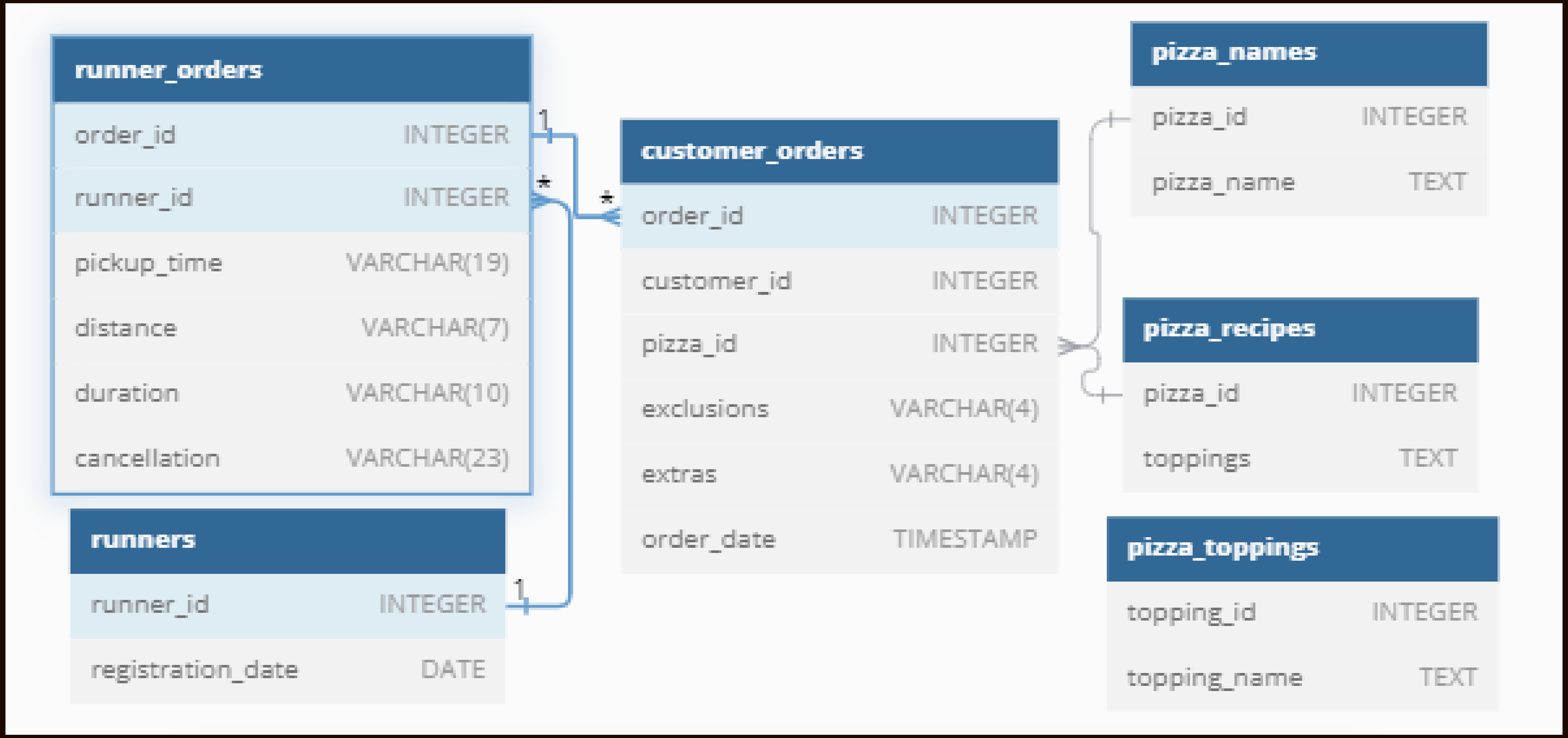
PIZZA RUNNER

PIZZA RUNNER

INTRODUCTION

Did you know that over 115 million kilograms of pizza is consumed daily worldwide???

PIZZA RUNNER



runner_orders

order_id	INTEGER
runner_id	INTEGER
pickup_time	VARCHAR(19)
distance	VARCHAR(7)
duration	VARCHAR(10)
cancellation	VARCHAR(23)

runners

runner_id	INTEGER
registration_date	DATE

customer_orders

order_id	INTEGER
customer_id	INTEGER
pizza_id	INTEGER
exclusions	VARCHAR(4)
extras	VARCHAR(4)
order_date	TIMESTAMP

pizza_names

pizza_id	INTEGER
pizza_name	TEXT

pizza_recipes

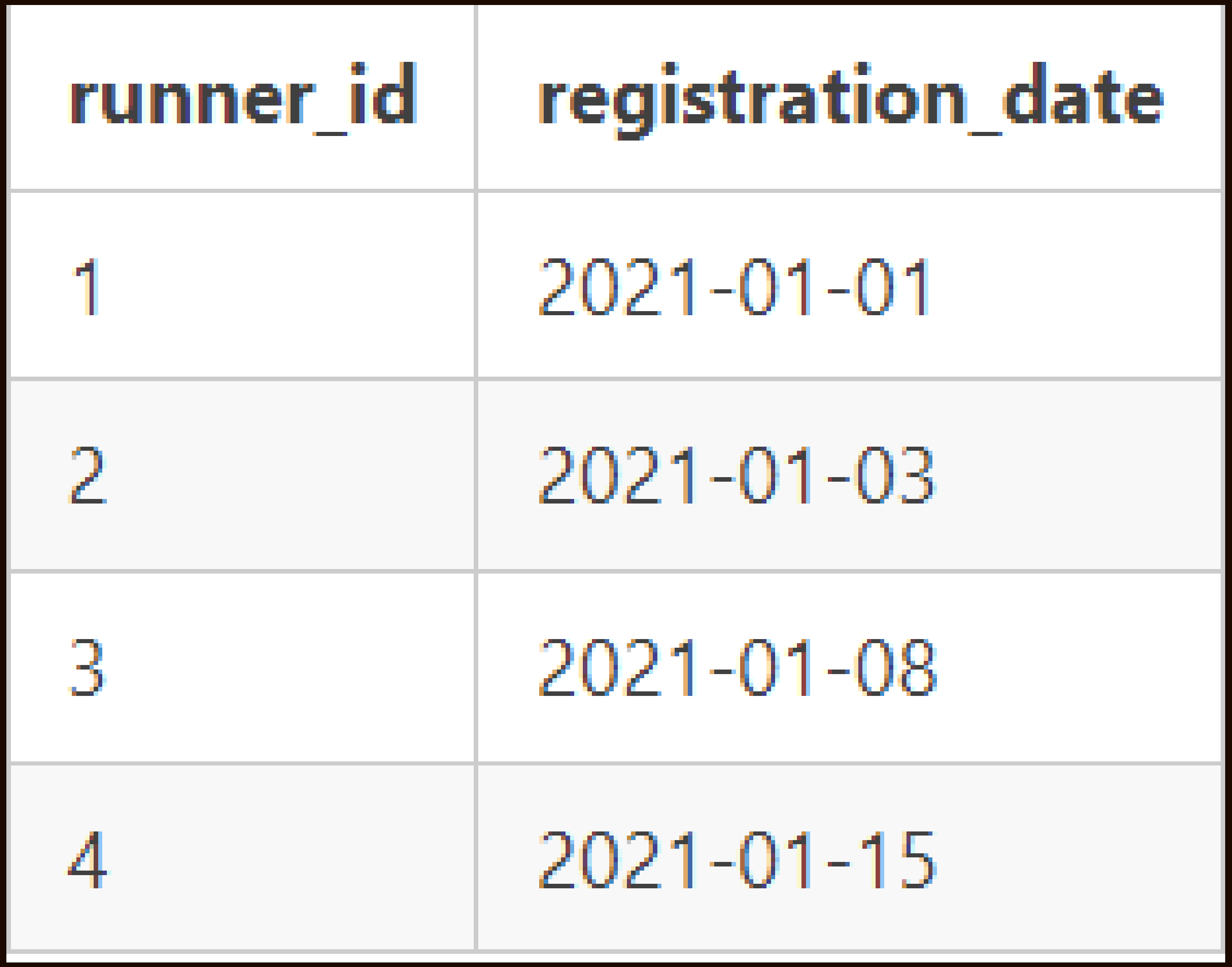
pizza_id	INTEGER
toppings	TEXT

pizza_toppings

topping_id	INTEGER
topping_name	TEXT

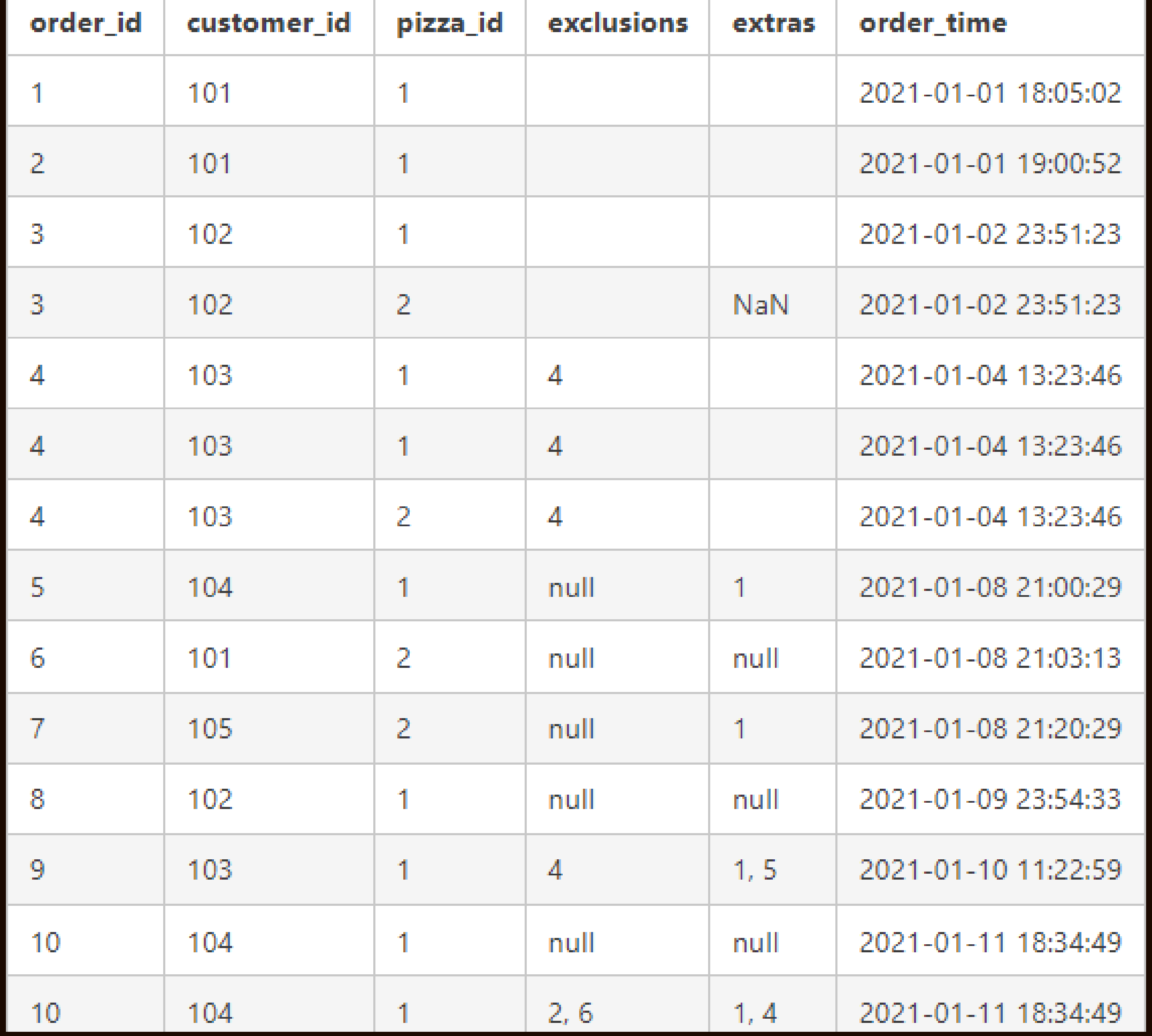
ENTITY RELATIONSHIP DIAGRAM

TABLE-1: RUNNERS



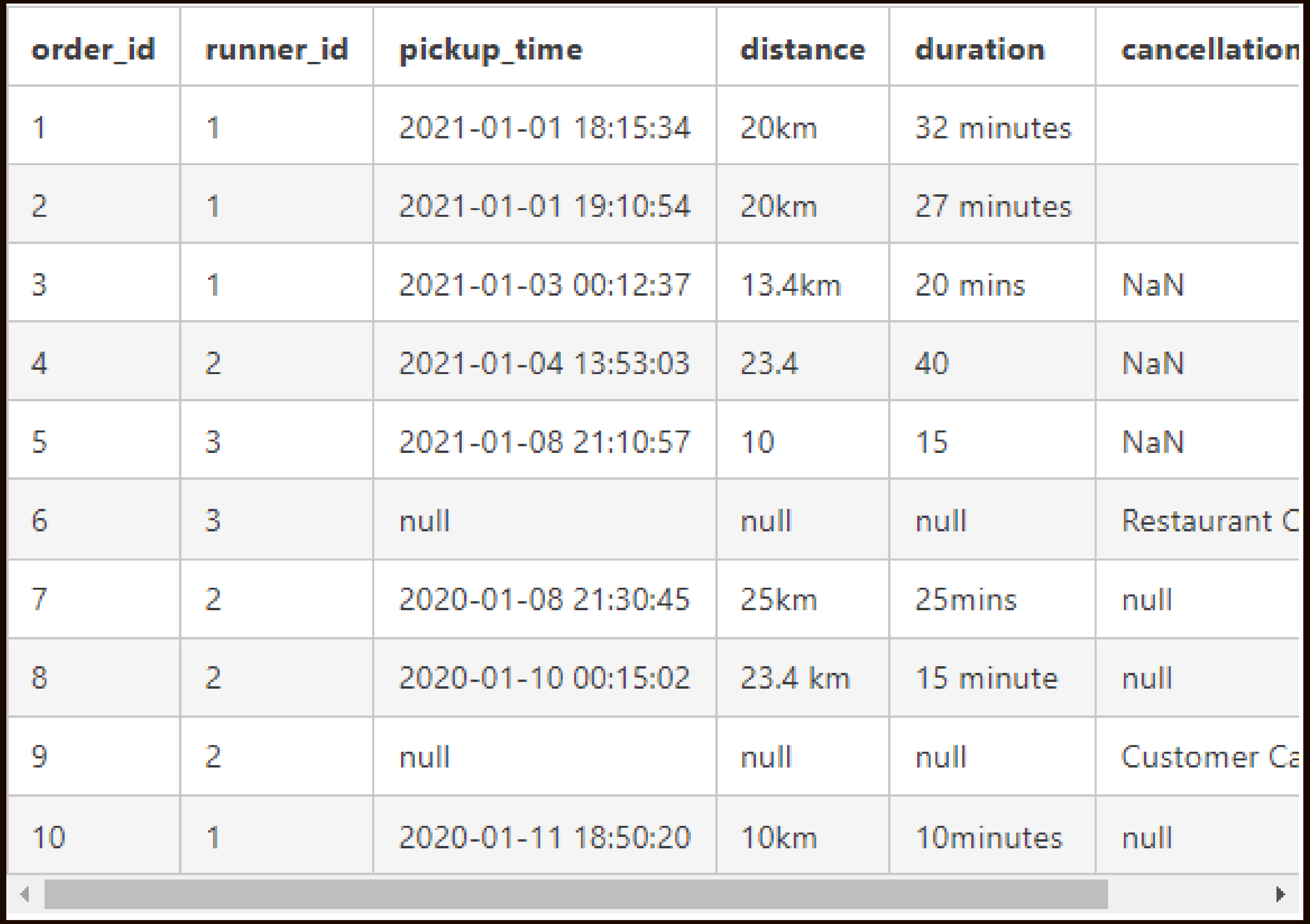
runner_id	registration_date
1	2021-01-01
2	2021-01-03
3	2021-01-08
4	2021-01-15

TABLE-2: CUSTOMER_ORDERS



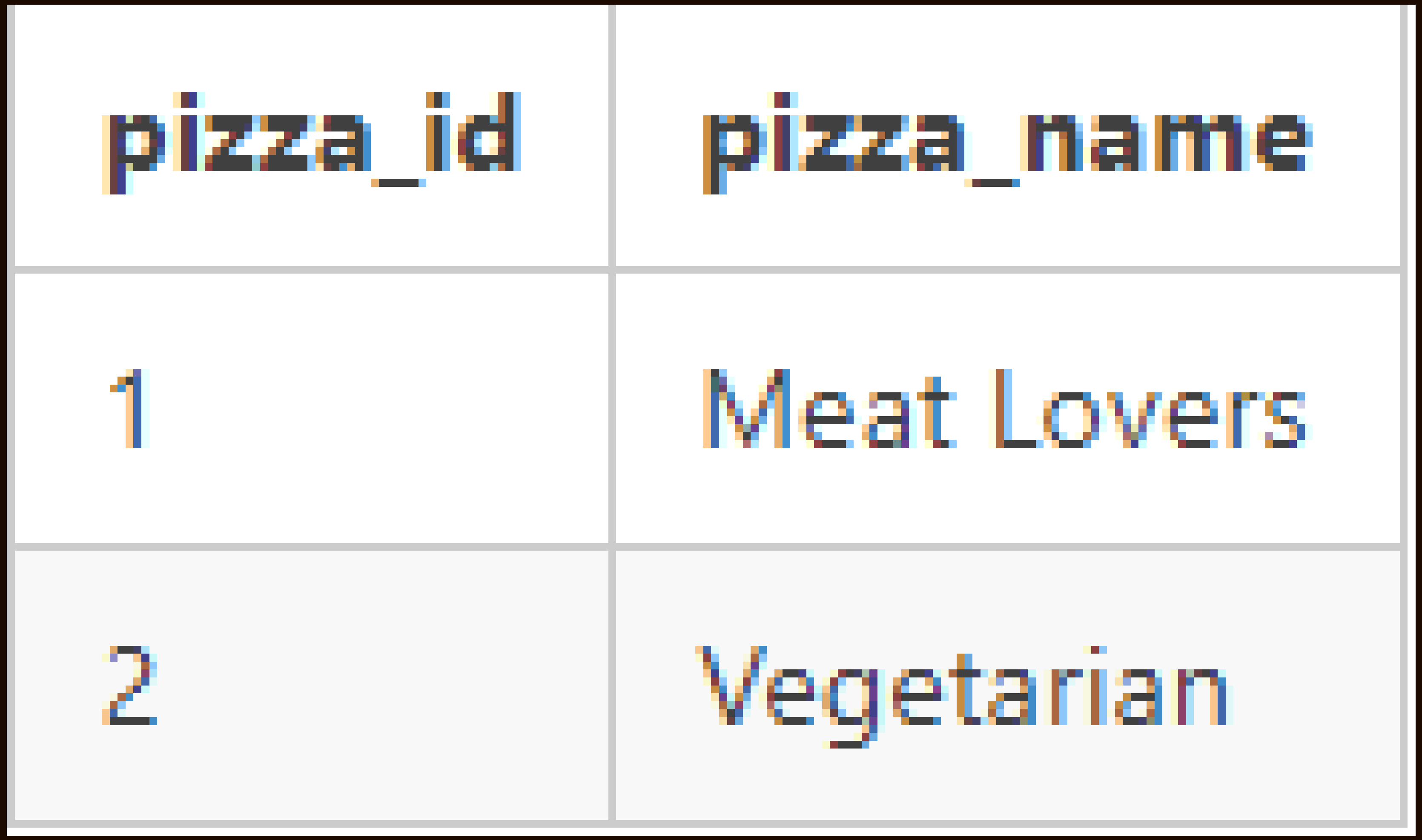
order_id	customer_id	pizza_id	exclusions	extras	order_time
1	101	1			2021-01-01 18:05:02
2	101	1			2021-01-01 19:00:52
3	102	1			2021-01-02 23:51:23
3	102	2		NaN	2021-01-02 23:51:23
4	103	1	4		2021-01-04 13:23:46
4	103	1	4		2021-01-04 13:23:46
4	103	2	4		2021-01-04 13:23:46
5	104	1	null	1	2021-01-08 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



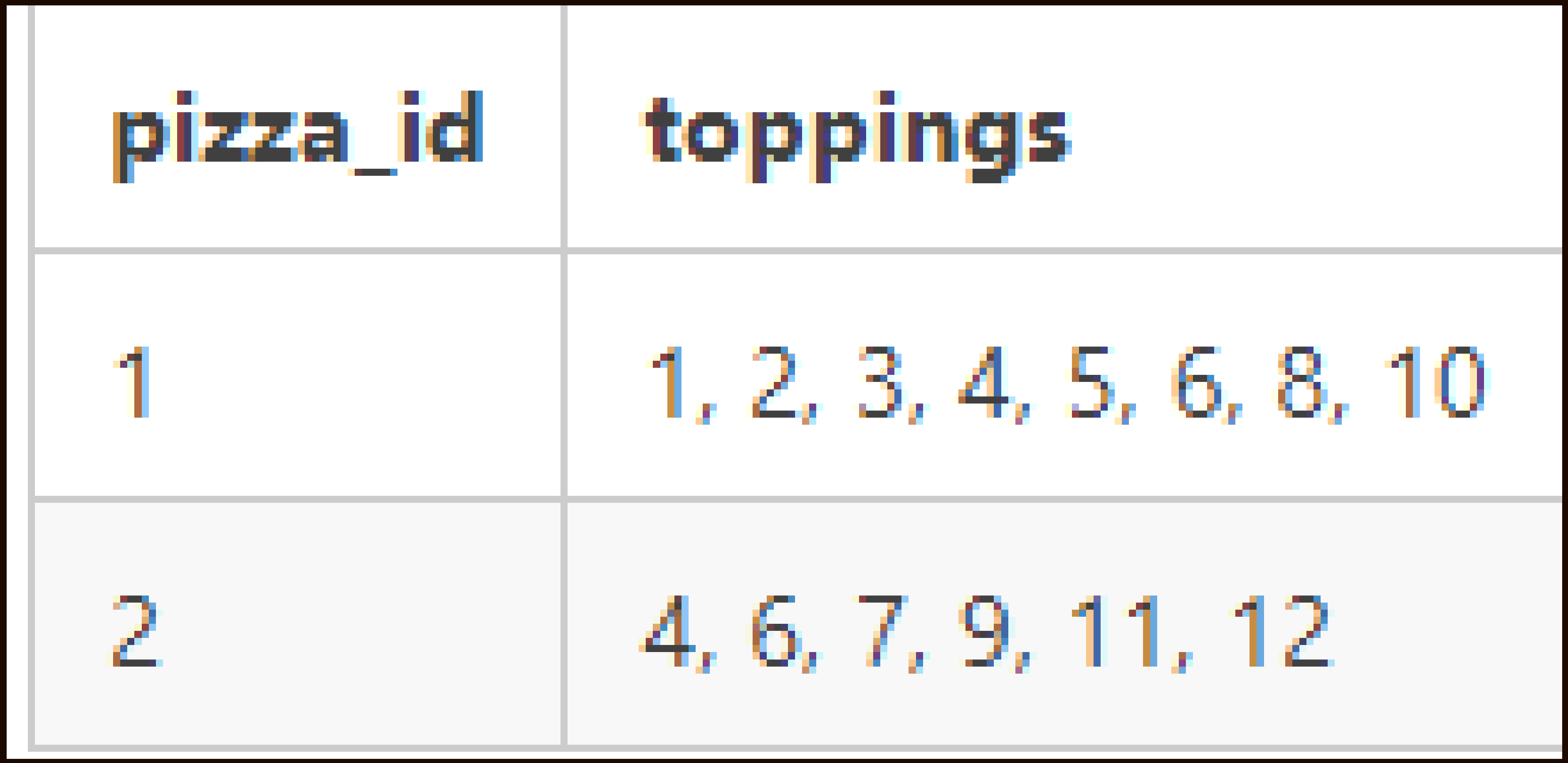
order_id	runner_id	pickup_time	distance	duration	cancellation
1	1	2021-01-01 18:15:34	20km	32 minutes	
2	1	2021-01-01 19:10:54	20km	27 minutes	
3	1	2021-01-03 00:12:37	13.4km	20 mins	NaN
4	2	2021-01-04 13:53:03	23.4	40	NaN
5	3	2021-01-08 21:10:57	10	15	NaN
6	3	null	null	null	Restaurant C
7	2	2020-01-08 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 Ca
10	1	2020-01-11 18:50:20	10km	10minutes	null

TABLE-4: PIZZA_NAMES



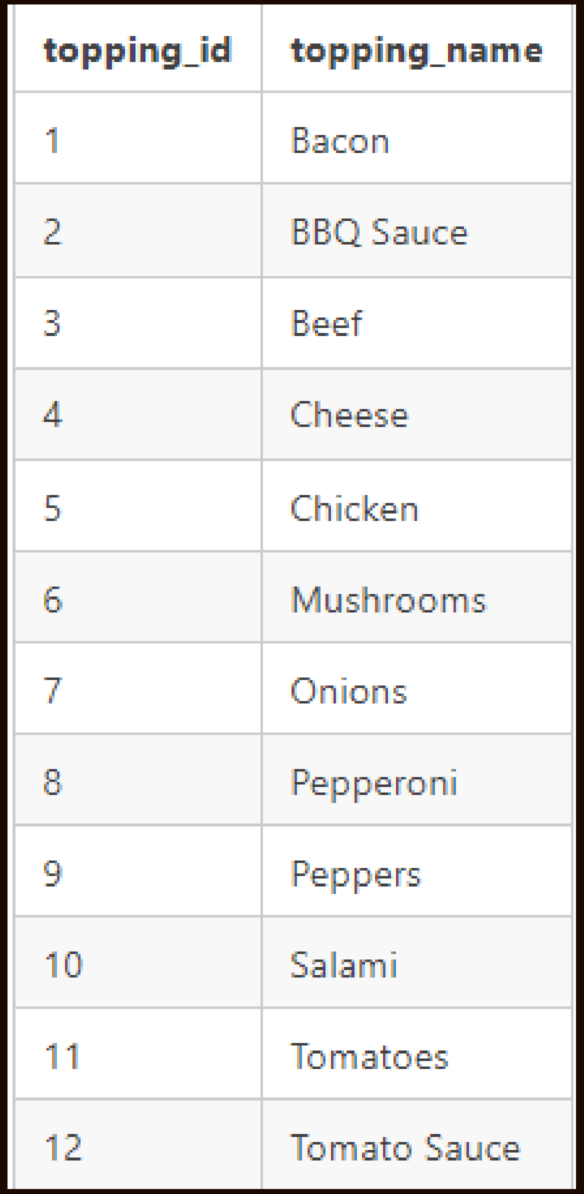
pizza_id	pizza_name
1	Meat Lovers
2	Vegetarian

TABLE-5: PIZZA_RECIPES



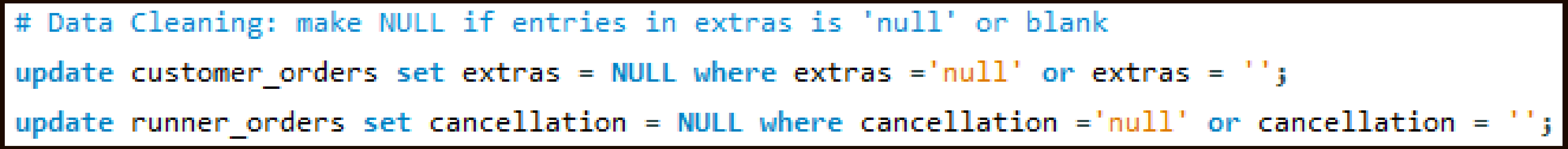
pizza_id	toppings
1	1, 2, 3, 4, 5, 6, 8, 10
2	4, 6, 7, 9, 11, 12

TABLE-6: PIZZA_TOPINGS

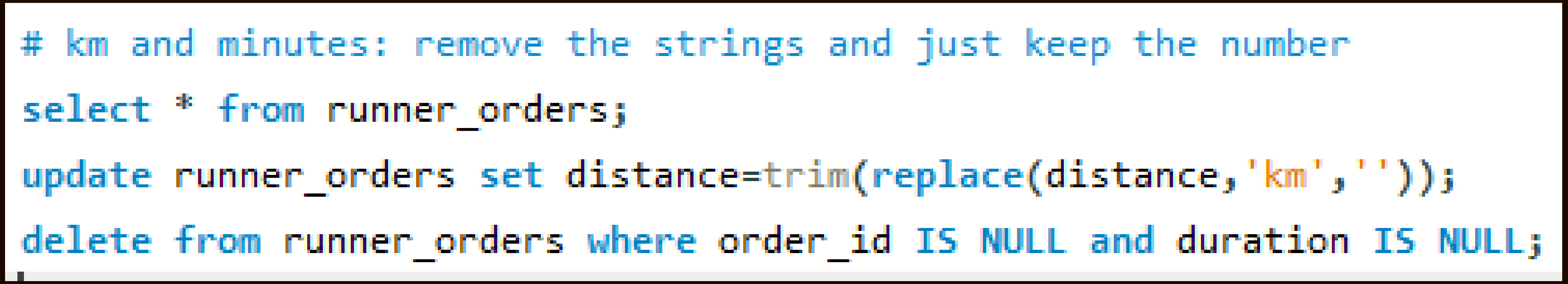


topping_id	topping_name
1	Bacon
2	BBQ Sauce
3	Beef
4	Cheese
5	Chicken
6	Mushrooms
7	Onions
8	Pepperoni
9	Peppers
10	Salami
11	Tomatoes
12	Tomato Sauce

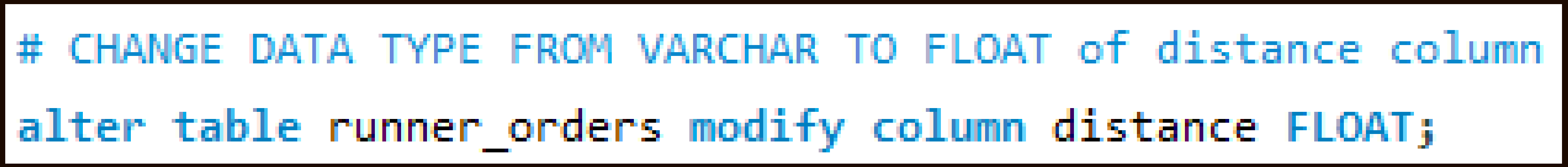
DATA CLEANING



```
# Data Cleaning: make NULL if entries in extras is 'null' or blank
update customer_orders set extras = NULL where extras ='null' or extras = '';
update runner_orders set cancellation = NULL where cancellation ='null' or cancellation = '';
```

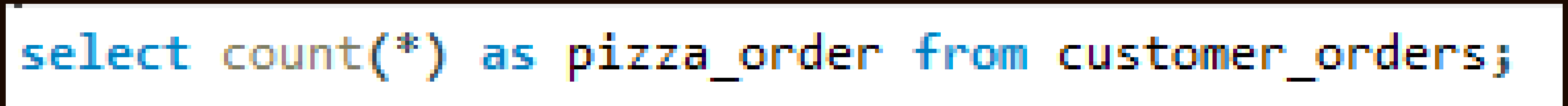


```
# km and minutes: remove the strings and just keep the number
select * from runner_orders;
update runner_orders set distance=trim(replace(distance,'km',''));
delete from runner_orders where order_id IS NULL and duration IS NULL;
```

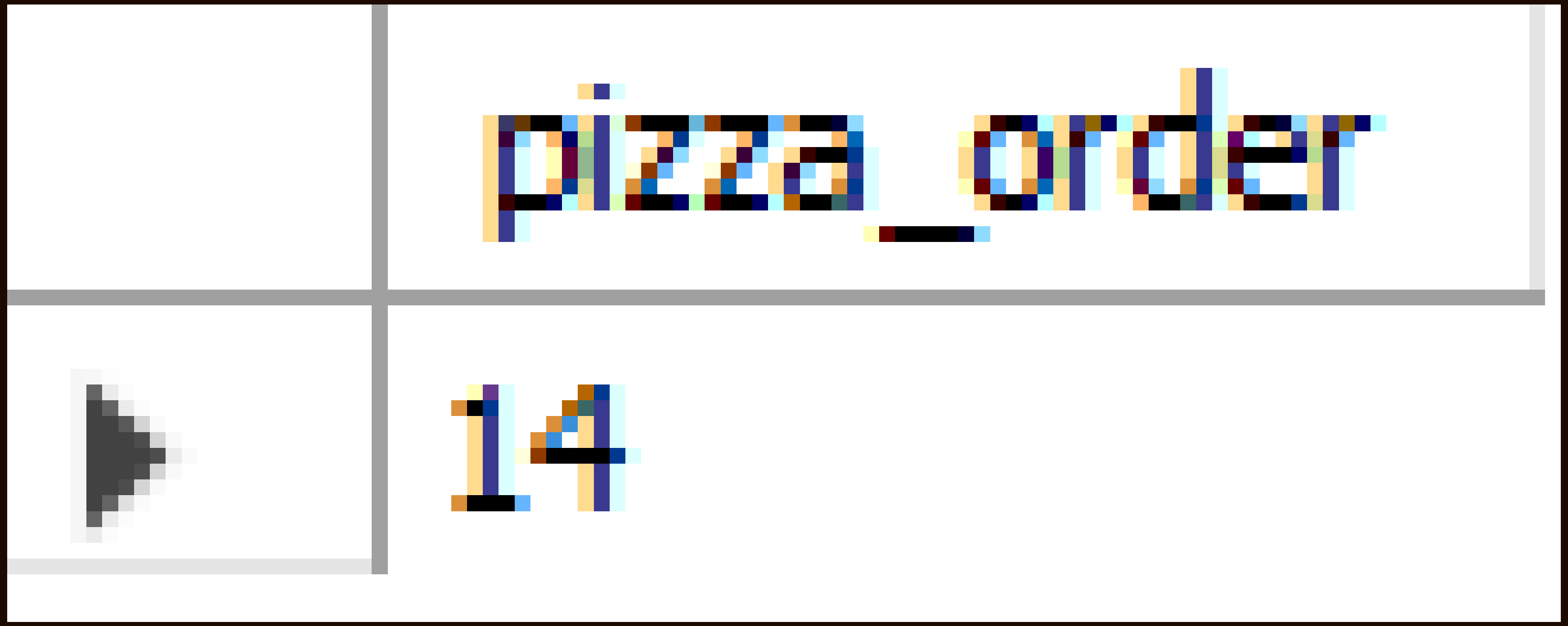


```
# CHANGE DATA TYPE FROM VARCHAR TO FLOAT of distance column
alter table runner_orders modify column distance FLOAT;
```

Q1. HOW MANY PIZZAS WERE ORDERED?

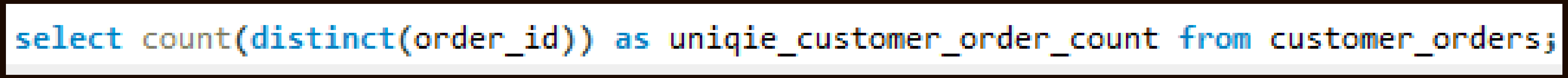


```sql
select count(*) as pizza_order from customer_orders;
```

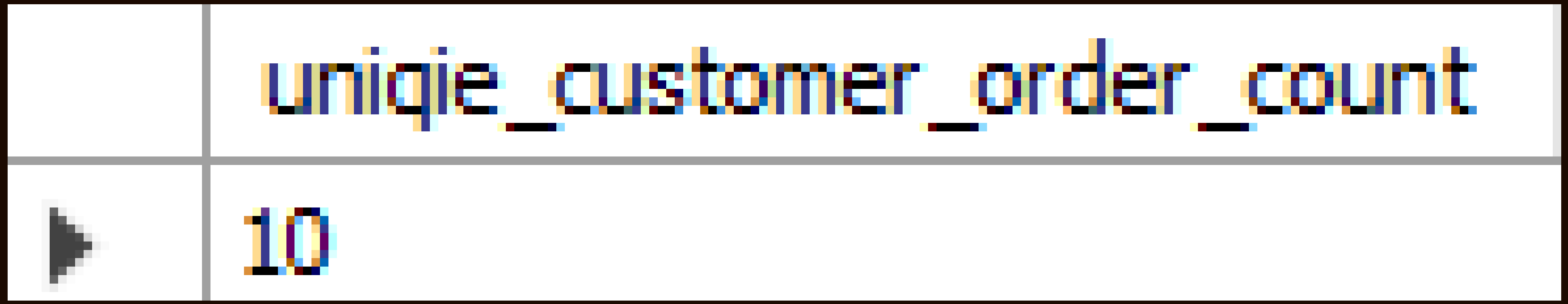


	pizza_order
▶	14

Q2. HOW MANY UNIQUE CUSTOMER ORDERS WERE MADE?

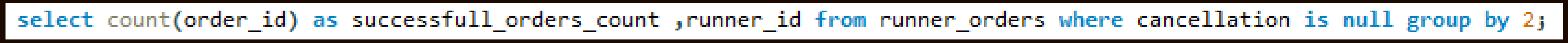


```sql
select count(distinct(order_id)) as uniqie_customer_order_count from customer_orders;
```

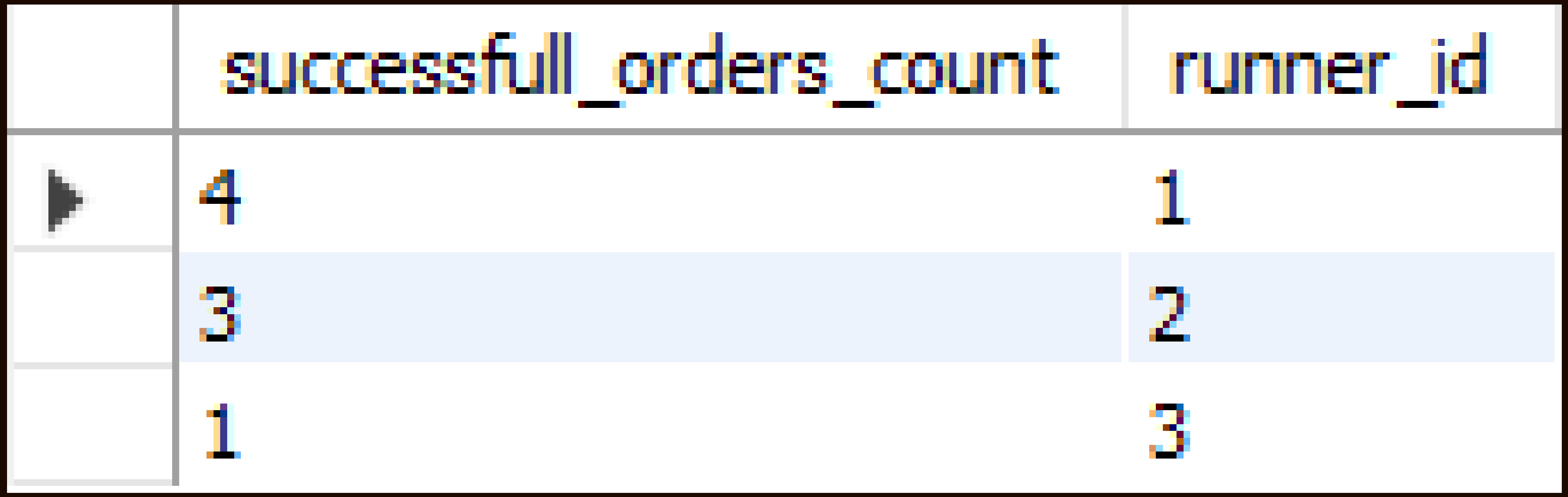


	uniqie_customer_order_count
▶	10

Q3. HOW MANY SUCCESSFULL ORDERS WERE DELIVERED BY EACH RUNNER?

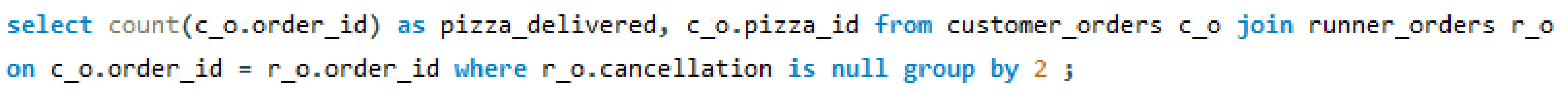


```
select count(order_id) as successfull_orders_count ,runner_id from runner_orders where cancellation is null group by 2;
```

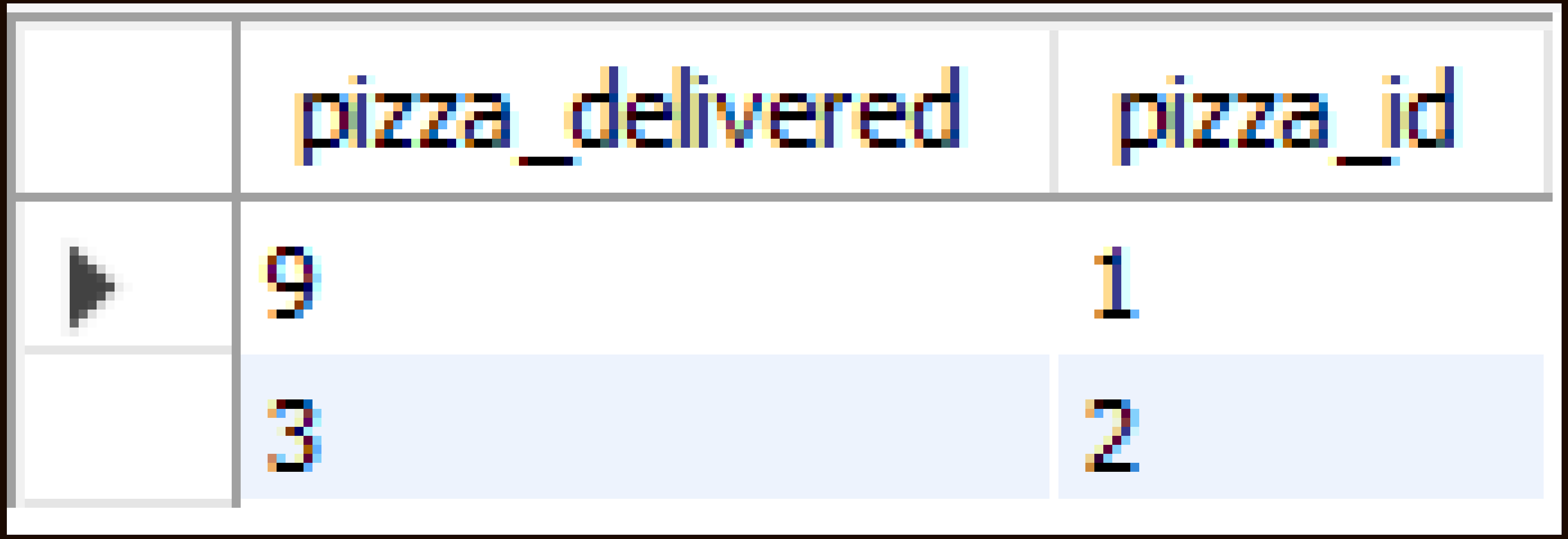


successfull_orders_count	runner_id
4	1
3	2
1	3

Q4. HOW MANY OF EACH TYPE OF PIZZA WAS DELIVERED?

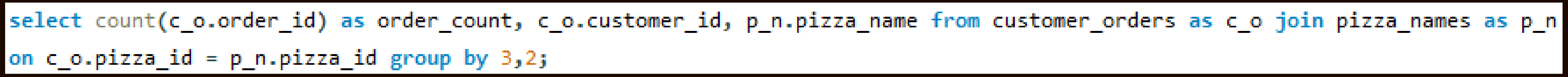


```
select count(c_o.order_id) as pizza_delivered, c_o.pizza_id from customer_orders c_o join runner_orders r_o
on c_o.order_id = r_o.order_id where r_o.cancellation is null group by 2 ;
```

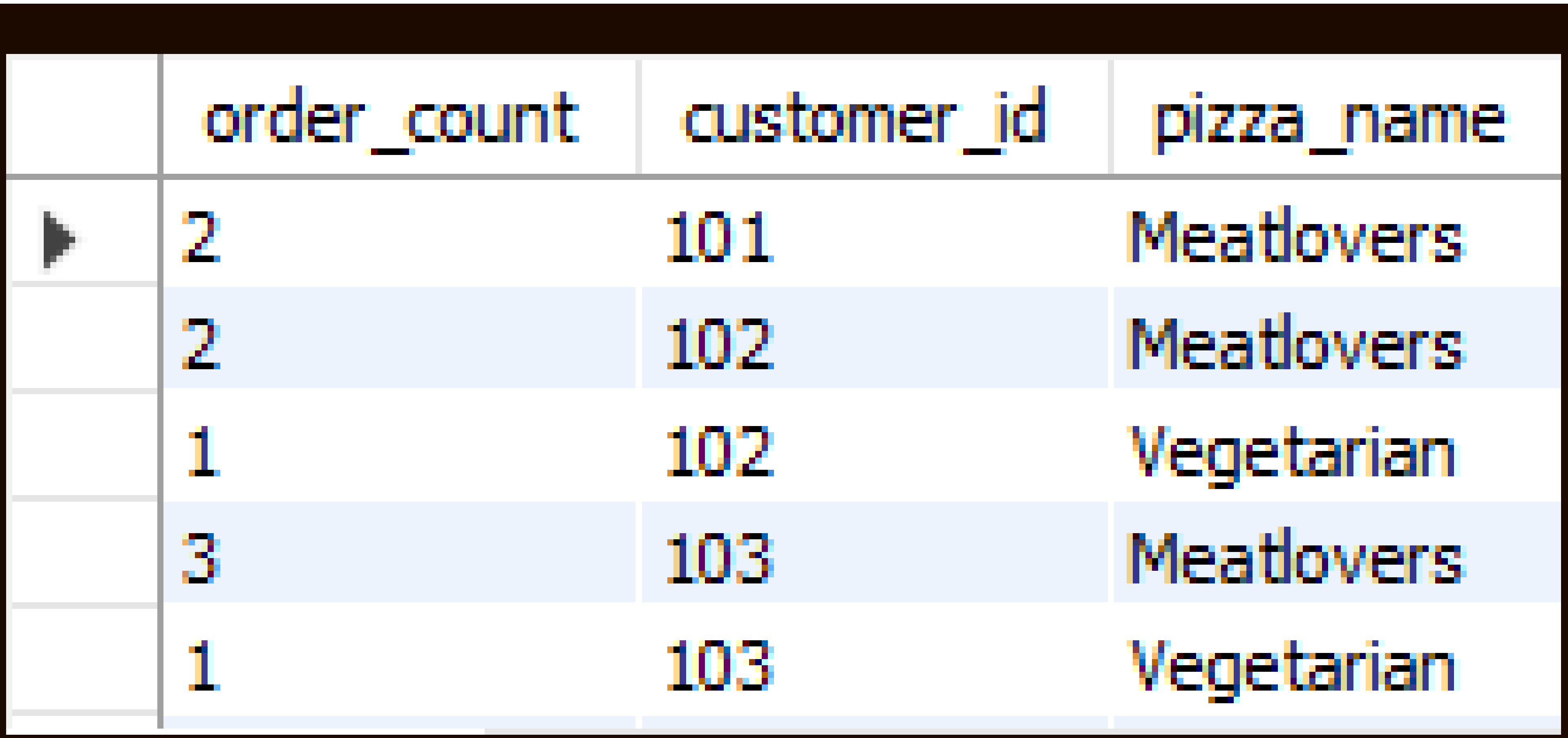


pizza_delivered	pizza_id
9	1
3	2

Q5. HOW MANY VEGETARIAN AND MEATLOVER WERE ORDERED BY EACH CUSTOMER?

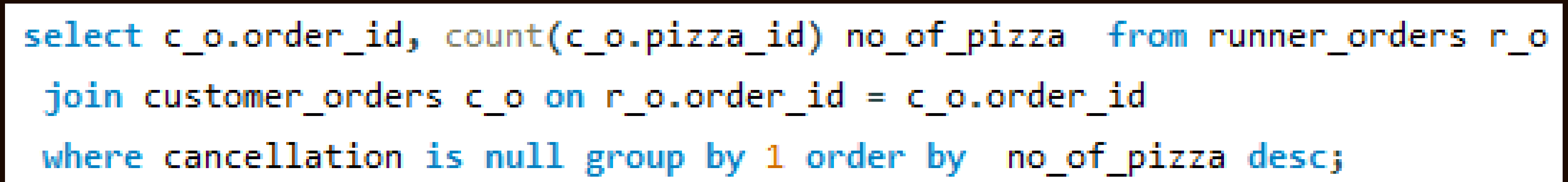


```
select count(c_o.order_id) as order_count, c_o.customer_id, p_n.pizza_name from customer_orders as c_o join pizza_names as p_n
on c_o.pizza_id = p_n.pizza_id group by 3,2;
```

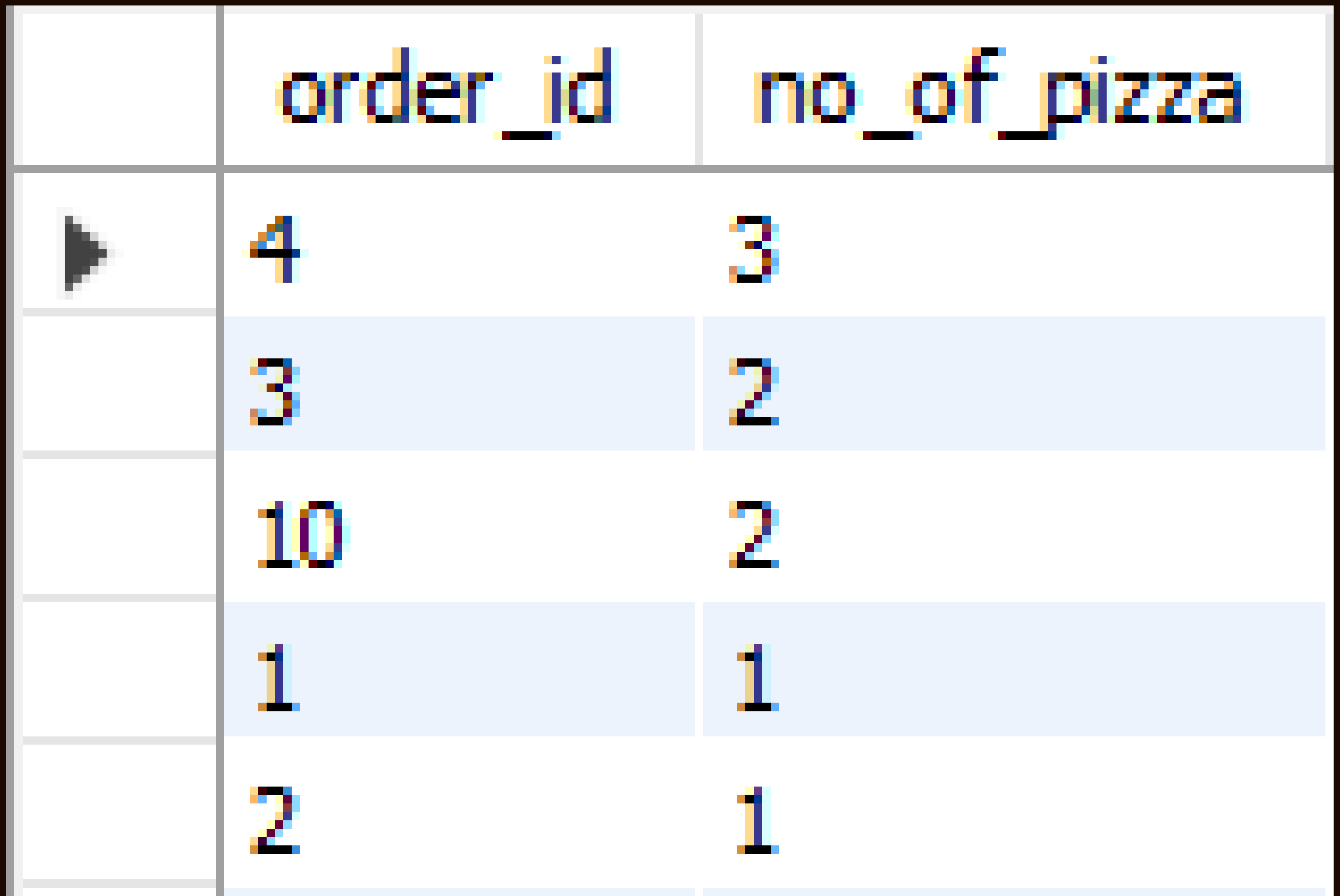


	order_count	customer_id	pizza_name
▶	2	101	Meatlovers
	2	102	Meatlovers
	1	102	Vegetarian
	3	103	Meatlovers
	1	103	Vegetarian

Q6. WHAT WAS THE MAXIMUM NUMBER OF PIZZAS DEIVERED IN A SINGLE ORDER?

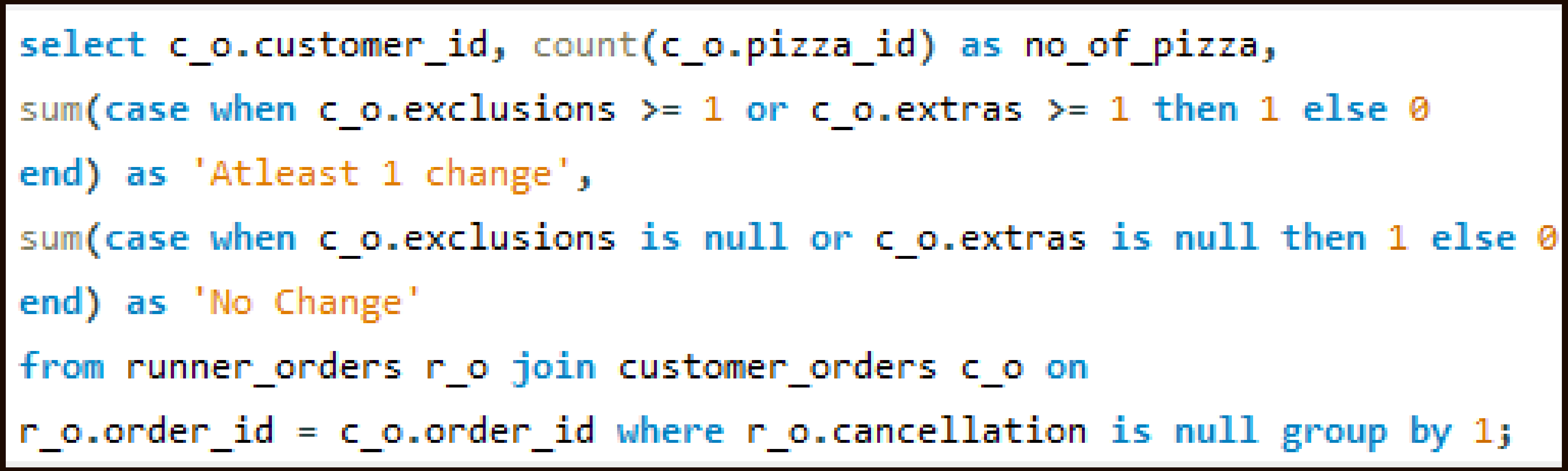


```
select c_o.order_id, count(c_o.pizza_id) no_of_pizza  from runner_orders r_o
 join customer_orders c_o on r_o.order_id = c_o.order_id
 where cancellation is null group by 1 order by  no_of_pizza desc;
```

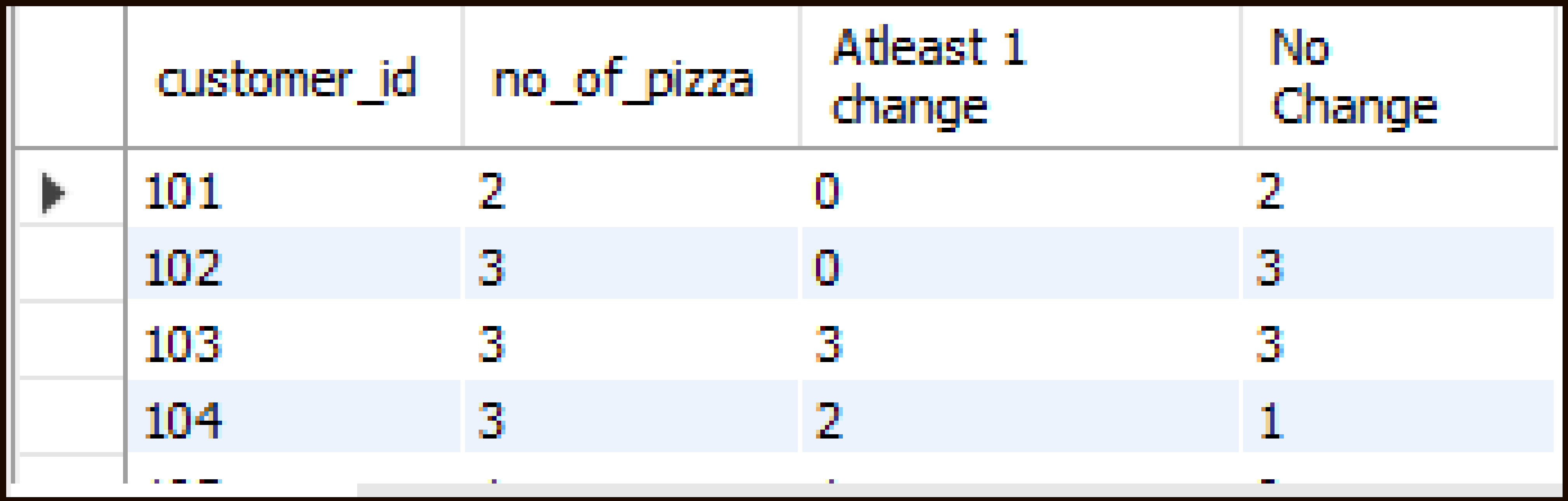


order_id	no_of_pizza
4	3
3	2
10	2
1	1
2	1

Q7. FOR EACH CUSTOMER, HOW MANY DELIVERED PIZZAS HAD ATLEAST 1 CHANGE AND HOW MANY HAD NO CHANGE?

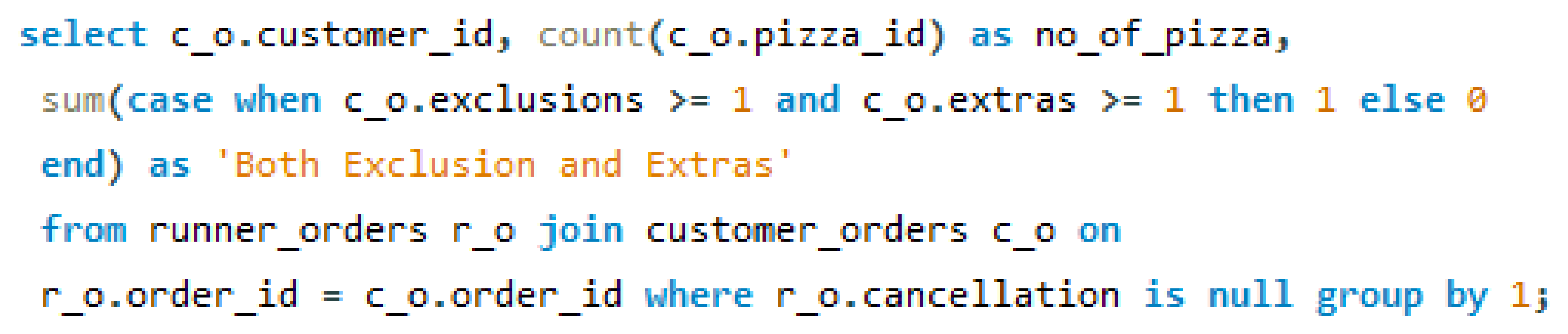


```
select c_o.customer_id, count(c_o.pizza_id) as no_of_pizza,
sum(case when c_o.exclusions >= 1 or c_o.extras >= 1 then 1 else 0
end) as 'Atleast 1 change',
sum(case when c_o.exclusions is null or c_o.extras is null then 1 else 0
end) as 'No Change'
from runner_orders r_o join customer_orders c_o on
r_o.order_id = c_o.order_id where r_o.cancellation is null group by 1;
```

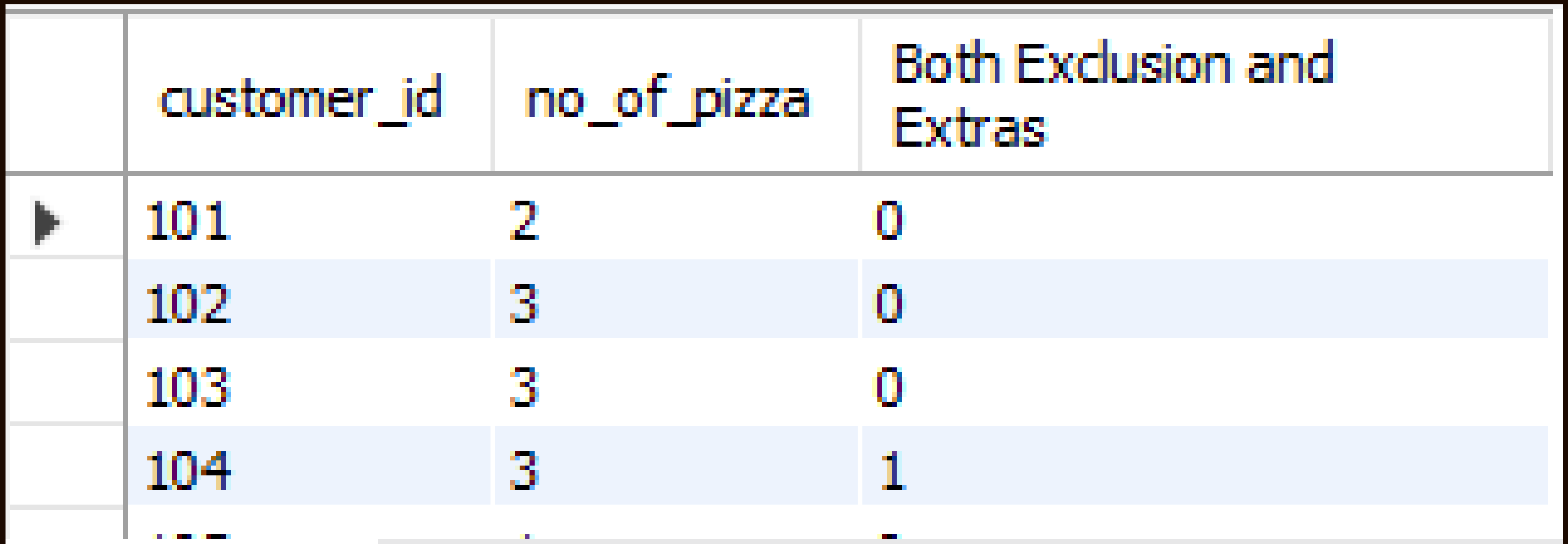


customer_id	no_of_pizza	Atleast 1 change	No Change
101	2	0	2
102	3	0	3
103	3	3	3
104	3	2	1

Q8. HOW MANY PIZZAS WERE DELIVERED THAT HAD BOTH EXCLUSIONS AND EXTRAS?

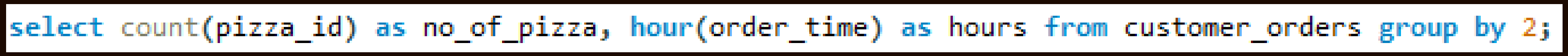


```
select c_o.customer_id, count(c_o.pizza_id) as no_of_pizza,
 sum(case when c_o.exclusions >= 1 and c_o.extras >= 1 then 1 else 0
 end) as 'Both Exclusion and Extras'
 from runner_orders r_o join customer_orders c_o on
 r_o.order_id = c_o.order_id where r_o.cancellation is null group by 1;
```

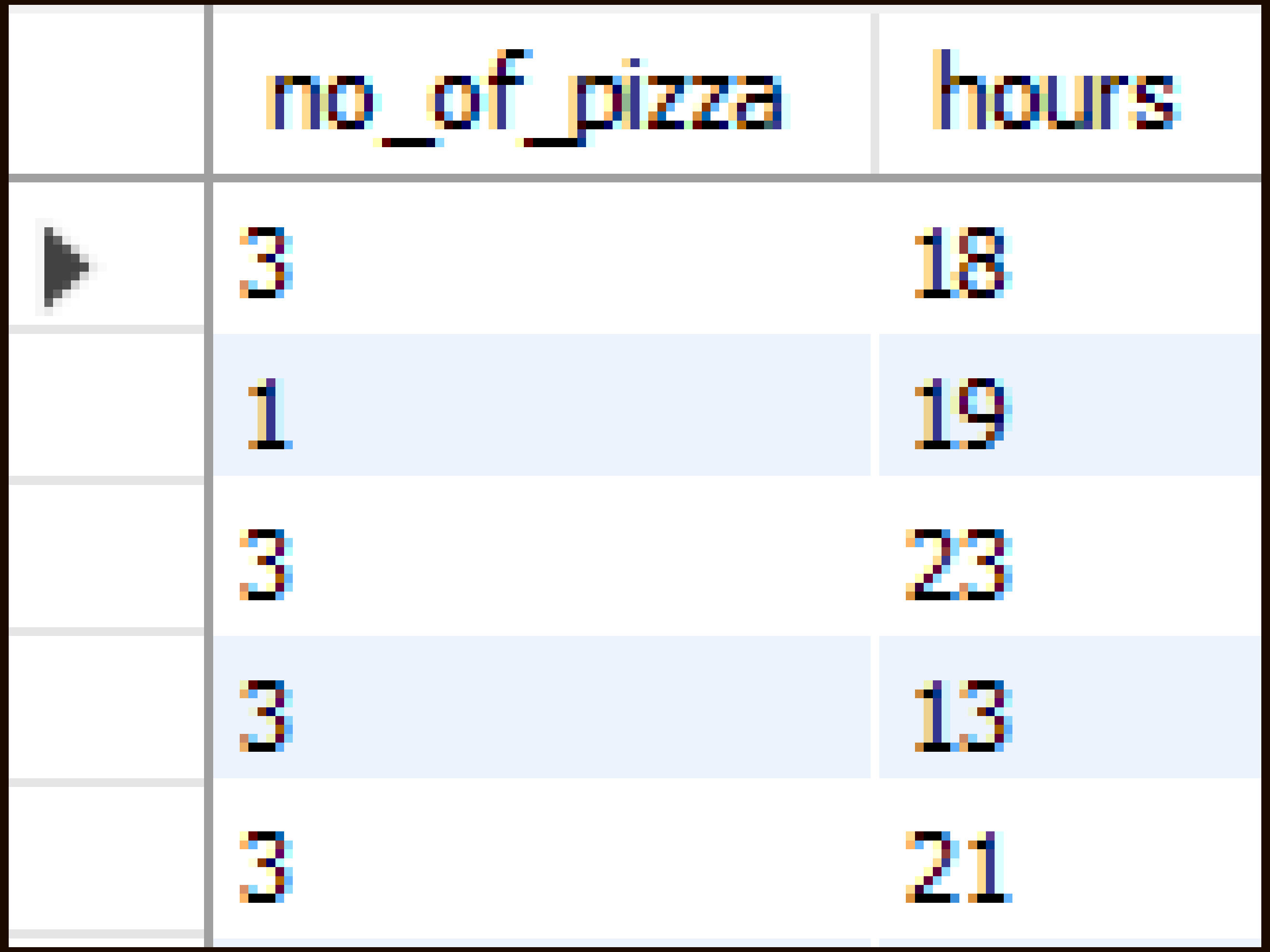


	customer_id	no_of_pizza	Both Exclusion and Extras
▶	101	2	0
	102	3	0
	103	3	0
	104	3	1

Q9. WHAT WAS THE TOTAL VOLUME OF PIZZAS ORDERED FOR EACH HOUR OF DAY?

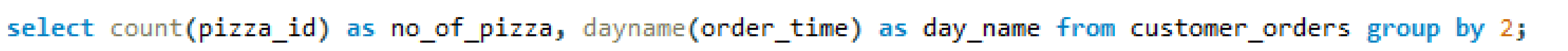


```sql
select count(pizza_id) as no_of_pizza, hour(order_time) as hours from customer_orders group by 2;
```

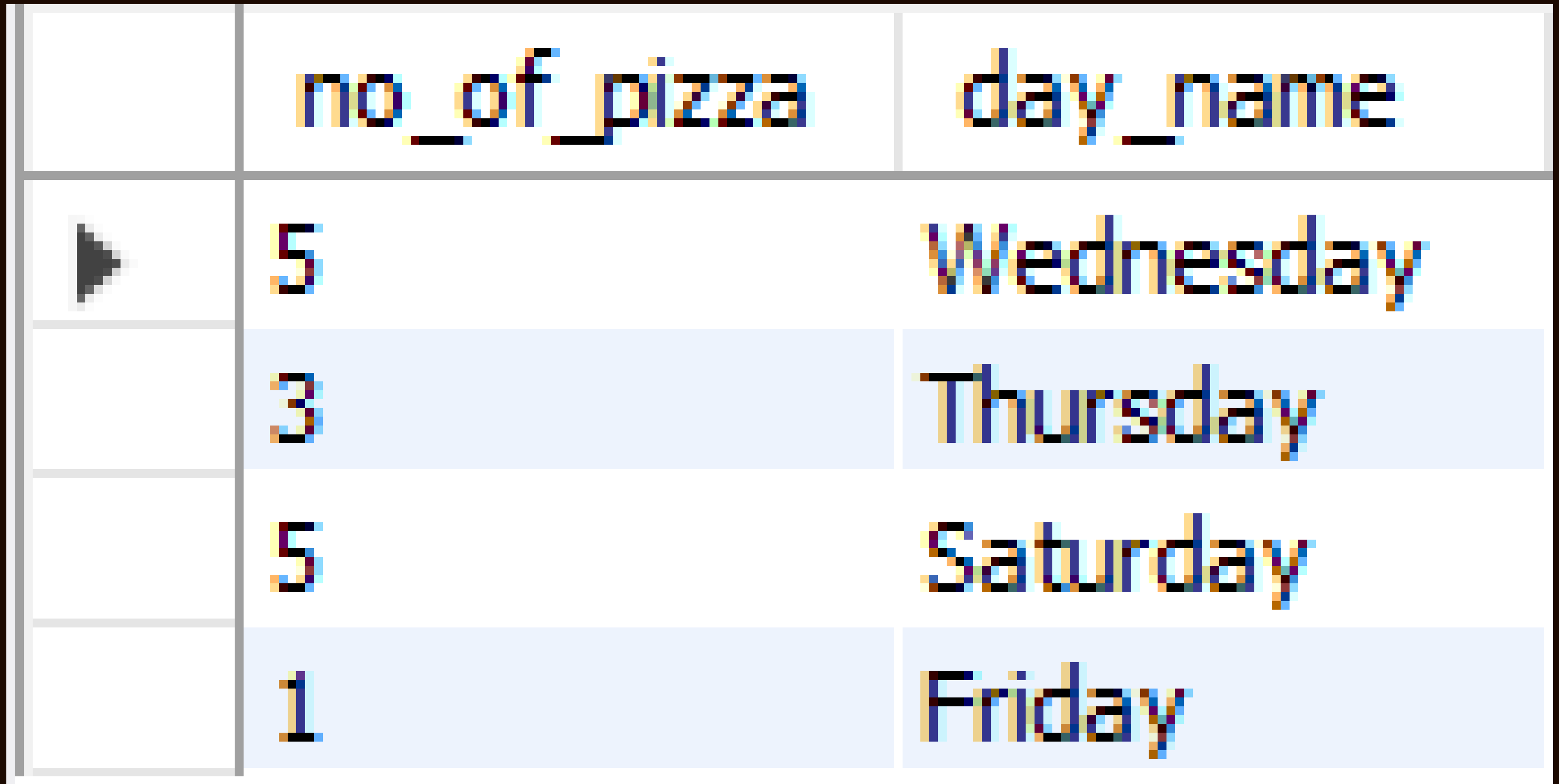


	no_of_pizza	hours
▶	3	18
	1	19
	3	23
	3	13
	3	21

Q10. WHAT WAS THE VOLUME OF ORDERS FOR EACH DAY OF THE WEEK?

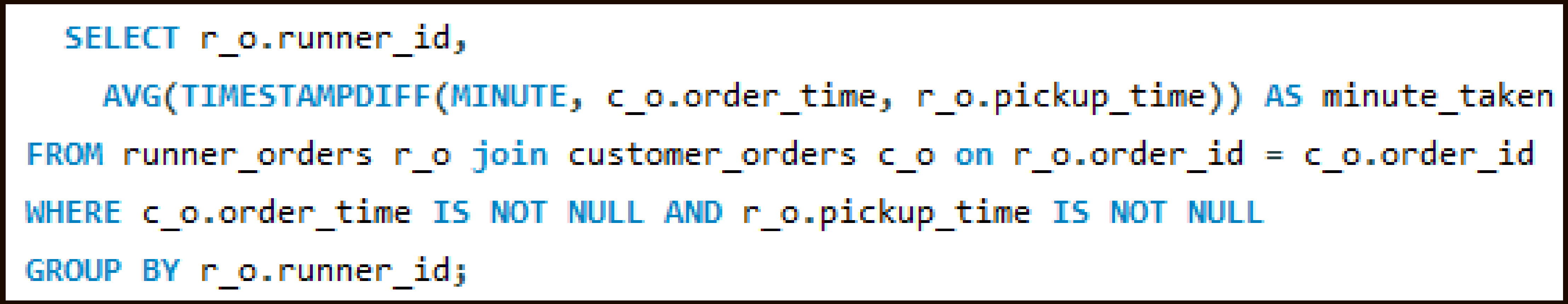


```sql
select count(pizza_id) as no_of_pizza, dayname(order_time) as day_name from customer_orders group by 2;
```

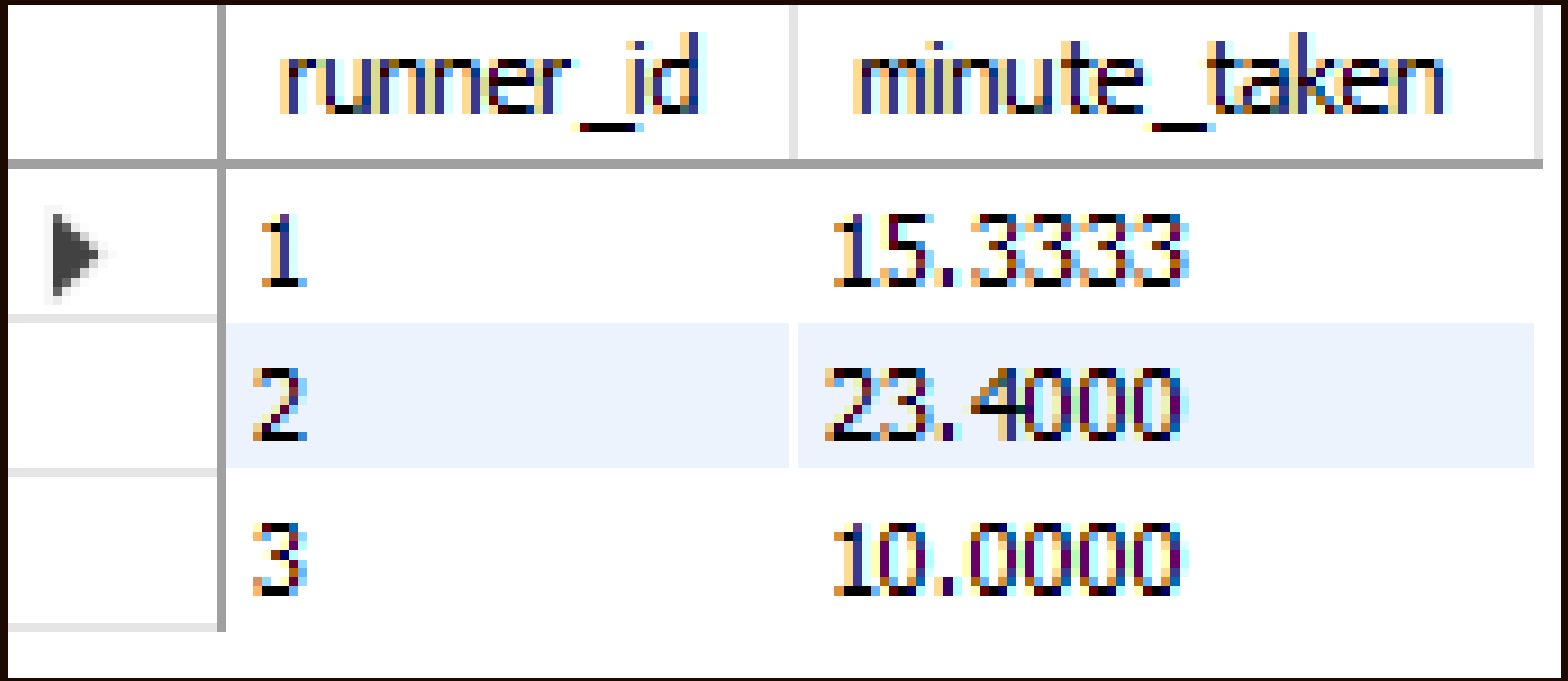


no_of_pizza	day_name
5	Wednesday
3	Thursday
5	Saturday
1	Friday

Q11. WHAT WAS THE AVERAGE TIME IN MINUTES IT TOOK FOR EACH RUNNER TO ARRIVE AT THE PIZZA RUNNER HQ TO PICKUP THE ORDER?

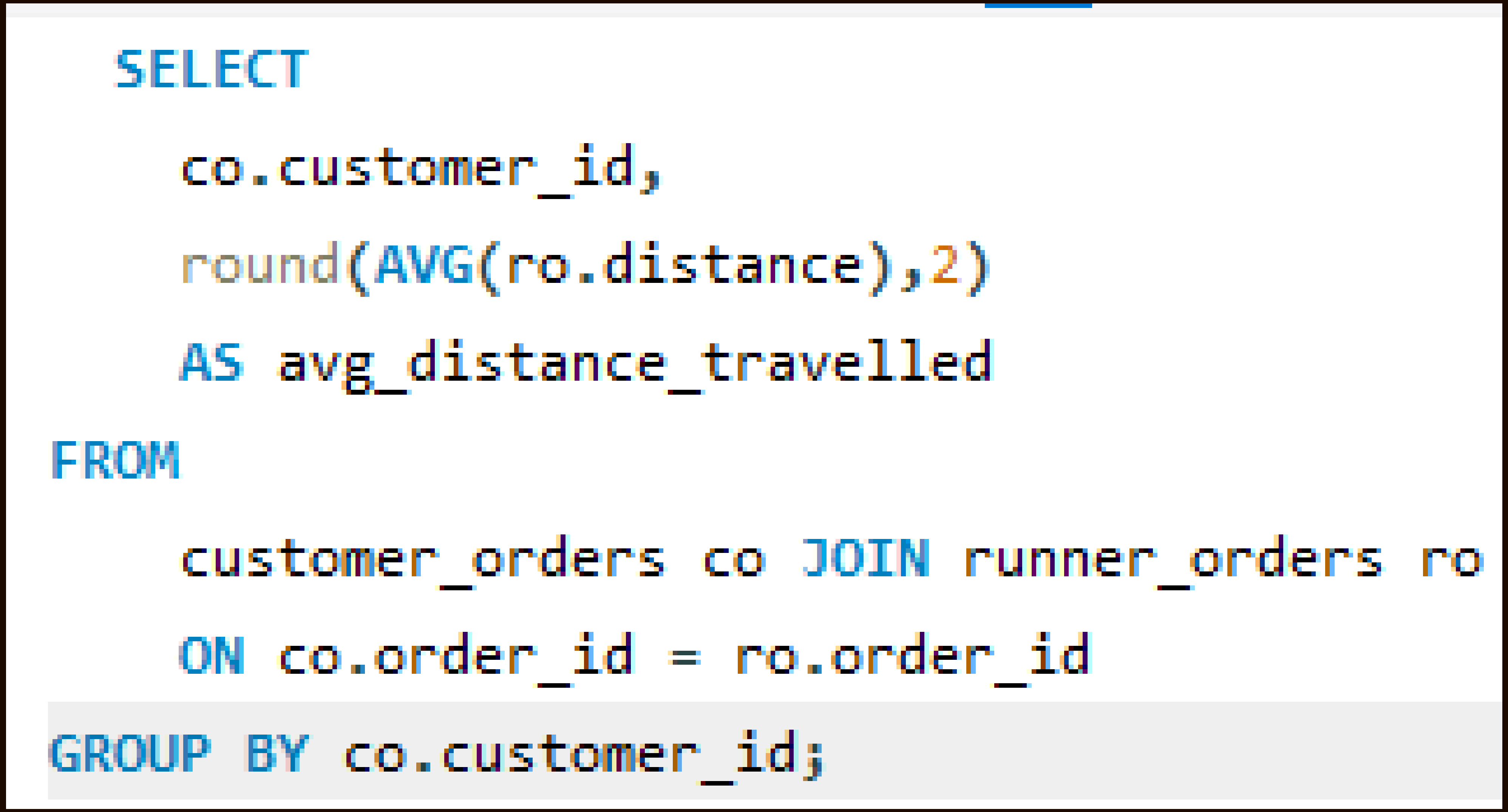


```
  SELECT r_o.runner_id,
    AVG(TIMESTAMPDIFF(MINUTE, c_o.order_time, r_o.pickup_time)) AS minute_taken
FROM runner_orders r_o join customer_orders c_o on r_o.order_id = c_o.order_id
WHERE c_o.order_time IS NOT NULL AND r_o.pickup_time IS NOT NULL
GROUP BY r_o.runner_id;
```

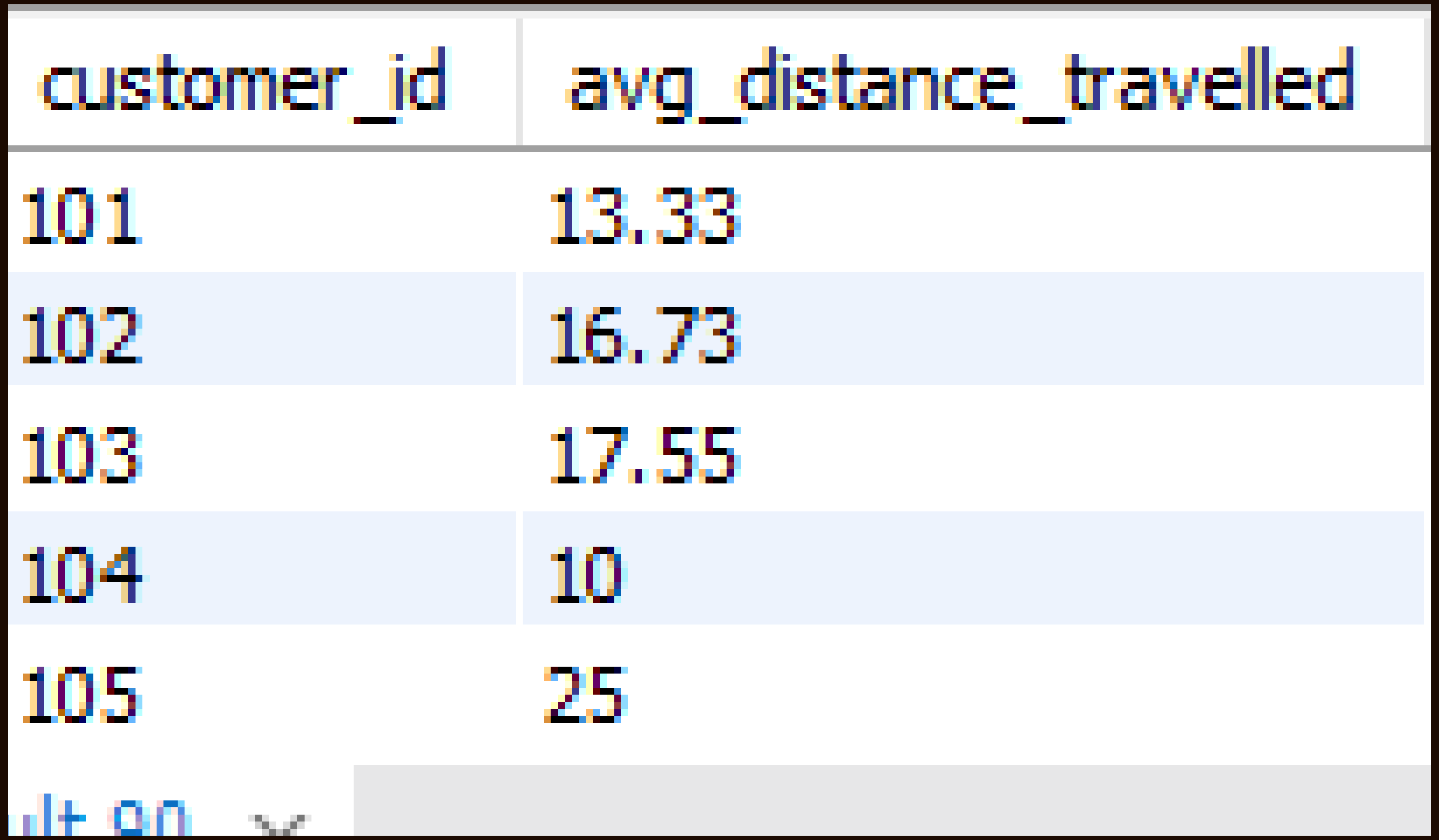


	runner_id	minute_taken
▶	1	15.3333
	2	23.4000
	3	10.0000

Q12. WHAT WAS THE AVERAGE DISTANCE TRAVELLED FOR EACH CUSTOMER?

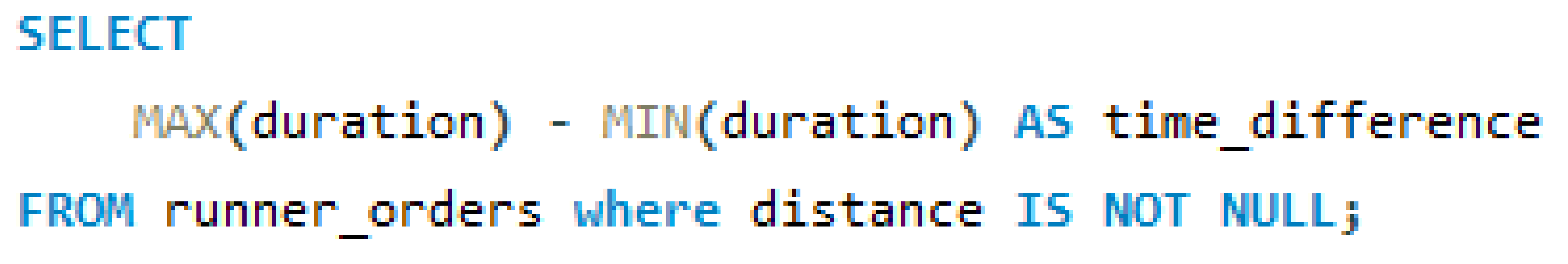


```
  SELECT
    co.customer_id,
    round(AVG(ro.distance),2)
    AS avg_distance_travelled
FROM
    customer_orders co JOIN runner_orders ro
    ON co.order_id = ro.order_id
GROUP BY co.customer_id;
```

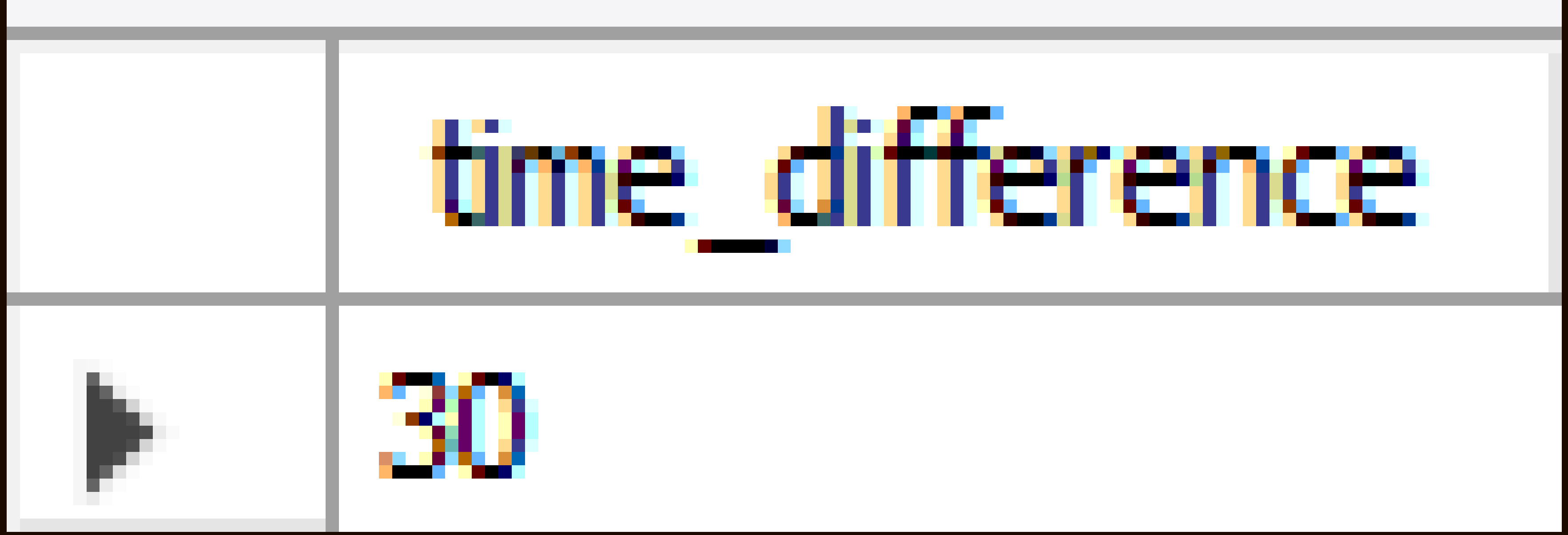


customer_id	avg_distance_travelled
101	13.33
102	16.73
103	17.55
104	10
105	25

Q13. WHAT WAS THE DIFFERENCE BETWEEN LARGEST AND SHORTEST DELIVERY TIMES FOR ALL ORDER?

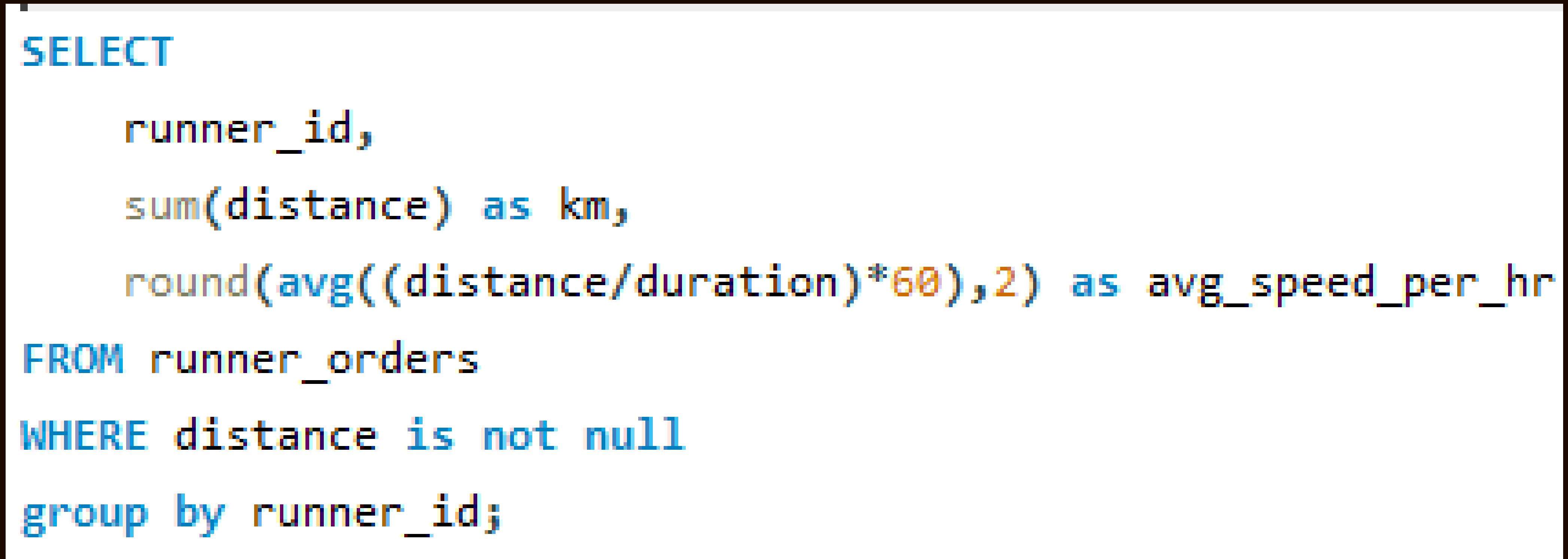


```
SELECT
    MAX(duration) - MIN(duration) AS time_difference
FROM runner_orders where distance IS NOT NULL;
```

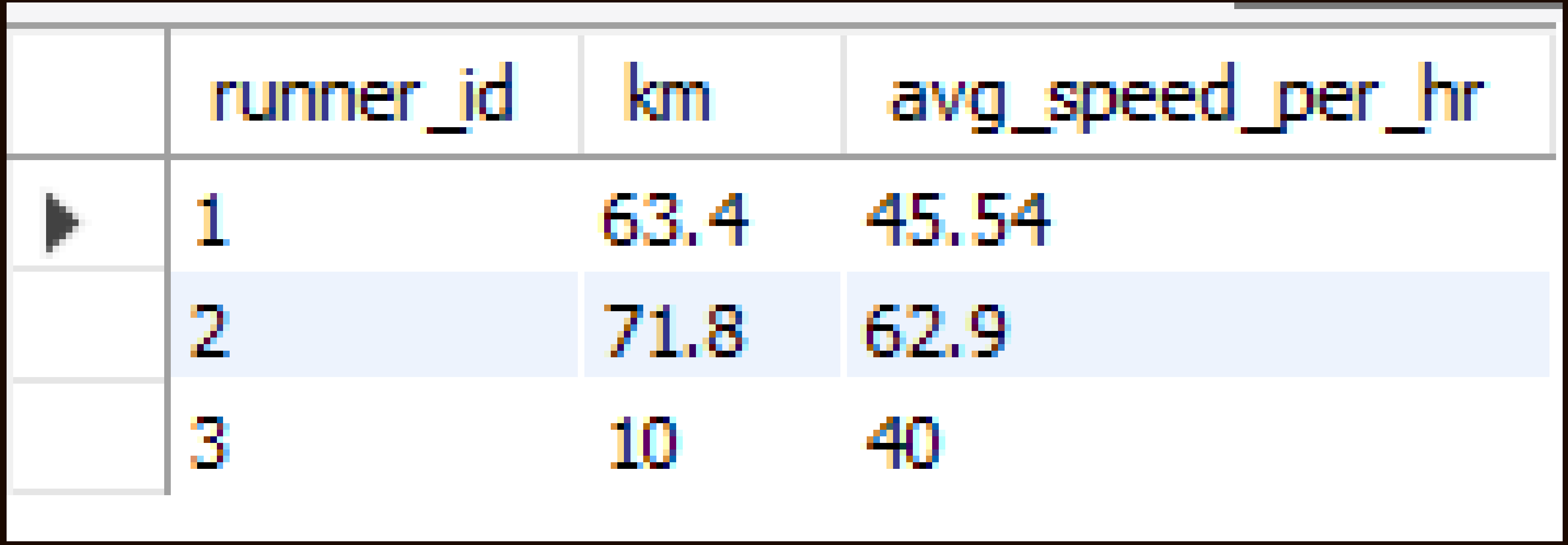


	time_difference
▶	30

Q14. WHAT WAS THE AVERAGE SPEED FOR EACH RUNNER FOR EACH DELIVERY?

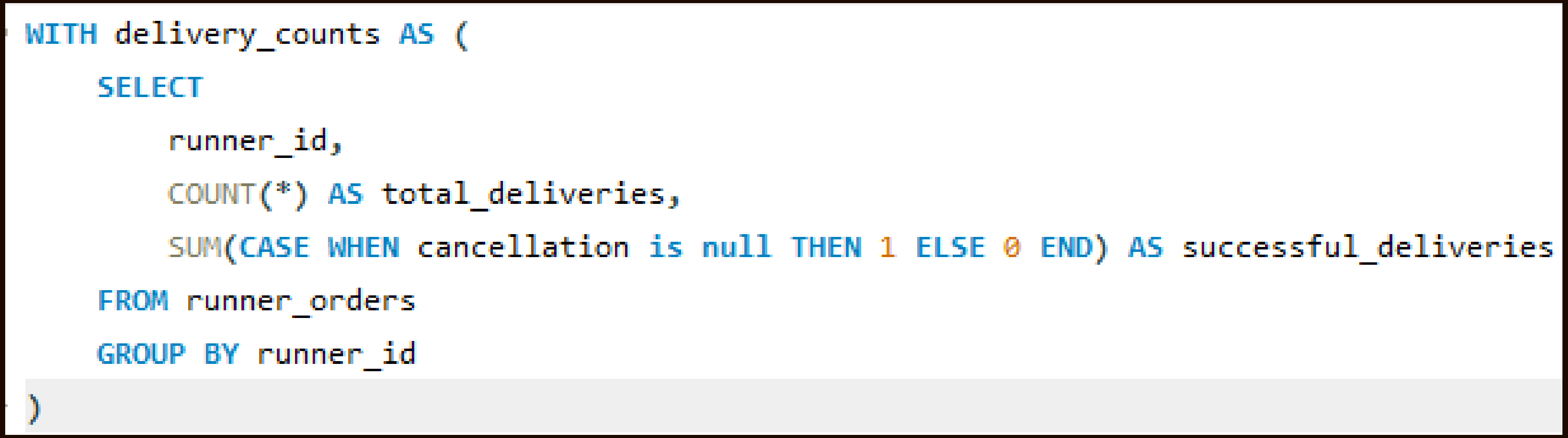


```
SELECT
    runner_id,
    sum(distance) as km,
    round(avg((distance/duration)*60),2) as avg_speed_per_hr
FROM runner_orders
WHERE distance is not null
group by runner_id;
```

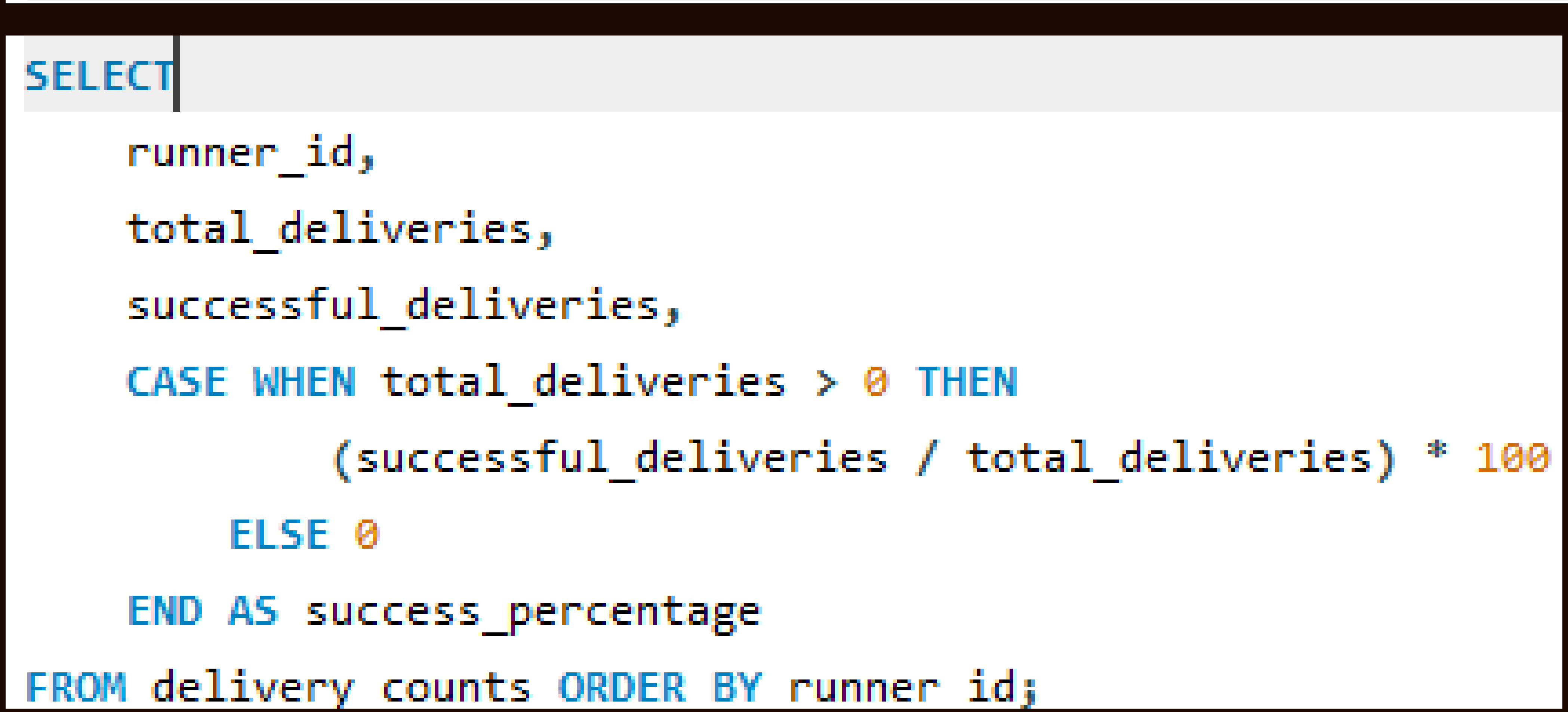


runner_id	km	avg_speed_per_hr
1	63.4	45.54
2	71.8	62.9
3	10	40

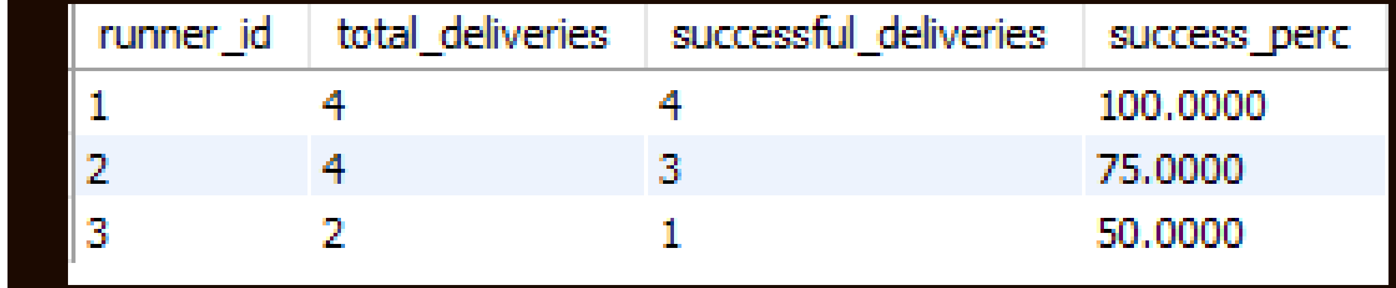
Q15. WHAT IS THE SUCCESSFULL DELIVERY % FOR EACH RUNNER?



```
WITH delivery_counts AS (
    SELECT
        runner_id,
        COUNT(*) AS total_deliveries,
        SUM(CASE WHEN cancellation is null THEN 1 ELSE 0 END) AS successful_deliveries
    FROM runner_orders
    GROUP BY runner_id
)
```



```
SELECT
    runner_id,
    total_deliveries,
    successful_deliveries,
    CASE WHEN total_deliveries > 0 THEN
            (successful_deliveries / total_deliveries) * 100
        ELSE 0
    END AS success_percentage
FROM delivery_counts ORDER BY runner_id;
```



runner_id	total_deliveries	successful_deliveries	success_perc
1	4	4	100.0000
2	4	3	75.0000
3	2	1	50.0000