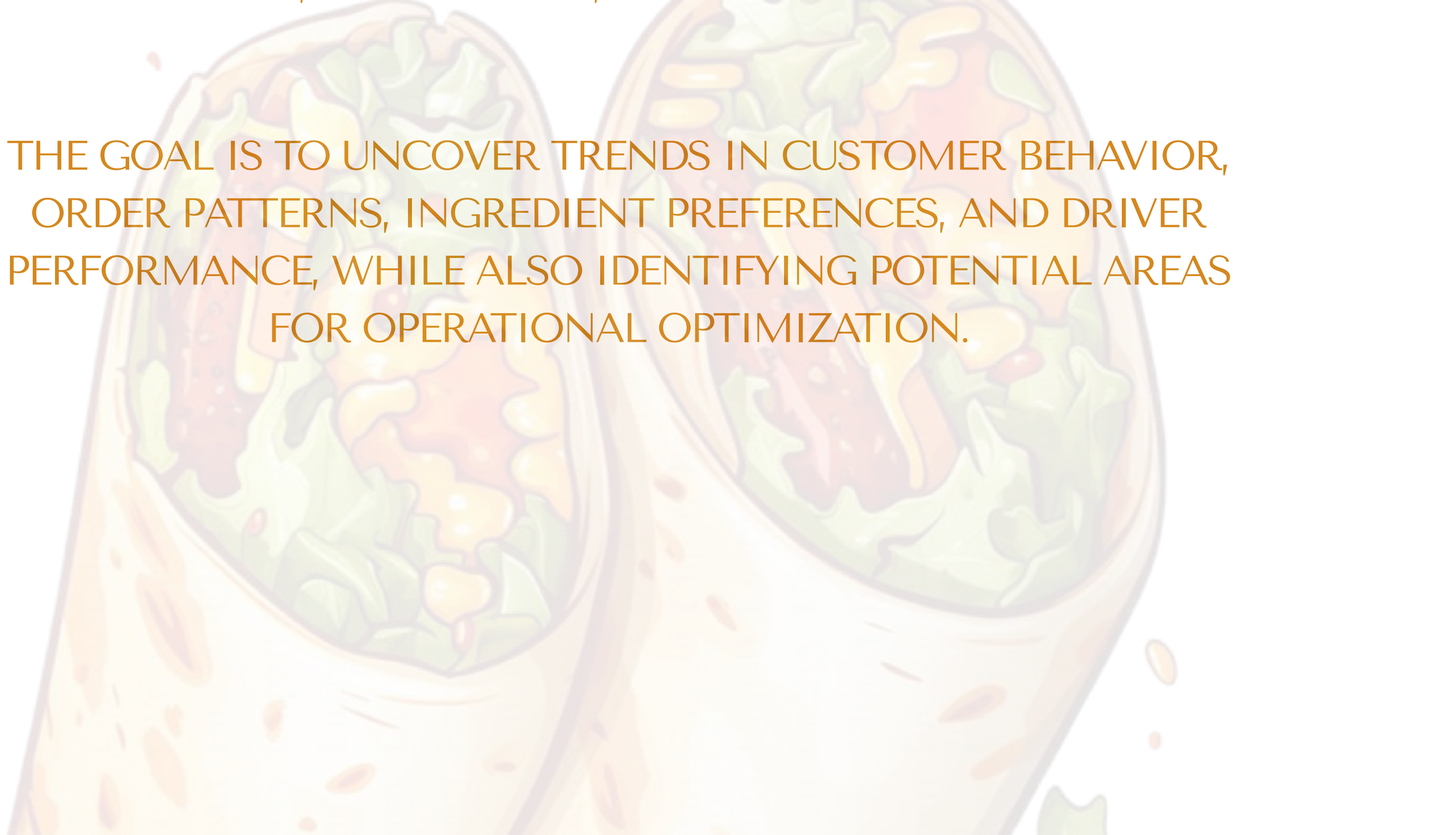
"ROLL ANALYTICS: SQL BASED FOOD DELIVERY DATA INSIGHT"

DIVE IN

PROJECT OVERVIEW: ROLL ANALYTICS – SQL-BASED FOOD DELIVERY DATA INSIGHTS

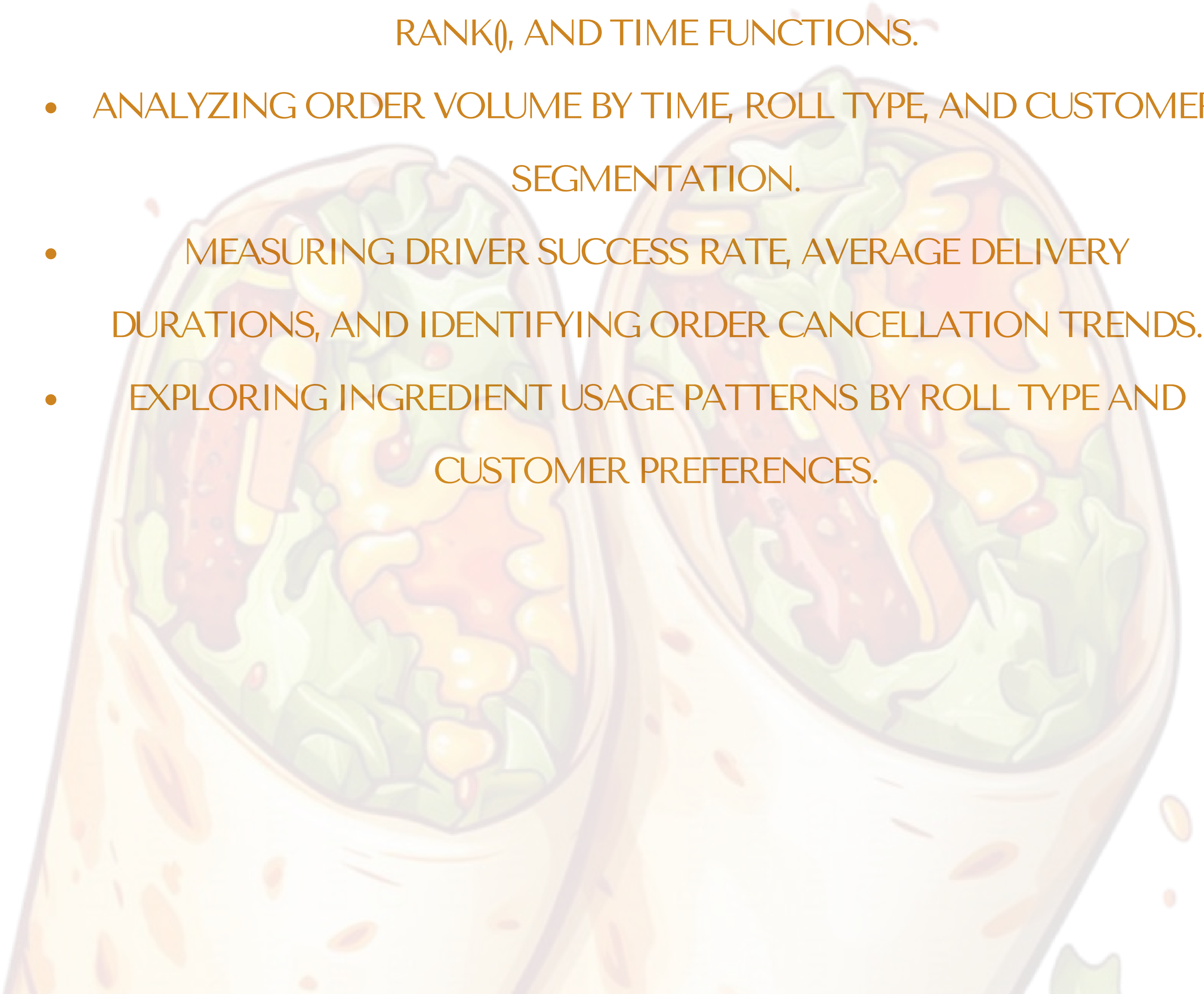
THIS PROJECT PRESENTS A COMPREHENSIVE ANALYSIS OF A FICTIONAL FOOD DELIVERY SERVICE SPECIALIZING IN CUSTOMIZABLE ROLLS. USING STRUCTURED SQL QUERIES, IT EXTRACTS MEANINGFUL BUSINESS INSIGHTS FROM MULTIPLE RELATIONAL DATASETS REPRESENTING CUSTOMERS, DRIVERS, ORDERS, INGREDIENTS, AND DELIVERY METRICS.

THE GOAL IS TO UNCOVER TRENDS IN CUSTOMER BEHAVIOR, ORDER PATTERNS, INGREDIENT PREFERENCES, AND DRIVER PERFORMANCE, WHILE ALSO IDENTIFYING POTENTIAL AREAS FOR OPERATIONAL OPTIMIZATION.

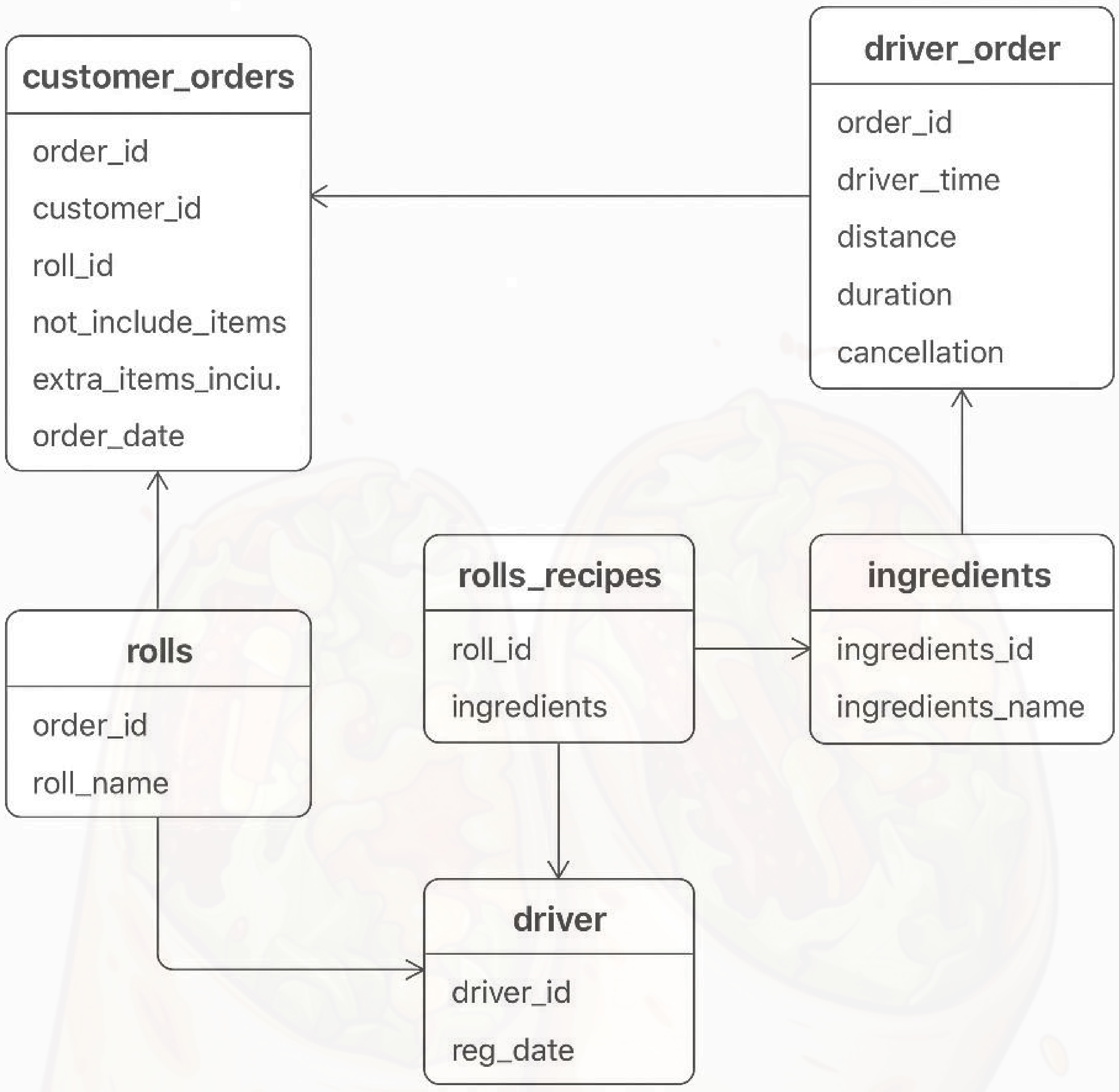


KEY ASPECTS OF THE PROJECT INCLUDE:

- DESIGNING AND POPULATING NORMALIZED TABLES SUCH AS CUSTOMER_ORDERS, DRIVER_ORDER, ROLLS, AND INGREDIENTS.
- HANDLING MESSY DATA (E.G., INCONSISTENT CANCELLATION VALUES) AND ENSURING DATA QUALITY FOR ANALYSIS.
- WRITING COMPLEX SQL QUERIES USING JOINS, GROUP BY, CASE, RANK(), AND TIME FUNCTIONS.
- ANALYZING ORDER VOLUME BY TIME, ROLL TYPE, AND CUSTOMER SEGMENTATION.
- MEASURING DRIVER SUCCESS RATE, AVERAGE DELIVERY DURATIONS, AND IDENTIFYING ORDER CANCELLATION TRENDS.
- EXPLORING INGREDIENT USAGE PATTERNS BY ROLL TYPE AND CUSTOMER PREFERENCES.



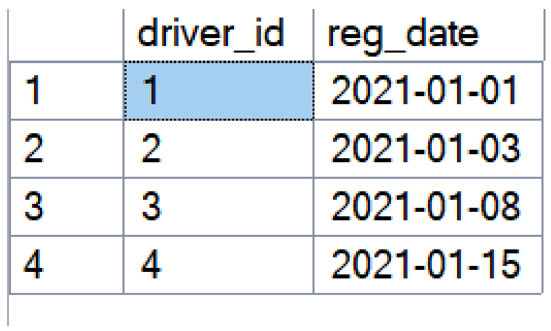
ENTITY RELATIONSHIP DIAGRAM (ERD)



DROP AND CREATE TABLE

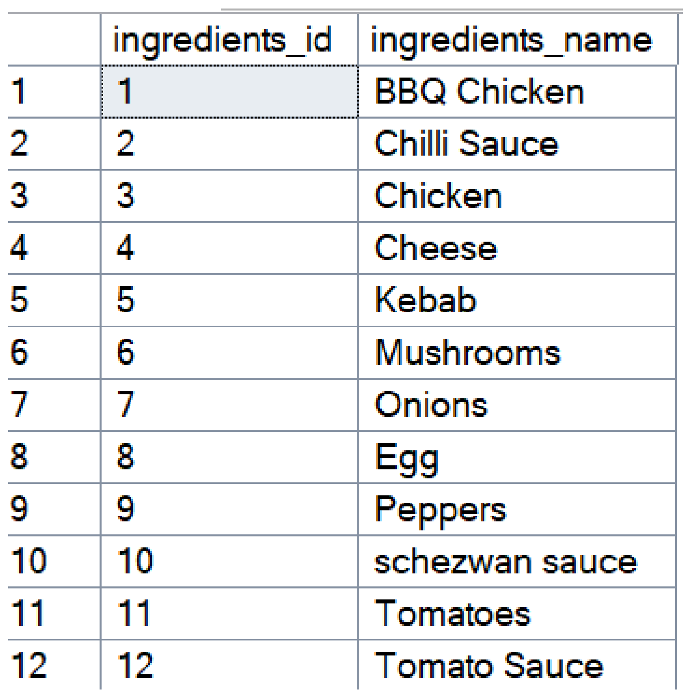
RESULT

```
drop table if exists driver;
CREATE TABLE driver(driver_id integer,reg_date date);

INSERT INTO driver(driver_id,reg_date)
 VALUES (1,'01-01-2021'),
(2,'01-03-2021'),
(3,'01-08-2021'),
(4,'01-15-2021');
```



	driver_id	reg_date
1	1	2021-01-01
2	2	2021-01-03
3	3	2021-01-08
4	4	2021-01-15

```
drop table if exists ingredients;
CREATE TABLE ingredients(ingredients_id integer,ingredients_name varchar(60));

INSERT INTO ingredients(ingredients_id ,ingredients_name)
 VALUES (1,'BBQ Chicken'),
(2,'Chilli Sauce'),
(3,'Chicken'),
(4,'Cheese'),
(5,'Kebab'),
(6,'Mushrooms'),
(7,'Onions'),
(8,'Egg'),
(9,'Peppers'),
(10,'schezwan sauce'),
(11,'Tomatoes'),
(12,'Tomato Sauce');
```



	ingredients_id	ingredients_name
1	1	BBQ Chicken
2	2	Chilli Sauce
3	3	Chicken
4	4	Cheese
5	5	Kebab
6	6	Mushrooms
7	7	Onions
8	8	Egg
9	9	Peppers
10	10	schezwan sauce
11	11	Tomatoes
12	12	Tomato Sauce

DROP AND CREATE TABLE

RESULT

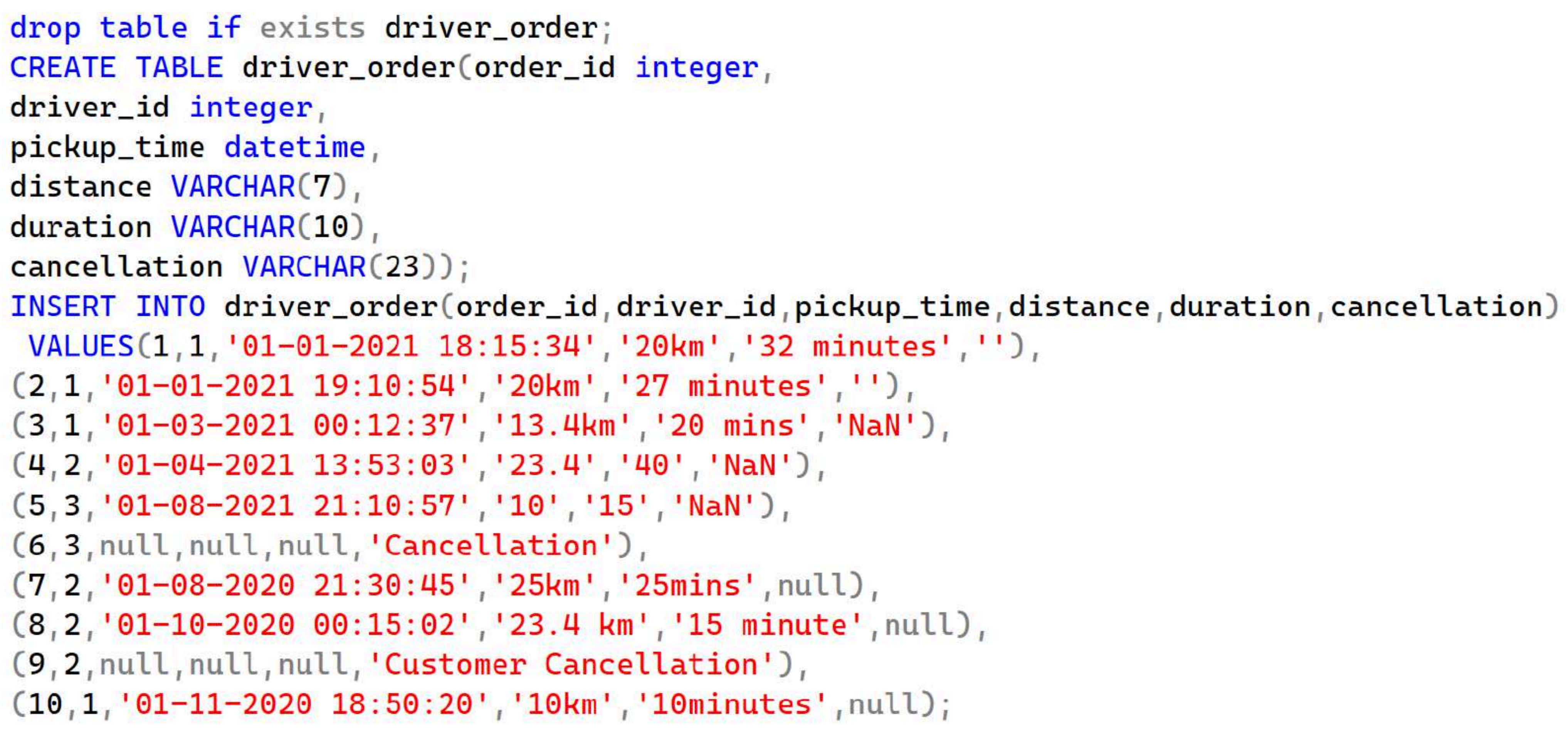
```
drop table if exists rolls;
CREATE TABLE rolls(roll_id integer,roll_name varchar(30));

INSERT INTO rolls(roll_id ,roll_name)
 VALUES (1   ,'Non Veg Roll'),
(2   ,'Veg Roll');
```

	roll_id	roll_name
1	1	Non Veg Roll
2	2	Veg Roll

```
drop table if exists rolls_recipes;
CREATE TABLE rolls_recipes(roll_id integer,ingredients varchar(24));

INSERT INTO rolls_recipes(roll_id ,ingredients)
 VALUES (1,'1,2,3,4,5,6,8,10'),
(2,'4,6,7,9,11,12');
```

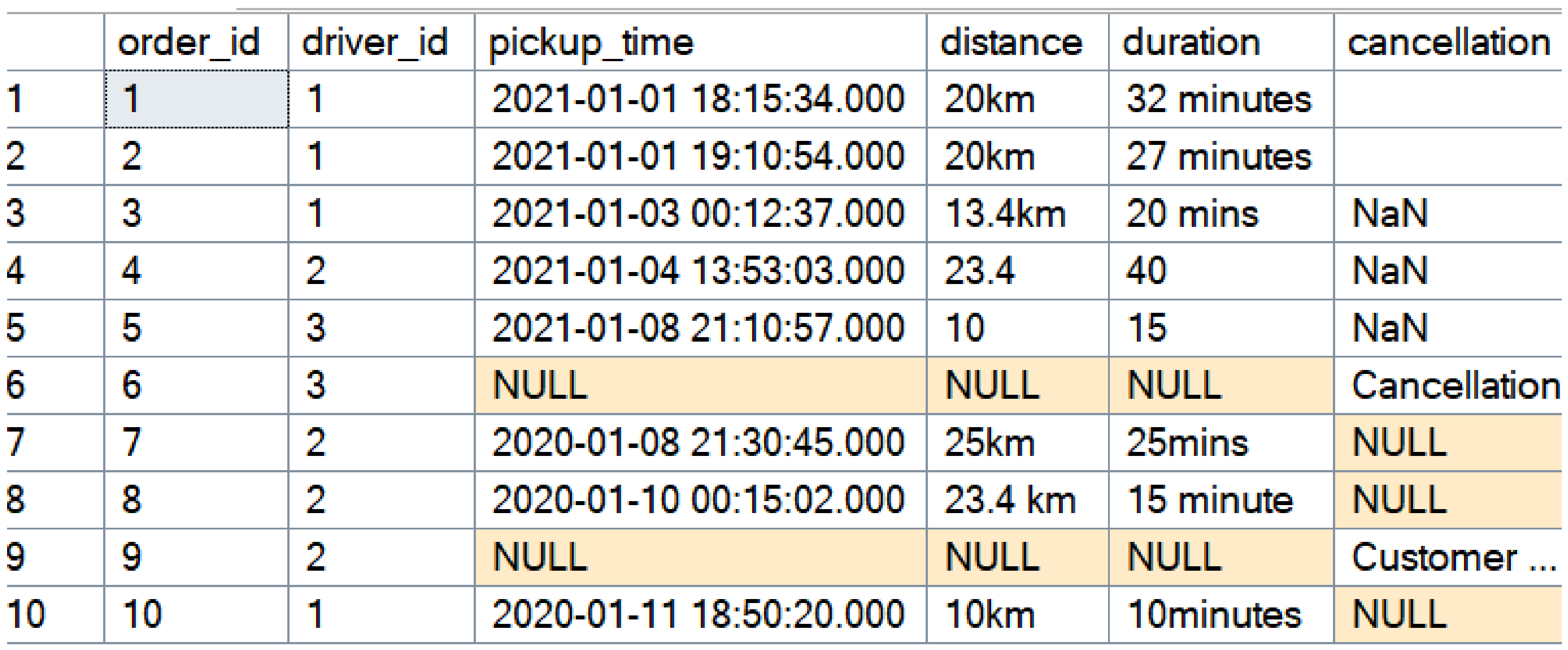
	roll_id	ingredients
1	1	1,2,3,4,5,6,8,10
2	2	4,6,7,9,11,12

DROP AND CREATE TABLE



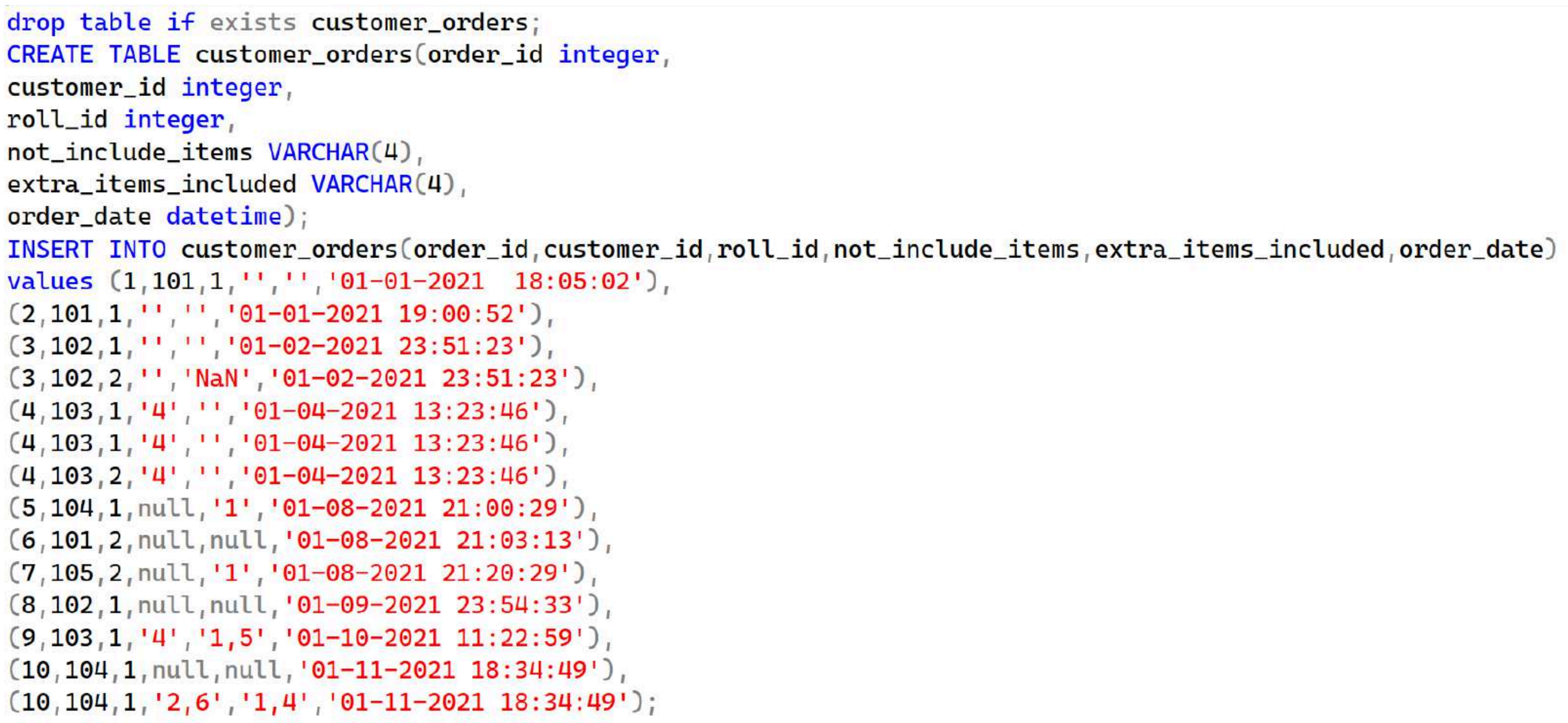
```sql
drop table if exists driver_order;
CREATE TABLE driver_order(order_id integer,
driver_id integer,
pickup_time datetime,
distance VARCHAR(7),
duration VARCHAR(10),
cancellation VARCHAR(23));
INSERT INTO driver_order(order_id,driver_id,pickup_time,distance,duration,cancellation)
 VALUES(1,1,'01-01-2021 18:15:34','20km','32 minutes',''),
(2,1,'01-01-2021 19:10:54','20km','27 minutes',''),
(3,1,'01-03-2021 00:12:37','13.4km','20 mins','NaN'),
(4,2,'01-04-2021 13:53:03','23.4','40','NaN'),
(5,3,'01-08-2021 21:10:57','10','15','NaN'),
(6,3,null,null,null,'Cancellation'),
(7,2,'01-08-2020 21:30:45','25km','25mins',null),
(8,2,'01-10-2020 00:15:02','23.4 km','15 minute',null),
(9,2,null,null,null,'Customer Cancellation'),
(10,1,'01-11-2020 18:50:20','10km','10minutes',null);
```

RESULT



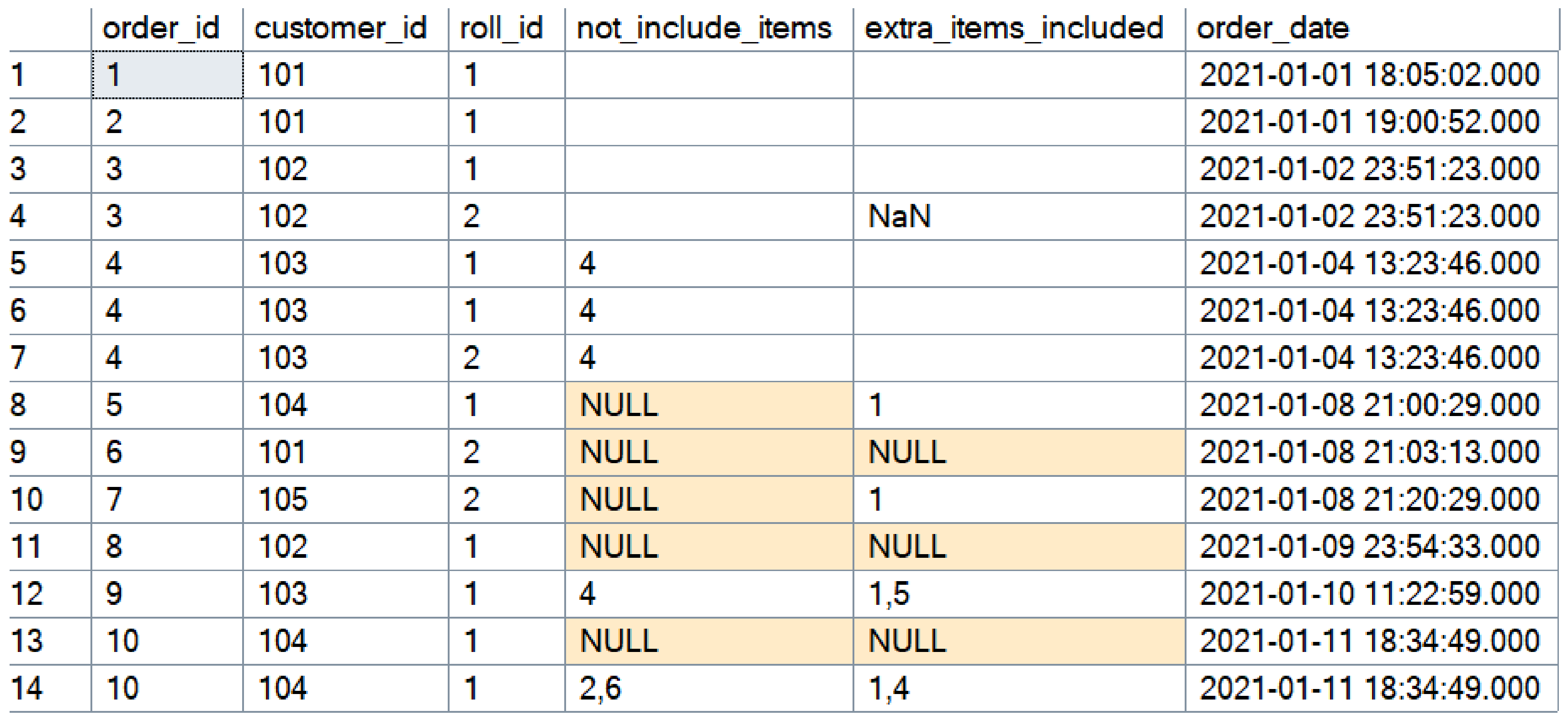
	order_id	driver_id	pickup_time	distance	duration	cancellation
1	1	1	2021-01-01 18:15:34.000	20km	32 minutes	
2	2	1	2021-01-01 19:10:54.000	20km	27 minutes	
3	3	1	2021-01-03 00:12:37.000	13.4km	20 mins	NaN
4	4	2	2021-01-04 13:53:03.000	23.4	40	NaN
5	5	3	2021-01-08 21:10:57.000	10	15	NaN
6	6	3	NULL	NULL	NULL	Cancellation
7	7	2	2020-01-08 21:30:45.000	25km	25mins	NULL
8	8	2	2020-01-10 00:15:02.000	23.4 km	15 minute	NULL
9	9	2	NULL	NULL	NULL	Customer ...
10	10	1	2020-01-11 18:50:20.000	10km	10minutes	NULL

DROP AND CREATE TABLE



```sql
drop table if exists customer_orders;
CREATE TABLE customer_orders(order_id integer,
customer_id integer,
roll_id integer,
not_include_items VARCHAR(4),
extra_items_included VARCHAR(4),
order_date datetime);
INSERT INTO customer_orders(order_id,customer_id,roll_id,not_include_items,extra_items_included,order_date)
values (1,101,1,'','','01-01-2021  18:05:02'),
(2,101,1,'','','01-01-2021 19:00:52'),
(3,102,1,'','','01-02-2021 23:51:23'),
(3,102,2,'','NaN','01-02-2021 23:51:23'),
(4,103,1,'4','','01-04-2021 13:23:46'),
(4,103,1,'4','','01-04-2021 13:23:46'),
(4,103,2,'4','','01-04-2021 13:23:46'),
(5,104,1,null,'1','01-08-2021 21:00:29'),
(6,101,2,null,null,'01-08-2021 21:03:13'),
(7,105,2,null,'1','01-08-2021 21:20:29'),
(8,102,1,null,null,'01-09-2021 23:54:33'),
(9,103,1,'4','1,5','01-10-2021 11:22:59'),
(10,104,1,null,null,'01-11-2021 18:34:49'),
(10,104,1,'2,6','1,4','01-11-2021 18:34:49');
```

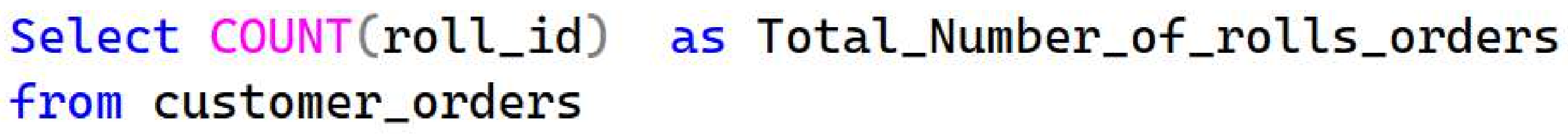
RESULT



	order_id	customer_id	roll_id	not_include_items	extra_items_included	order_date
1	1	101	1			2021-01-01 18:05:02.000
2	2	101	1			2021-01-01 19:00:52.000
3	3	102	1			2021-01-02 23:51:23.000
4	3	102	2		NaN	2021-01-02 23:51:23.000
5	4	103	1	4		2021-01-04 13:23:46.000
6	4	103	1	4		2021-01-04 13:23:46.000
7	4	103	2	4		2021-01-04 13:23:46.000
8	5	104	1	NULL	1	2021-01-08 21:00:29.000
9	6	101	2	NULL	NULL	2021-01-08 21:03:13.000
10	7	105	2	NULL	1	2021-01-08 21:20:29.000
11	8	102	1	NULL	NULL	2021-01-09 23:54:33.000
12	9	103	1	4	1,5	2021-01-10 11:22:59.000
13	10	104	1	NULL	NULL	2021-01-11 18:34:49.000
14	10	104	1	2,6	1,4	2021-01-11 18:34:49.000

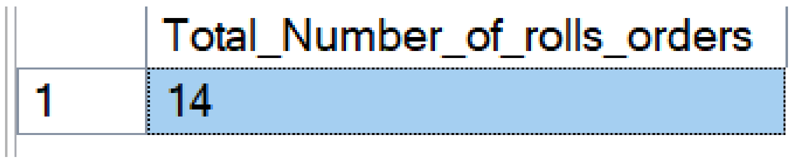
ROLL METRICS.

--1. HOW MANY ROLLS WERE ORDERED ?



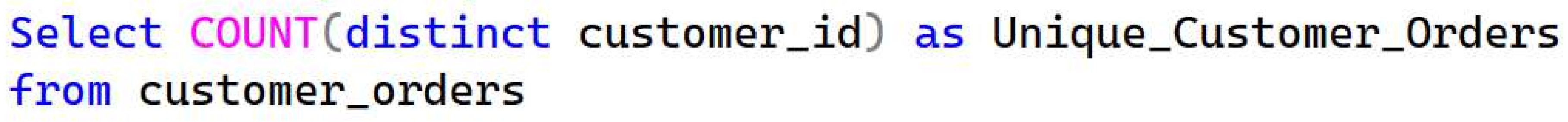
```
Select COUNT(roll_id)  as Total_Number_of_rolls_orders
from customer_orders
```

RESULT



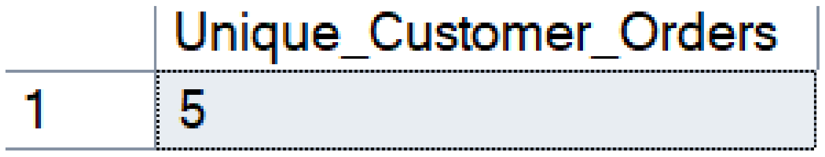
	Total_Number_of_rolls_orders
1	14

--2. HOW MANY UNIQUE CUSTOMER ORDER WERE MADE ?



```
Select COUNT(distinct customer_id) as Unique_Customer_Orders
from customer_orders
```

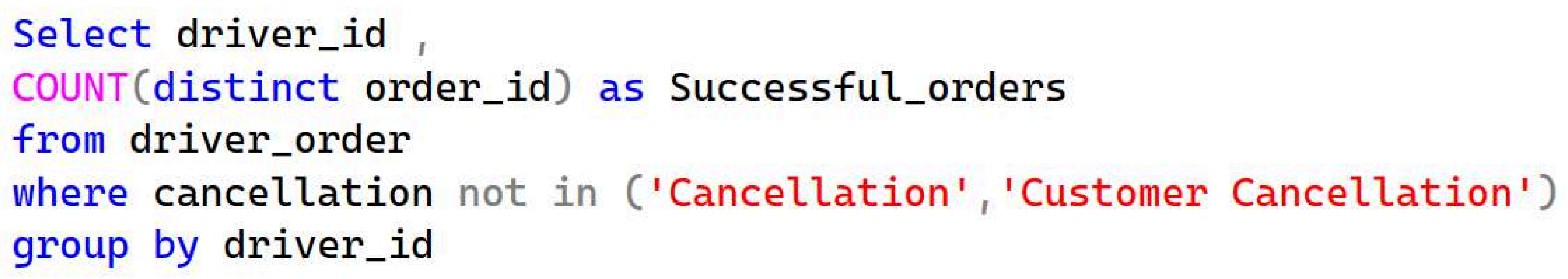
RESULT



	Unique_Customer_Orders
1	5

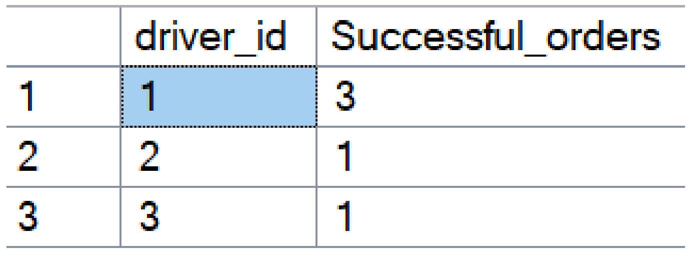
ROLL METRICS.

--3. HOW MANY SUCCESSFUL ORDERS WERE DELIVERED BY EACH DRIVER ?



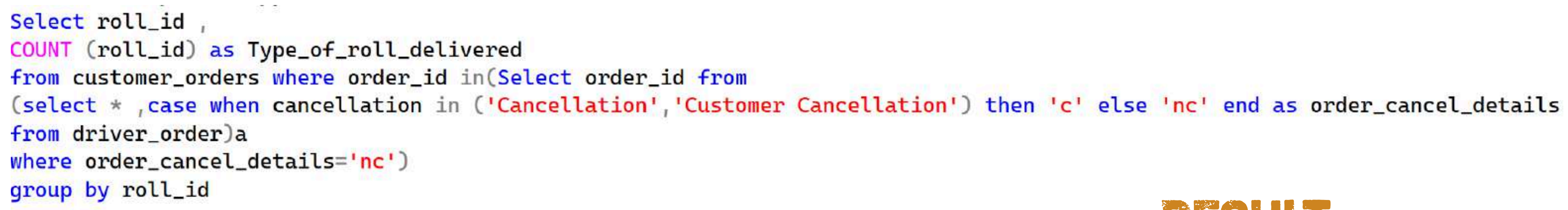
```
Select driver_id ,
COUNT(distinct order_id) as Successful_orders
from driver_order
where cancellation not in ('Cancellation','Customer Cancellation')
group by driver_id
```

RESULT



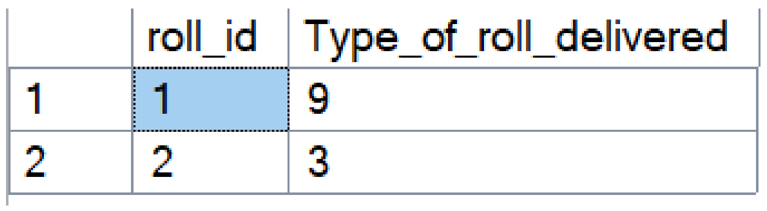
	driver_id	Successful_orders
1	1	3
2	2	1
3	3	1

--4. HOW MANY EACH TYPE OF ROLL WAS DELIVERED ?



```
Select roll_id ,
COUNT (roll_id) as Type_of_roll_delivered
from customer_orders where order_id in(Select order_id from
(select * ,case when cancellation in ('Cancellation','Customer Cancellation') then 'c' else 'nc' end as order_cancel_details
from driver_order)a
where order_cancel_details='nc')
group by roll_id
```

RESULT



	roll_id	Type_of_roll_delivered
1	1	9
2	2	3

ROLL METRICS.

--5. HOW MANY VEG AND NON-VEG ROLLS WERE ORDERED BY EACH CUSTOMER ?

```
Select a.*, b.roll_name
from (select customer_id, roll_id, COUNT(roll_id) Total_count
      from customer_orders
      group by customer_id,roll_id)a
inner join rolls b on a.roll_id = b.roll_id
```

RESULT

	customer_id	roll_id	Total_count	roll_name
1	101	1	2	Non Veg Roll
2	102	1	2	Non Veg Roll
3	103	1	3	Non Veg Roll
4	104	1	3	Non Veg Roll
5	101	2	1	Veg Roll
6	102	2	1	Veg Roll
7	103	2	1	Veg Roll
8	105	2	1	Veg Roll

--6. WHAT WAS THE MAXIMUM NUMBER OF ROLL DELIVERED IN A SINGLE ORDER ?

```
Select * from(
select *, RANK() over(order by cnt  desc) rnk from
(select order_id, COUNT(roll_id)cnt from(
select * from customer_orders where order_id in (
select order_id from(
    select * ,case when cancellation in ('Cancellation','Customer Cancellation') then 'c' else 'nc' end as order_cancel_details
    from driver_order)a
    where order_cancel_details='nc'))b
group by order_id)c)d where rnk =1
```

RESULT

	order_id	cnt	rnk
1	4	3	1

ROLL METRICS.

--7. WHAT WAS THE TOTAL NUMBER OF ROLLS ORDERS FOR EACH HOUR OF THE DAY ?

```sql
select hour_bucket, COUNT(hour_bucket) Total_no_roll_in_each_hour
from
    (select * , CONCAT(cast(datepart(hour,order_date)as varchar), '-',
    cast(datepart(hour,order_date)+1 as varchar)) hour_bucket
    from customer_orders)a
group by hour_bucket
```

RESULT

	hour_bucket	Total_no_roll_in_each_hour
1	11-12	1
2	13-14	3
3	18-19	3
4	19-20	1
5	21-22	3
6	23-24	3

--8. WHAT WAS THE NUMBER OF ORDER FOR EACH DAY OF THE WEEK ?

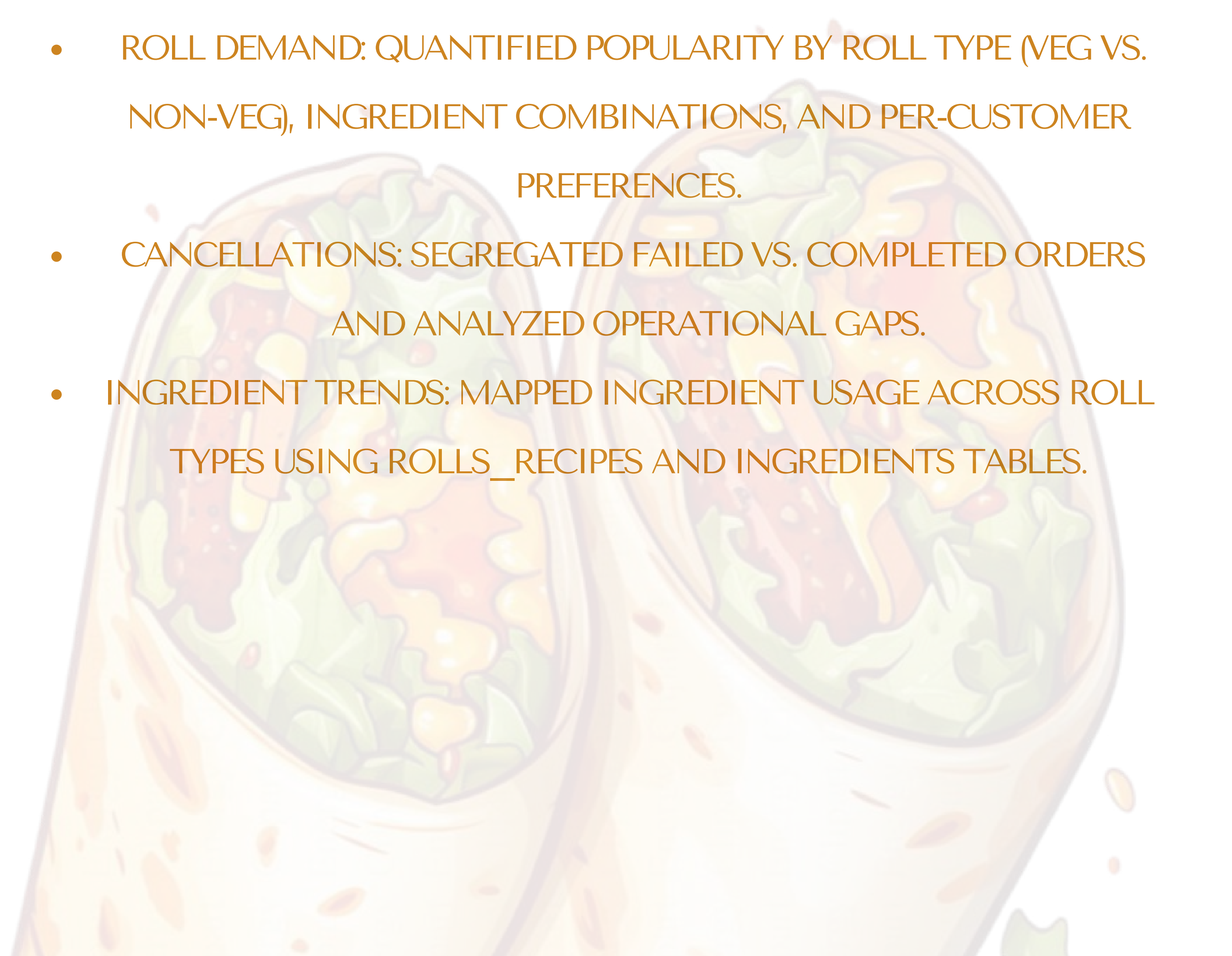
```sql
Select day_of_week,COUNT(distinct order_id) Number_of_order_in_each_day
from
    (select * , DATENAME(dw,order_date) day_of_week
    from customer_orders)a
group by  day_of_week
```

RESULT

	day_of_week	Number_of_order_in_each_day
1	Friday	5
2	Monday	2
3	Saturday	2
4	Sunday	1

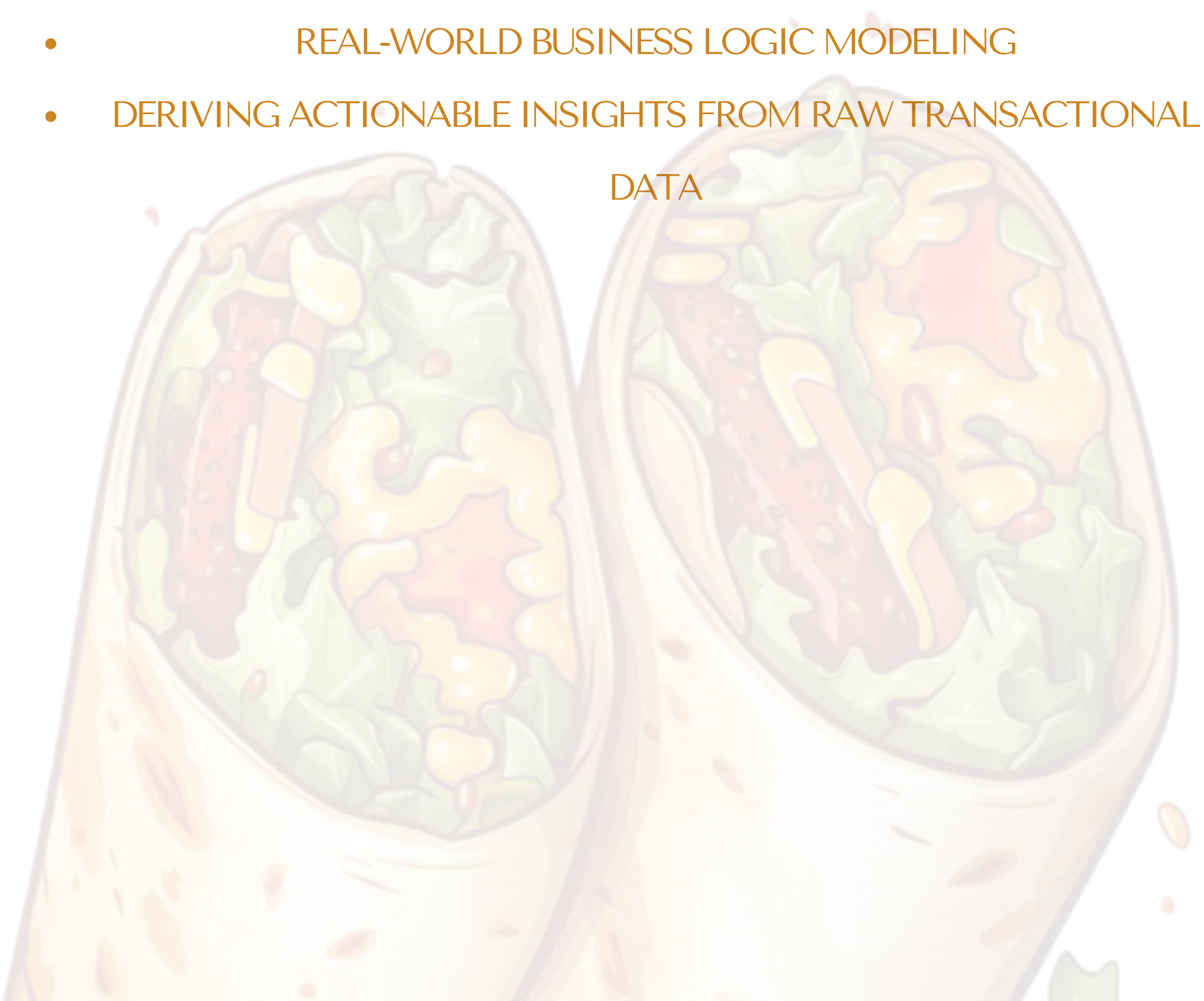
SUMMARY OF KEY INSIGHTS:

- DRIVER PERFORMANCE: IDENTIFIED SUCCESSFUL DELIVERY PATTERNS, WITH DETAILED METRICS ON DRIVER EFFICIENCY AND CANCELLATION TRENDS.
- ORDER TIMING ANALYSIS: FOUND PEAK ORDERING HOURS AND MOST ACTIVE DAYS OF THE WEEK USING SQL TIME FUNCTIONS.
- ROLL DEMAND: QUANTIFIED POPULARITY BY ROLL TYPE (VEG VS. NON-VEG), INGREDIENT COMBINATIONS, AND PER-CUSTOMER PREFERENCES.
- CANCELLATIONS: SEGREGATED FAILED VS. COMPLETED ORDERS AND ANALYZED OPERATIONAL GAPS.
- INGREDIENT TRENDS: MAPPED INGREDIENT USAGE ACROSS ROLL TYPES USING ROLLS_RECIPES AND INGREDIENTS TABLES.



SKILLS APPLIED & LEARNED:

- ADVANCED SQL QUERYING (JOIN , GROUP BY , RANK , CASE, DATEPART , DATENAME)
- DATA CLEANING AND HANDLING INCONSISTENCIES (NULL , 'NAN' , BLANK VALUES)
- RELATIONAL SCHEMA DESIGN AND USE OF FOREIGN KEYS
- REAL-WORLD BUSINESS LOGIC MODELING
- DERIVING ACTIONABLE INSIGHTS FROM RAW TRANSACTIONAL DATA



THANK YOU