-- Firstly we need to clean the Customer_orders table
drop table if exists customer_orders1;
CREATE TABLE customer_orders1 AS (SELECT order_id,
    customer_id,
    roll_id,
    not_include_items,
    extra_items_included,
    order_date FROM
    customer_orders);

UPDATE customer_orders1
SET
    not_include_items = CASE
        WHEN not_include_items = 'null' THEN NULL
        WHEN not_include_items = ' ' THEN NULL
        ELSE not_include_items
    END,
    extra_items_included = CASE
        WHEN extra_items_included = 'null' THEN NULL
        WHEN extra_items_included = ' ' THEN NULL
        WHEN extra_items_included = 'NaN' THEN NULL
        ELSE extra_items_included
    END;
     -- end of cleaning process of customer_orders table
```

order_id	customer_id	roll_id	not_include_items	extra_items_included	order_date
1	101	1	NULL	NULL	2021-01-01 18:05:02
2	101	1	NULL	NULL	2021-01-01 19:00:52
3	102	1	NULL	NULL	2021-01-02 23:51:23
3	102	2	NULL	NULL	2021-01-02 23:51:23
4	103	1	4	NULL	2021-01-04 13:23:46
4	103	1	4	NULL	2021-01-04 13:23:46
4	103	2	4	NULL	2021-01-04 13:23:46
5	104	1	NULL	1	2021-01-08 21:00:29
6	101	2	NULL	NULL	2021-01-08 21:03:13
7	105	2	NULL	1	2021-01-08 21:20:29
8	102	1	NULL	NULL	2021-01-09 23:54:33
9	103	1	4	1,5	2021-01-10 11:22:59
10	104	1	NULL	NULL	2021-01-11 18:34:49
10	104	1	2,6	1,4	2021-01-11 18:34:49

Data Cleaning

Copying all the data of driver_order to the new table for cleaning operations.

```
-- cleaning the customer_orders table
drop table if exists driver_order1;
create table driver_order1 as
(select order_id, driver_id, pickup_time,
case
when distance like '%km' then trim('km' from distance)
else distance
end as distance,
case
when duration like '%minutes' then trim('minutes' from duration)
when duration like '%minute' then trim('minute' from duration)
when duration like '%mins' then trim('mins' from duration)
when duration like '%km' then trim('km' from duration)
when duration like ' ' then NULL
else duration
end as duration,
cancellation from driver_order);
update driver_order1
set
cancellation = case
cancellation when "null" then null
             when "nan" then null
             when " " then null
             else cancellation
             end;
select * from driver_order1;
select * from customer_orders1;
-- end of cleaning process of driver_order table
```

order_id	driver_id	pickup_time	distance	duration	cancellation
1	1	2021-01-01 18:15:34	20	32	NULL
2	1	2021-01-01 19:10:54	20	27	NULL
3	1	2021-01-03 00:12:37	13.4	20	NULL
4	2	2021-01-04 13:53:03	23.4	40	NULL
5	3	2021-01-08 21:10:57	10	15	NULL
6	3	NULL	NULL	NULL	Cancellation
7	2	2021-01-08 21:30:45	25	25	NULL
8	2	2021-01-10 00:15:02	23.4	15	NULL
9	2	NULL	NULL	NULL	Customer Cancellation
10	1	2021-01-11 18:50:20	10	10	NULL

Roll’s Sales

Exploring the potential of rolls in the market today

Roll Metrics Analysis

1. How many rolls were ordered?
2. How many unique customer orders were made?
3. How many successful orders were delivered by each delivery person?
4. How many each type of roll was delivered?
5. How many veg and non veg rolls were ordered by each customer?
6. What was the maximum number of rolls delivered in a single order?
7. for each customer, how many delivered rolls had at least one change and how many had no change?
8. How many rolls were delivered that had both exclusions and extras?
9. What was the total volume of rolls ordered for each hour of the day?
10. What was the volume of orders for each day of the week?

Roll’s Sales

Roll Orders Metrics Analysis

```sql
-- 1. How many rolls were ordered?
SELECT
    COUNT(order_id) AS total_rolls_ordered
FROM
    customer_orders;
```

total_rolls_ordered
14

```sql
-- 2. How many unique customer orders were made?
SELECT
    COUNT(DISTINCT order_id) AS unique_orders
FROM
    customer_orders;
```

unique_orders
10

```sql
-- 3. How many successful orders were delivered by each delivery person?
SELECT
    driver_id, COUNT(order_id) AS succesful_orders
FROM
    driver_order1
WHERE
    duration IS NOT NULL
GROUP BY driver_id;
```

driver_id	succesful_orders
1	4
2	3
3	1

Roll’s Sales

Roll Orders Metrics Analysis

```sql
-- 4. How many each type of roll was delivered?
SELECT
    roll_id, COUNT(c.order_id) AS total_orders
FROM
    driver_order1 AS d
        JOIN
    customer_orders AS c ON d.order_id = c.order_id
WHERE
    duration IS NOT NULL
GROUP BY roll_id;
```

roll_id	total_orders
1	9
2	3

```sql
-- 5. How many veg and non veg rolls were ordered by each customer
SELECT
    a.*, r.roll_name
FROM
    (SELECT
        customer_id, roll_id, COUNT(roll_id) AS count
    FROM
        customer_orders1
    GROUP BY customer_id , roll_id
    ORDER BY customer_id) a
        INNER JOIN
    rolls r ON a.roll_id = r.roll_id;
```

customer_id	roll_id	count	roll_name
101	1	2	Non Veg Roll
102	1	2	Non Veg Roll
103	1	3	Non Veg Roll
104	1	3	Non Veg Roll
101	2	1	Veg Roll
102	2	1	Veg Roll
103	2	1	Veg Roll
105	2	1	Veg Roll

Roll’s Sales

Roll Orders Metrics Analysis

```sql
-- 6. What was the maximum number of rolls delivered in a single order?
SELECT
    d.order_id, COUNT(c.roll_id) AS total_rolls
FROM
    driver_order AS d
        JOIN
    customer_orders AS c ON d.order_id = c.order_id
WHERE
    duration IS NOT NULL
GROUP BY d.order_id
ORDER BY total_rolls DESC
LIMIT 1;
```

order_id	total_rolls
4	3

```sql
-- 7. for each customer, how many delivered rolls had at least one change and how many had no change?
SELECT
    c.customer_id,
    SUM(CASE
        WHEN
            (not_include_items IS NOT NULL
                AND extra_items_included IS NULL)
                OR (not_include_items IS NULL
                AND extra_items_included IS NOT NULL)
        THEN
            1
        ELSE 0
    END) AS AtLeastNoChange,
    SUM(CASE
        WHEN
            (not_include_items IS NULL
                AND extra_items_included IS NULL)
        THEN
            1
        ELSE 0
    END) AS NoChange
FROM
    customer_orders1 AS c
        JOIN
    driver_order1 AS d ON c.order_id = d.order_id
WHERE
    d.duration IS NOT NULL
GROUP BY customer_id;
```

customer_id	AtLeastNoChange	NoChange
101	0	2
102	0	3
103	3	0
104	1	1
105	1	0

Roll’s Sales

Roll Orders Metrics Analysis

```sql
-- 8. How many rolls were delivered that had both exclusions and extras?
SELECT
    c.customer_id,
    SUM(CASE
        WHEN
            (not_include_items IS NOT NULL
                AND extra_items_included IS NOT NULL)
        THEN
            1
        ELSE 0
    END) AS ExtrasAndExclusions
FROM
    customer_orders1 AS c
        JOIN
    driver_order1 AS d ON c.order_id = d.order_id
WHERE
    d.duration IS NOT NULL
GROUP BY customer_id
ORDER BY ExtrasAndExclusions DESC;
```

customer_id	AtLeastNoChange	NoChange
101	0	2
102	0	3
103	3	0
104	1	1
105	1	0

Roll’s Sales

Roll Orders Metrics Analysis

```sql
-- 9. What was the total volume of rolls ordered for each hour of the day?

SELECT
    EXTRACT(HOUR FROM order_date) AS HourlyData,
    COUNT(order_id) AS TotalOrders
FROM
    customer_orders1
GROUP BY HourlyData
ORDER BY TotalOrders DESC;
```

HourlyData	TotalOrders
18	3
23	3
13	3
21	3
19	1
11	1

```sql
-- 10. What was the volume of orders for each day of the week?

SELECT
    DAYNAME(order_date) AS days, COUNT(order_id) AS TotalOrders
FROM
    customer_orders1
GROUP BY days
ORDER BY days;
```

days	TotalOrders
Friday	5
Monday	5
Saturday	3
Sunday	1

Roll’s Sales

Exploring the potential of rolls in the market today

Customer & Delivery Experience analysis

1. What was the average time in minutes it took for each driver to arrive to pick up the order?
2. Is there any relationship between the number of rolls and how long the order takes to prepare? How many successful orders were delivered by each delivery person?
3. What was the average distance traveled for each customer?
4. What was the difference between the longest and shortest delivery times for all orders?
5. What was the average speed for each driver for each delivery and do you notice any trend for these values?
6. What is the successful delivery percentage for each driver?
7. What are the standard ingredients for each roll?
8. If a Meat Lovers pizza costs $12 and Vegetarian costs $10 and there were no charges for changes — how much money has Pizza Runner made so far if there are no delivery fees?
9. Generate an order item for each record in the customers orders table.

Roll’s Sales

Customer & Delivery Experience analysis

```sql
-- 1. What was the average time in minutes it took for each driver to arrive to pick up the order?

select driver_id, avg(timediff)
from
(select driver_id,timestampdiff(minute, order_date, pickup_time) as timediff from
customer_orders1 as c join driver_order1 as d
on c.order_id = d.order_id
where duration is not null) as subquery
group by driver_id;
```

driver_id	avg(timediff)
1	15.3333
2	23.4000
3	10.0000

```sql
-- 2. Is there any relationship between the number of rolls and how long the order takes to prepare?
SELECT
    subquery.order_id,
    COUNT(subquery.order_id) AS order_count,
    SUM(subquery.timetaken) AS total_timetaken
FROM
    (SELECT
        c.order_id,
        TIMESTAMPDIFF(MINUTE, c.order_date, d.pickup_time) AS timetaken
     FROM
        customer_orders1 AS c
     JOIN
        driver_order1 AS d
     ON
        c.order_id = d.order_id
     WHERE
        d.duration IS NOT NULL
    ) AS subquery
GROUP BY
    subquery.order_id;
```

order_id	order_count	total_timetaken
1	1	10
2	1	10
3	2	42
4	3	87
5	1	10
7	1	10
8	1	20
10	2	30

Roll’s Sales

Customer & Delivery Experience analysis

```sql
-- 3. What was the average distance traveled for each customer?

SELECT
    customer_id, ROUND(AVG(distance), 2) AvgDistanceTravelled
FROM
    customer_orders1 AS c
        JOIN
    driver_order1 AS d ON c.order_id = d.order_id
WHERE
    d.duration IS NOT NULL
GROUP BY customer_id;
```

customer_id	AvgDistanceTravelled
101	20
102	16.73
103	23.4
104	10
105	25

```sql
-- 4. What was the difference between the longest and shortest delivery times for all orders?

SELECT
    (MAX(subquery2.ordertime) - MIN(subquery2.ordertime)) AS TimeDiffMin
FROM
    (SELECT
        subquery1.order_id,
            TIMESTAMPDIFF(MINUTE, subquery1.order_date, subquery1.pickup_time) AS ordertime
    FROM
        (SELECT
        c.order_id, order_date, pickup_time
    FROM
        customer_orders1 AS c
    JOIN driver_order1 AS d ON c.order_id = d.order_id
    WHERE
        d.duration IS NOT NULL) AS subquery1) AS subquery2;
```

TimeDiffMin
19

Roll’s Sales

Customer & Delivery Experience analysis

```sql
-- 5. What was the average speed for each driver for each delivery and do you notice any trend for these values?

SELECT
    *,
    ROUND((subquery1.distance * 60 / subquery1.duration),
            2) AS speedkmh
FROM
    (SELECT
        c.order_id, driver_id, distance, duration
    FROM
        customer_orders1 AS c
    JOIN driver_order1 AS d ON c.order_id = d.order_id
    WHERE
        duration IS NOT NULL) AS subquery1;

-- from the results we can analyze that there is no trend but average speed(KMPH) of each driver is as follows for delivering the orders
-- driverId 1 = 47.06
-- driverId 2 = 51.78
-- driverId 3 = 40
```

order_id	driver_id	distance	duration	speedkmh
1	1	20	32	37.5
2	1	20	27	44.44
3	1	13.4	20	40.2
3	1	13.4	20	40.2
4	2	23.4	40	35.1
4	2	23.4	40	35.1
4	2	23.4	40	35.1
5	3	10	15	40
7	2	25	25	60
8	2	23.4	15	93.6
10	1	10	10	60
10	1	10	10	60

```sql
-- 6.  What is the successful delivery percentage for each driver?
SELECT
    *,
    ROUND(((subquery1.OrdersDelivered * 100) / TotalOrders),
            2) AS SuccDelPer
FROM
    (SELECT driver_id,
            SUM(CASE WHEN
                    cancellation = 'Cancellation'
                        OR cancellation = 'Customer Cancellation'
                THEN 1
                ELSE 1
            END) AS TotalOrders,
            SUM(CASE WHEN
                    cancellation = 'Customer Cancellation'
                        OR cancellation = 'Cancellation'
                THEN 0
                ELSE 1
            END) AS OrdersDelivered
    FROM
        driver_order1
    GROUP BY driver_id) AS subquery1;
```

driver_id	TotalOrders	OrdersDelivered	SuccDelPer
1	4	4	100.00
2	4	3	75.00
3	2	1	50.00

Roll’s Sales

Customer & Delivery Experience analysis

```sql
-- 7. What are the standard ingredients for each roll?

-- to get the standard ingridents for each roll we have to clean the rolls_recipies table first

drop table if exists rolls_recipies1;
CREATE TABLE rolls_recipies1 (
    roll_id INT,
    ingredients_id INT
);
insert into rolls_recipies1(roll_id, ingredients_id) values
(1,1),(1,2),(1,3),(1,4),(1,5),(1,6),(1,8),(1,10),
(2,4),(2,6),(2,7),(2,9),(2,11),(2,12);

SELECT
    roll_id, GROUP_CONCAT(ingredients_name) as all_ingredients
FROM
    rolls_recipies1 AS r
        INNER JOIN
    ingredients AS i ON i.ingredients_id = r.ingredients_id
GROUP BY roll_id;
```

roll_id	all_ingredients
1	BBQ Chicken,Chilli Sauce,Chicken,Cheese,Keba...
2	Cheese,Mushrooms,Onions,Peppers,Tomatoes,...

```sql
-- 8. If a Meat Lovers pizza costs $12 and Vegetarian costs $10 and there were no charges for changes,
--    how much money has Pizza Runner made so far if there are no delivery fees?

    select sum(case
    when c.roll_id= 1 then 12
    else 10
    end) as TotalAmount
    from driver_order as d inner join
    customer_orders as c
    on c.order_id = d.order_id
    where d.duration is not null;
```

TotalAmount
138

Roll’s Sales

Customer & Delivery Experience analysis

```sql
-- 9. Generate an order item for each record in the customers_orders table in the format of one of the following:
	-- Meat Lovers
	-- Meat Lovers - Exclude Beef
	-- Meat Lovers - Extra Bacon
	-- Meat Lovers - Exclude Cheese, Bacon - Extra Mushroom, Peppers
	-- veg lovers

		drop table if exists roll_customizations;
		CREATE TABLE roll_customizations (
			roll_id INT,
			exclusions VARCHAR(255),
			extras VARCHAR(255),
			order_item_name VARCHAR(255)
			);
			INSERT INTO roll_customizations (roll_id, exclusions, extras, order_item_name) VALUES
			(1, NULL, NULL, 'Meat Lovers'),
			(2, NULL, NULL, 'Veg Lovers'),
			(2, '4', NULL, 'Veg Lovers - Exclude Cheese'),
			(1, '4', NULL, 'Meat Lovers - Exclude Cheese'),
			(1, '3', NULL, 'Meat Lovers - Exclude Beef'),
			(1, NULL, '1', 'Meat Lovers - Extra Bacon'),
			(1, '1,4', '6,9', 'Meat Lovers - Exclude Cheese, Bacon - Extra Mushroom, Peppers'),
			(1, '2,6', '1,4', 'Meat Lovers - Exclude BBQ Sauce, Mushroom - Extra Bacon, Cheese'),
			(1, '4', '1,5', 'Meat Lovers - Exclude Cheese - Extra Bacon, Chicken');

			SELECT
				c.order_id,
				c.roll_id,
				c.customer_id,
				c.not_include_items,
				c.extra_items_included,
				r.roll_name,
				rc.order_item_name AS order_item
			FROM
				customer_orders1 AS c
					INNER JOIN
				rolls AS r ON r.roll_id = c.roll_id
					LEFT JOIN
				roll_customizations AS rc ON c.roll_id = rc.roll_id
					AND (rc.exclusions = c.not_include_items
					OR rc.exclusions IS NULL
					AND c.not_include_items IS NULL)
					AND (rc.extras = c.extra_items_included
					OR rc.extras IS NULL
					AND c.extra_items_included IS NULL);
```

roll_id	customer_id	not_include_items	extra_items_included	roll_name	order_item
1	101	NULL	NULL	Non Veg Roll	Meat Lovers
1	101	NULL	NULL	Non Veg Roll	Meat Lovers
1	102	NULL	NULL	Non Veg Roll	Meat Lovers
2	102	NULL	NULL	Veg Roll	Veg Lovers
1	103	4	NULL	Non Veg Roll	Meat Lovers - Exclude Cheese
1	103	4	NULL	Non Veg Roll	Meat Lovers - Exclude Cheese
2	103	4	NULL	Veg Roll	Veg Lovers - Exclude Cheese
1	104	NULL	1	Non Veg Roll	Meat Lovers - Extra Bacon
2	101	NULL	NULL	Veg Roll	Veg Lovers
2	105	NULL	1	Veg Roll	NULL
1	102	NULL	NULL	Non Veg Roll	Meat Lovers
1	103	4	1,5	Non Veg Roll	Meat Lovers - Exclude Chees...
1	104	NULL	NULL	Non Veg Roll	Meat Lovers
1	104	2,6	1,4	Non Veg Roll	Meat Lovers - Exclude BBQ S...

SQL Concepts Used -

1. DDL, DML Commands
2. Sub Query
3. Case Statement
4. Group by, Order by Statement
5. Aggregation Functions
6. SQL Joins

Conclusion

In this project, we conducted a comprehensive analysis of the sales data from a company specializing in veg and non-veg rolls. Our primary goal was to uncover insights that could enhance operational aspects, including:

- Inventory management
- Customer preference tracking
- Product delivery management
- Customer experience
- Pricing strategies

Key Insights and Trends

- Identified popular roll types and ingredient combinations
- Highlighted peak ordering times and customer behavior patterns
- Provided actionable recommendations for inventory optimization
- Suggested improvements for delivery efficiency and customer service