DATAWITHDANNY.COM

START YOUR SQL ENGINES

PIZZA RUNNER

CASE STUDY #2

8 WEEK SQL CHALLENGE

8WEEKSQLCHALLENGE.COM

Introduction

Did you know that over 115 million kilograms of pizza is consumed daily worldwide??? (Well according to Wikipedia anyway…)

Danny was scrolling through his Instagram feed when something really caught his eye - "80s Retro Styling and Pizza Is The Future!"

Danny was sold on the idea, but he knew that pizza alone was not going to help him get seed funding to expand his new Pizza Empire - so he had one more genius idea to combine with it - he was going to *Uberize* it - and so Pizza Runner was launched!

Danny started by recruiting "runners" to deliver fresh pizza from Pizza Runner Headquarters (otherwise known as Danny's house) and also maxed out his credit card to pay freelance developers to build a mobile app to accept orders from customers.

Because Danny had a few years of experience as a datascientist-he was very aware that data collection was going to be critical for his business' growth.

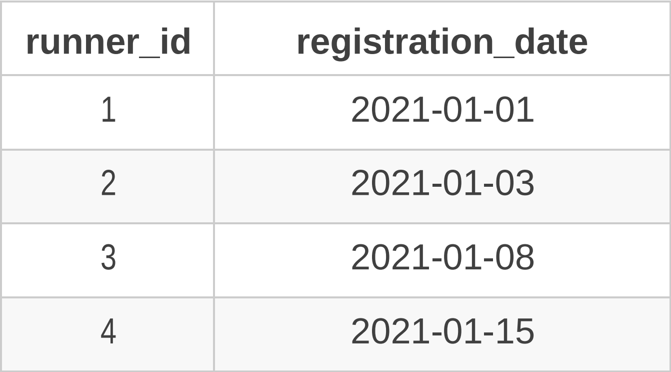
He has prepared for us an entity relationship diagram of his database design but requires further assistance to clean his data and apply some basic calculations so he can better direct his runners and optimize Pizza Runner's operations.

All datasets exist within the **pizza_runner** database schema - be sure to include this reference within your SQL scripts as you start exploring the data and answering the case study question.

DataSets

Table 1: runners

The runners table shows the **registration_date** for each new runner



runner_id	registration_date
1	2021-01-01
2	2021-01-03
3	2021-01-08
4	2021-01-15

Table 2: customer_orders

Customer pizza orders are captured in the **customer_orders** table with 1 row for each individual pizza that is part of the order.

The **pizza_id** relates to the type of pizza which was ordered whilst the **exclusions** are the **ingredient_id** values which should be removed from the pizza and the **extras** are the **ingredient_id** values which need to be added to the pizza.

Note that customers can order multiple pizzas in a single order with varying **exclusions** and **extras** values even if the pizza is the same type!

The**exclusions** and**extras** columnswillneedtobecleanedupbefore using them in yourqueries.

order_id	customer_id	pizza_id	exclusions	extras	order_time
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
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

Table 3: runner_orders

After each orders are received through the system - they are assigned to a runner - however not all orders are fully completed and can be cancelled by the restaurant or the customer.

The **pickup_time** is the timestamp at which the runner arrives at the Pizza Runner headquarters to pick up the freshly cooked pizzas.
The **distance** and **duration** fields are related to how far and long the runner had to travel to deliver the order to the respective customer.

There are some known data issues with this table so be careful when using this in your queries - make sure to check the data types for each column in the schema SQL!

order_id	runner_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	Restaurant 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

Table 4: pizza_names

At the moment - Pizza Runner only has 2 pizzas available the Meat Lovers or Vegetarian!

pizza_id	pizza_name
1	Meat Lovers
2	Vegetarian

Table 5: pizza_recipes

Each **pizza_id** has a standard set of **toppings** which are used as part of the pizza recipe.

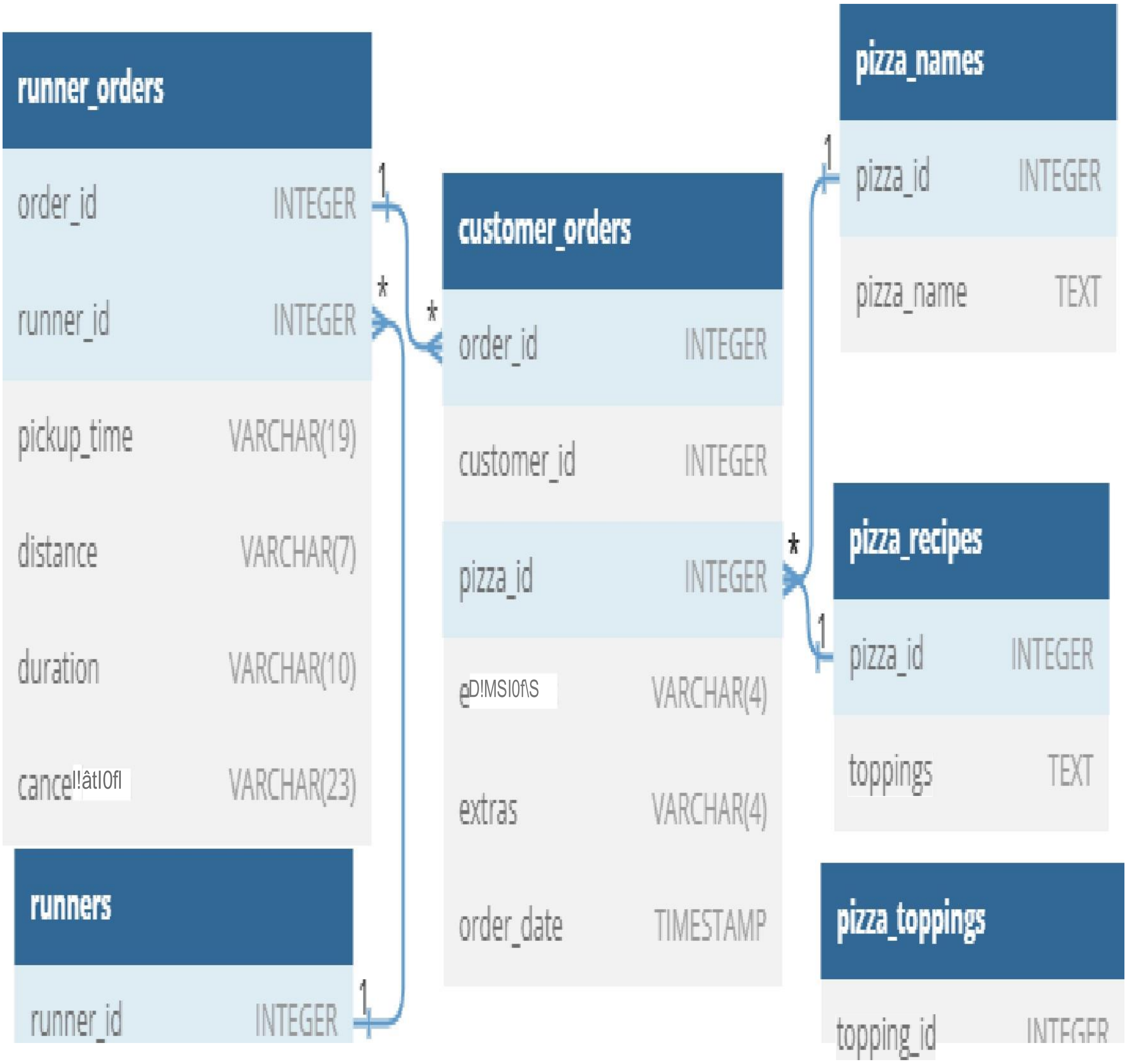
pizza_id	toppings
1	1, 2, 3, 4, 5, 6, 8, 10
2	4, 6, 7, 9, 11, 12

Table 6: pizza_toppings

This table contains all of the **topping_name** values with their corresponding **topping_id** value.

topping_id	topping_name
1	Bacon
2	BBQ Sauce
3	Beef
4	Cheese
5	Chicken
6	Mushrooms
7	Onions
8	Pepperoni
9	Peppers
10	Salami
11	Tomatoes
12	Tomato Sauce

Entity Relationship Diagram



runner_orders

order_id	INTEGER
runner_id	INTEGER
pickup_time	VARCHAR(19)
distance	VARCHAR(7)
duration	VARCHAR(10)
cancel!!âtI0fl	VARCHAR(23)

runners

runner_id	INTEGER

customer_orders

order_id	INTEGER
customer_id	INTEGER
pizza_id	INTEGER
eD!MSI0f\S	VARCHAR(4)
extras	VARCHAR(4)
order_date	TIMESTAMP

pizza_names

pizza_id	INTEGER
pizza_name	TEXT

pizza_recipes

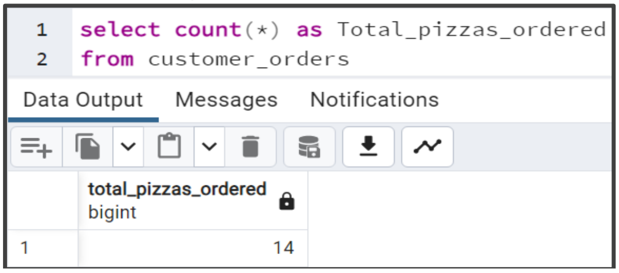
pizza_id	INTEGER
toppings	TEXT

pizza_toppings

topping_id	INTEGER

Case Study Questions

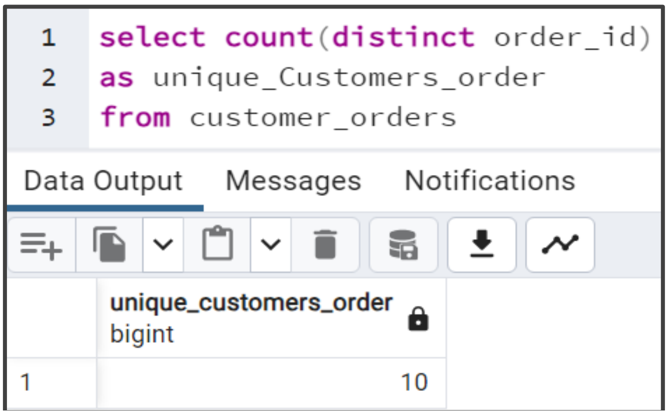
1. **How many pizzas were ordered?**



```
select count(*) as Total_pizzas_ordered
from customer_orders
```

	total_pizzas_ordered bigint
1	14

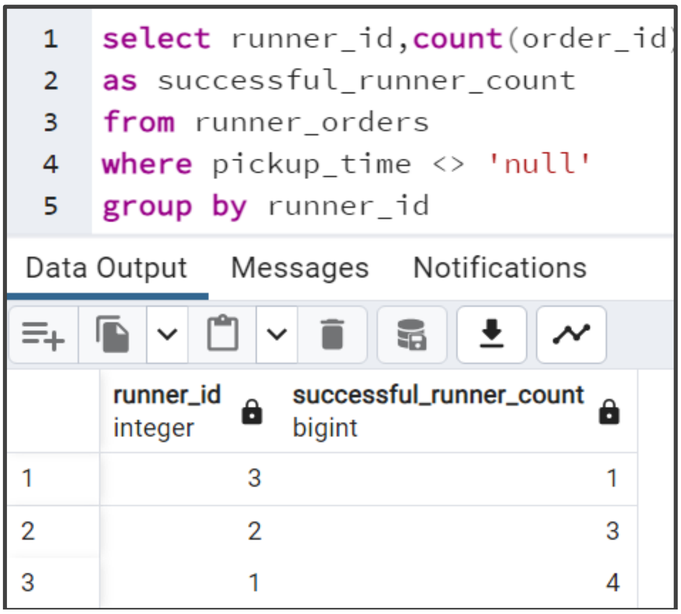
2. **How many unique customer orders were made?**



```
select count(distinct order_id)
as unique_Customers_order
from customer_orders
```

	unique_customers_order bigint
1	10

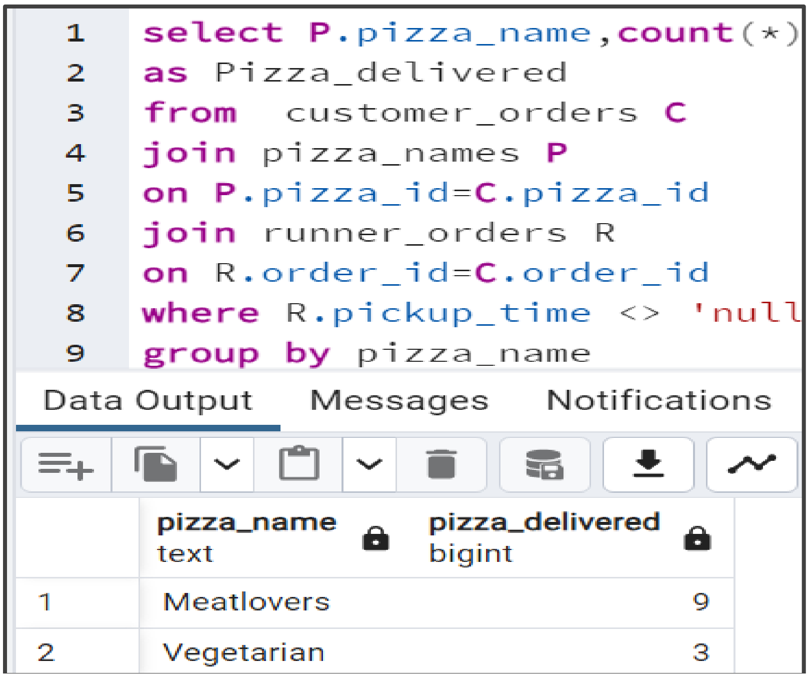
3. **How many successful orders were delivered by each runner?**



```
select runner_id,count(order_id)
as successful_runner_count
from runner_orders
where pickup_time <> 'null'
group by runner_id
```

	runner_id integer	successful_runner_count bigint
1	3	1
2	2	3
3	1	4

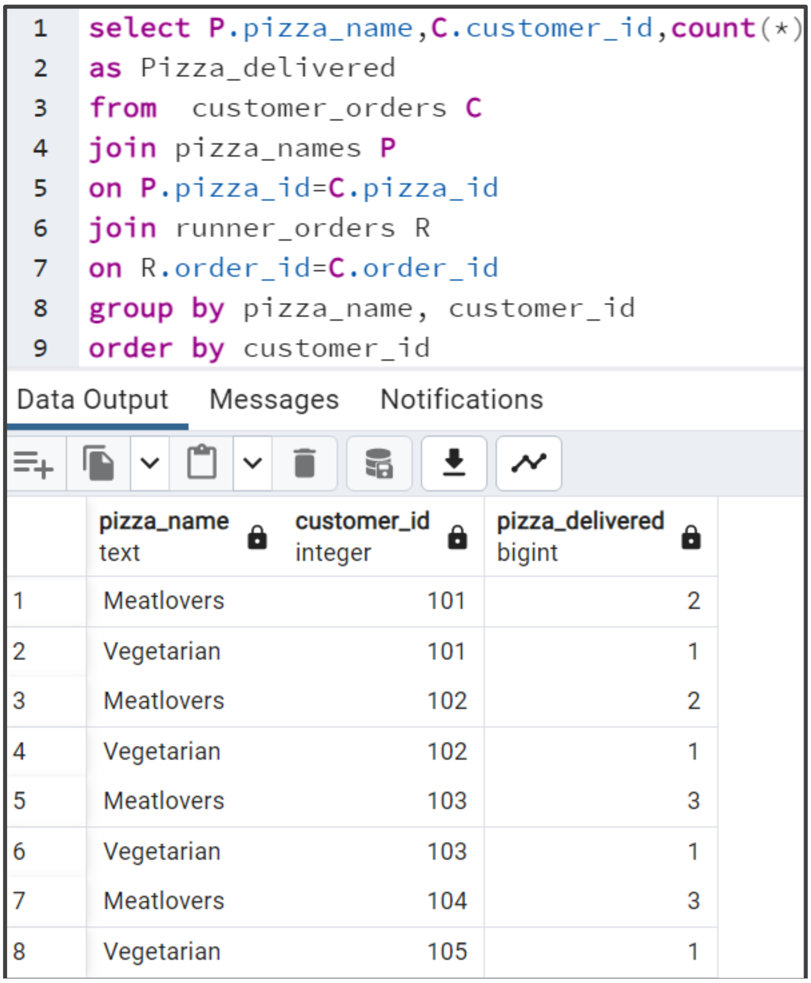
4. **How many of each type of pizza was delivered?**



```sql
select P.pizza_name,count(*)
as Pizza_delivered
from  customer_orders C
join pizza_names P
on P.pizza_id=C.pizza_id
join runner_orders R
on R.order_id=C.order_id
where R.pickup_time <> 'null
group by pizza_name
```

	pizza_name text	pizza_delivered bigint
1	Meatlovers	9
2	Vegetarian	3

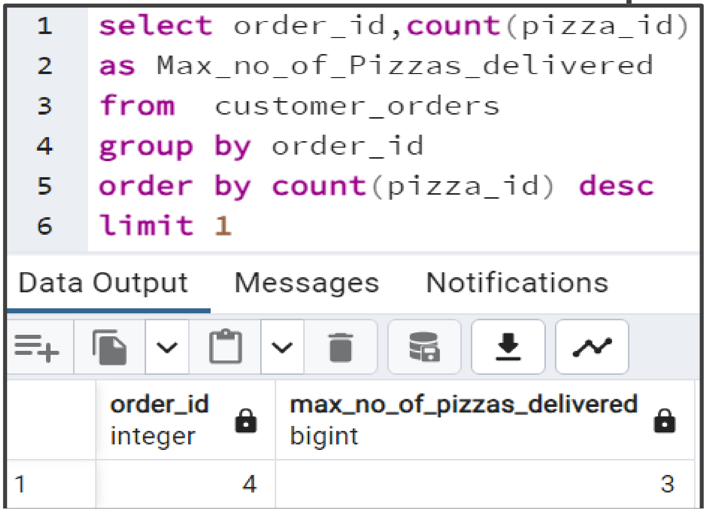
5. **How many Vegetarian and Meatlovers were ordered by each customer?**



```sql
select P.pizza_name,C.customer_id,count(*)
as Pizza_delivered
from  customer_orders C
join pizza_names P
on P.pizza_id=C.pizza_id
join runner_orders R
on R.order_id=C.order_id
group by pizza_name, customer_id
order by customer_id
```

	pizza_name text	customer_id integer	pizza_delivered bigint
1	Meatlovers	101	2
2	Vegetarian	101	1
3	Meatlovers	102	2
4	Vegetarian	102	1
5	Meatlovers	103	3
6	Vegetarian	103	1
7	Meatlovers	104	3
8	Vegetarian	105	1

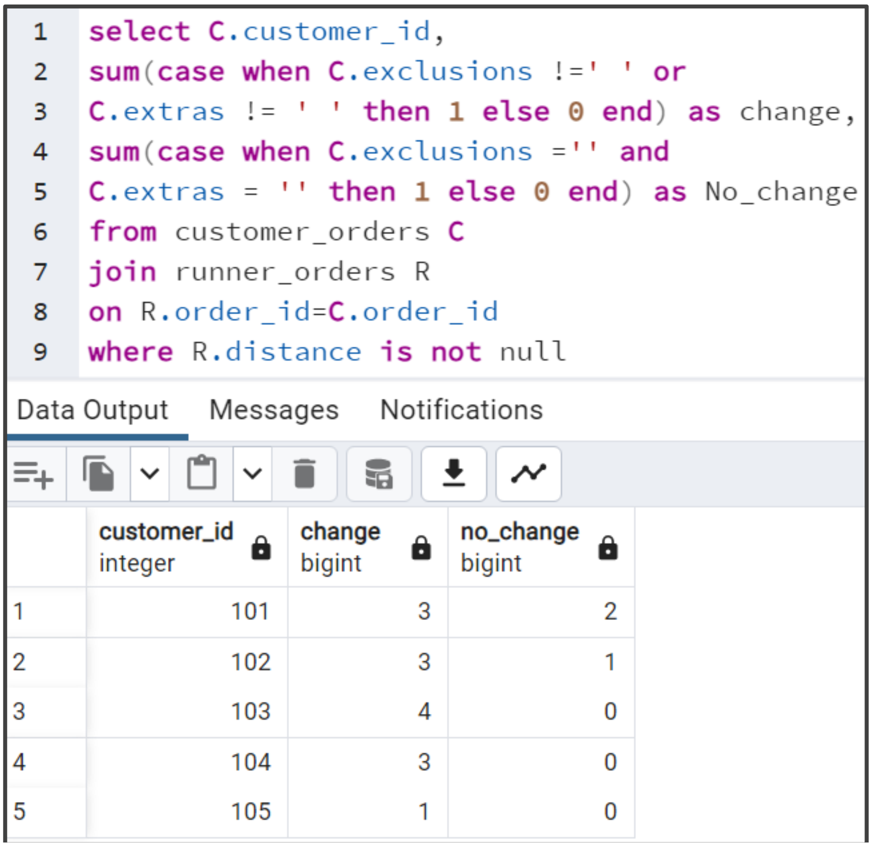
6. **What was the maximum number of pizzas delivered in a single order?**



```sql
select order_id,count(pizza_id)
as Max_no_of_Pizzas_delivered
from  customer_orders
group by order_id
order by count(pizza_id) desc
limit 1
```

	order_id integer	max_no_of_pizzas_delivered bigint
1	4	3

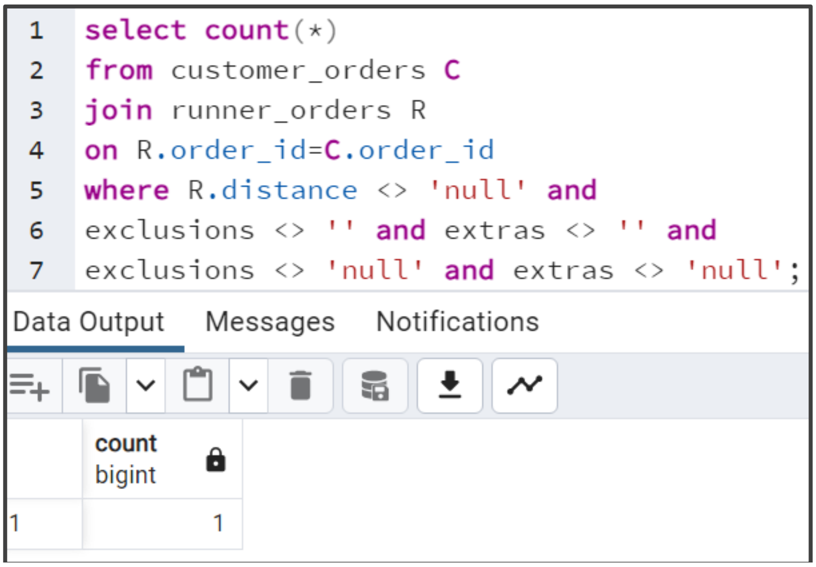
7. **For each customer, how many delivered pizzas had at least 1 change and how many had no changes?**



```sql
select C.customer_id,
sum(case when C.exclusions !=' ' or
C.extras != ' ' then 1 else 0 end) as change,
sum(case when C.exclusions ='' and
C.extras = '' then 1 else 0 end) as No_change
from customer_orders C
join runner_orders R
on R.order_id=C.order_id
where R.distance is not null
```

	customer_id integer	change bigint	no_change bigint
1	101	3	2
2	102	3	1
3	103	4	0
4	104	3	0
5	105	1	0

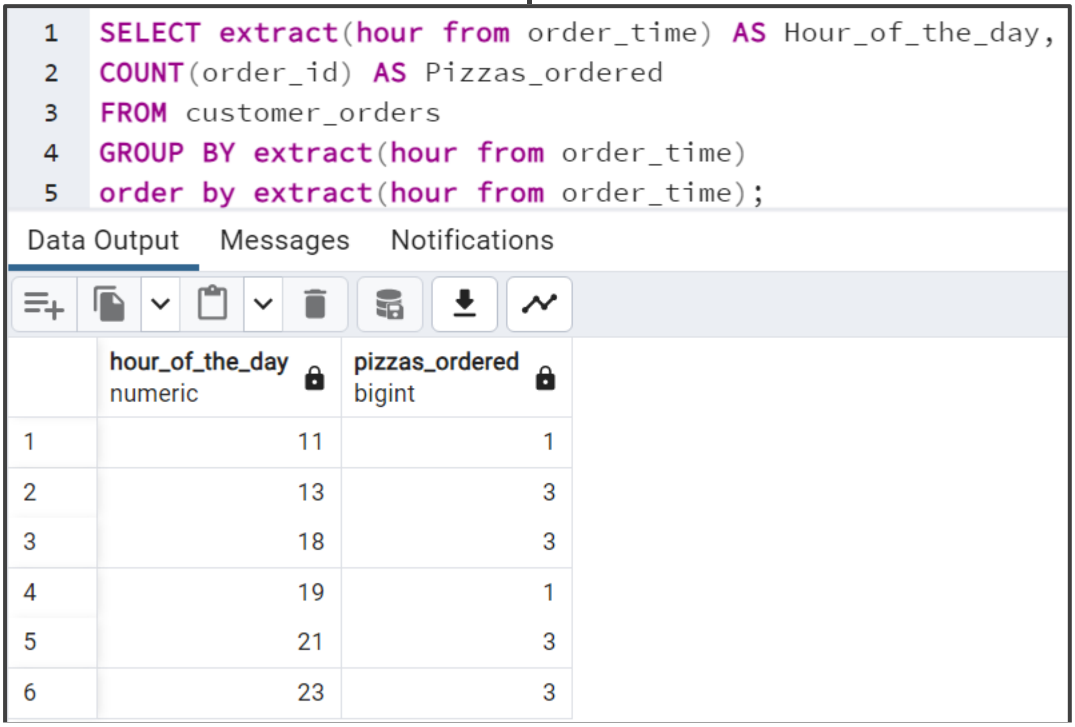
8. **How many pizzas were delivered that had both exclusions and extras?**



```sql
select count(*)
from customer_orders C
join runner_orders R
on R.order_id=C.order_id
where R.distance <> 'null' and
exclusions <> '' and extras <> '' and
exclusions <> 'null' and extras <> 'null';
```

	count bigint
1	1

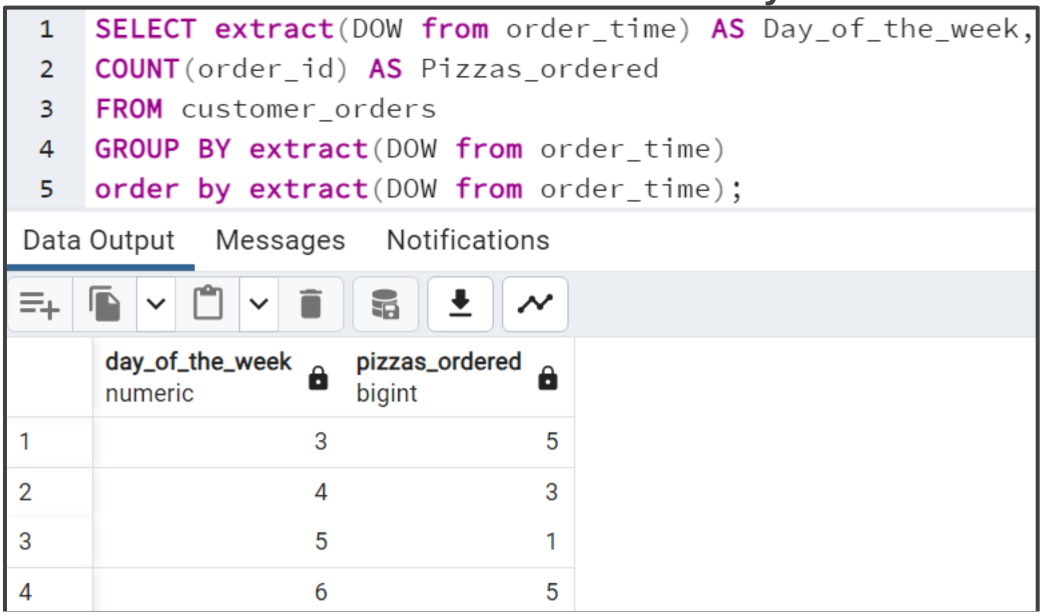
9. **What was the total volume of pizzas ordered for each hour of the day?**



```sql
SELECT extract(hour from order_time) AS Hour_of_the_day,
COUNT(order_id) AS Pizzas_ordered
FROM customer_orders
GROUP BY extract(hour from order_time)
order by extract(hour from order_time);
```

	hour_of_the_day numeric	pizzas_ordered bigint
1	11	1
2	13	3
3	18	3
4	19	1
5	21	3
6	23	3

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



```sql
SELECT extract(DOW from order_time) AS Day_of_the_week,
COUNT(order_id) AS Pizzas_ordered
FROM customer_orders
GROUP BY extract(DOW from order_time)
order by extract(DOW from order_time);
```

	day_of_the_week numeric	pizzas_ordered bigint
1	3	5
2	4	3
3	5	1
4	6	5

Insights

- There are total 14 pizzas ordered.
- There are 10 unique customer orders made.
- Runner 1 has delivered highest number of pizza whereas runner 3 has delivered the least number of pizzas.
- Meatlovers pizza was delivered 9 times and vegetarian pizza was delivered 3 times.
- Maximum number of pizzas delivered in a single order is three.
- Only one pizza was delivered that had both extras and exclusions.
- Highest number of pizza ordered at 13 (1:00 pm), 18 (6:00 pm) and 21 (9:00 pm).
- Pizza runner 2 takes a longer time whereas pizza runner 3 takes the shortest time to arrive at pizza HQ to pickup the order.
- Pizza runner 1 has the highest successful delivery percentage.

THANK YOU