SQL CASE STUDY

BURGER BASH

INTRODUCTION

Alex have started a new business of selling burger because he read on his Instagram feed that “Burger Is the Future!”

But he knew that burger alone was not going to help him get seed funding to expand his new Burger Empire - so he had one more genius idea to combine with it - He was going to Uberize it - and so Burger Runner was launched!

He started by recruiting “runners” to deliver fresh burger from Burger Runner Headquarters and also maxed out his credit card to pay freelance developers to build a mobile app to accept orders from customers.

Schema Used

runner_orders	
order_id	int
runner_id	Int
pickup_time	timestamp
distance	varchar
duration	varchar
cancellation	varchar

burger_names	
burger_id	int
burger_name	varchar

customer_orders	
order_id	int
customer_id	Int
burger_id	int
exclusions	varchar
extras	varchar
order_time	timestamp

burger_runner	
runner_id	int
registration_date	date

Question 1

How many burgers were ordered ?

```
1 •    SELECT COUNT(*) as 'no of orders'
2      FROM runner_orders;
3
```

	no of orders
▶	10

Question 2

How many unique customer orders were made ?

```sql
SELECT COUNT(DISTINCT order_id) AS unique_orders
FROM customer_orders;
```

	unique_orders
▶	10

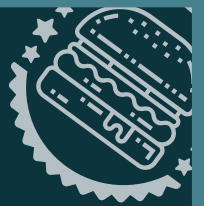
Question 3

How many successful orders were delivered by each runner ?

```sql
SELECT
    runner_id,
    COUNT(DISTINCT order_id) AS successful_orders
FROM runner_orders
WHERE cancellation IS NULL
GROUP BY runner_id
ORDER BY successful_orders DESC;
```

runner_id	successful_orders
1	4
2	3
3	1

Question 4



How many of each type of burger was delivered ?

```sql
SELECT p.burger_name, COUNT(c.burger_id) AS delivered_burger_count
FROM customer_orders AS c
JOIN runner_orders AS r
    ON c.order_id = r.order_id
JOIN burger_names AS p
    ON c.burger_id = p.burger_id
WHERE r.distance !=0
GROUP BY p.burger_name;
```

burger_name	delivered_burger_count
Meatlovers	9
Vegetarian	3

Question 5

How many Vegetarian and Meatlovers were ordered by each customer ?

```sql
SELECT c.customer_id, p.burger_name, COUNT(p.burger_name) AS order_count
FROM customer_orders AS c
JOIN burger_names AS p
    ON c.burger_id = p.burger_id
GROUP BY c.customer_id, p.burger_name
ORDER BY c.customer_id;
```

customer_id	burger_name	order_count
101	Meatlovers	2
101	Vegetarian	1
102	Meatlovers	2
102	Vegetarian	1
103	Meatlovers	3
103	Vegetarian	1
104	Meatlovers	3
105	Vegetarian	1

Question 6

What was the maximum number of burgers delivered in a single order ?

```sql
WITH burger_count_cte AS
(
SELECT c.order_id, COUNT(c.burger_id) AS burger_per_order
FROM customer_orders AS c
JOIN runner_orders AS r
    ON c.order_id = r.order_id
WHERE r.distance !=0
GROUP BY c.order_id
)
SELECT MAX(burger_per_order) AS burger_count
FROM burger_count_cte;
```

	burger_count
▶	3

Question 7

For each customer, how many delivered burgers 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 at_least_1_change,
SUM(CASE
    WHEN c.exclusions = ' ' AND c.extras = ' ' THEN 1
    ELSE 0
    END) AS no_change
FROM customer_orders AS c
JOIN runner_orders AS r
    ON c.order_id = r.order_id
WHERE r.distance !=0
GROUP BY c.customer_id
ORDER BY c.customer_id;
```

customer_id	at_least_1_change	no_change
101	0	0
102	0	0
103	3	0
104	2	0
105	1	0

Question 8

What was the total volume of burgers ordered for each hour of the day ?

```
1 •  SELECT EXTRACT(HOUR FROM order_time) AS hour_of_day,
2        COUNT(order_id) AS burger_count
3    FROM customer_orders
4    GROUP BY EXTRACT(HOUR from order_time);
```

hour_of_day	burger_count
18	3
19	1
23	3
13	3
21	3
11	1

Question 9

How many runners signed up for each 1 week period ?

```sql
SELECT EXTRACT(WEEK FROM registration_date) AS registration_week,
    COUNT(runner_id) AS runner_signup
FROM burger_runner
GROUP BY EXTRACT(WEEK FROM registration_date);
```

registration_week	runner_signup
0	1
1	2
2	1

Question 10

What was the average distance travelled for each customer ?

```sql
SELECT c.customer_id, AVG(r.distance) AS avg_distance
FROM customer_orders AS c
JOIN runner_orders AS r
    ON c.order_id = r.order_id
WHERE r.duration !=0
GROUP BY c.customer_id;
```

customer_id	avg_distance
101	20
102	16.733333333333334
103	23.399999999999995
104	10
105	25

Thank You

Dini Nabila Martiani

dininabilaa33@gmail.com

dininabilam