# 8 Week SQL Challenge

DANNY’S DINNER

1. What is the total amount each customer spent at the restaurant?

SELECT

SUM(m.price) as price\_percustomer, s.customer\_id

FROM dannys\_diner.menu m

JOIN dannys\_diner.sales s

ON m.product\_id=s.product\_id

GROUP BY s.customer\_id;

| price\_percustomer | customer\_id |

| ----------------- | ----------- |

| 74 | B |

| 36 | C |

| 76 | A |

2. How many days has each customer visited the restaurant?

SELECT

customer\_id, COUNT( DISTINCT order\_date)

FROM dannys\_diner.sales

GROUP BY customer\_id;

|  |  |
| --- | --- |
| **customer\_id** | **count** |
| A | 4 |
| B | 6 |
| C | 2 |

3. What was the first item from the menu purchased by each customer?

SELECT

DISTINCT customer\_id, LEFT(CAST(MIN(order\_date) as varchar), 10) as first\_time\_entrance

FROM dannys\_diner.sales

GROUP BY customer\_id;

|  |  |
| --- | --- |
| **customer\_id** | **first\_time\_entrance** |
| B | 2021-01-01 |
| C | 2021-01-01 |
| A | 2021-01-01 |

4. What is the most purchased item on the menu and how many times was it purchased by all customers?

SELECT COUNT(sales.product\_id), sales.customer\_id, menu.product\_name

FROM dannys\_diner.sales

JOIN dannys\_diner.menu ON

sales.product\_id = menu.product\_id

WHERE sales.product\_id =

(SELECT a.product\_id

FROM (SELECT product\_id, COUNT(product\_id)

FROM dannys\_diner.sales

GROUP BY product\_id

ORDER BY COUNT(product\_id) DESC

LIMIT 1 )

as a)

GROUP BY sales.customer\_id, menu.product\_name;

/\* Find Max Count without using max , Find id for that max count and run query on that.

\*/

|  |  |  |
| --- | --- | --- |
| **count** | **customer\_id** | **product\_name** |
| 2 | B | ramen |
| 3 | A | ramen |
| 3 | C | ramen |

5. Which item was the most popular for each customer?

SELECT product\_id , max, m.customer\_id

FROM (

SELECT MAX(y.total) as max, customer\_id

FROM

(SELECT

COUNT(product\_id) as total , product\_id,

customer\_id

FROM dannys\_diner.sales

GROUP BY customer\_id , product\_id ) as y

GROUP BY customer\_id ) as m

JOIN dannys\_diner.sales ON m.customer\_id= sales.customer\_id

GROUP BY sales.product\_id, m.max,m.customer\_id, sales.customer\_id

HAVING COUNT(sales.product\_id) = m.max

AND sales.customer\_id=m.customer\_id ;

|  |  |  |
| --- | --- | --- |
| **product\_id** | **max** | **customer\_id** |
| 3 | 3 | A |
| 3 | 3 | C |
| 3 | 2 | B |
| 1 | 2 | B |
| 2 | 2 | B |

6. Which item was purchased first by the customer after they became a member?

SELECT LEFT(CAST(MIN(y.order\_date) as varchar),10) as first\_order, y.customer\_id

FROM(

SELECT

s.order\_date, m.customer\_id

FROM dannys\_diner.sales s JOIN

dannys\_diner.members m

ON m.customer\_id = s.customer\_id

WHERE s.order\_date > m.join\_date

GROUP BY m.customer\_id, s.order\_date

ORDER BY s.order\_date) as y

GROUP BY y.customer\_id ;

|  |  |
| --- | --- |
| **first\_order** | **customer\_id** |
| 2021-01-11 | B |
| 2021-01-10 | A |

7. Which item was purchased just before the customer became a member?

SELECT LEFT(CAST(MAX(y.order\_date) as varchar),10) as first\_order\_before, y.customer\_id

FROM(

SELECT

s.order\_date, m.customer\_id

FROM dannys\_diner.sales s JOIN

dannys\_diner.members m

ON m.customer\_id = s.customer\_id

WHERE s.order\_date < m.join\_date

GROUP BY m.customer\_id, s.order\_date

ORDER BY s.order\_date) as y

GROUP BY y.customer\_id;

|  |  |
| --- | --- |
| **first\_order\_before** | **customer\_id** |
| 2021-01-04 | B |
| 2021-01-01 | A |

9. If each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many points would each customer have?

SELECT customer\_id, SUM(total\_points)\*10 as points

FROM (SELECT customer\_id , s.product\_id ,

CASE WHEN s.product\_id=1 THEN SUM(price)\*2

WHEN s.product\_id <> 1 THEN SUM(price)

END AS total\_points

FROM dannys\_diner.sales s JOIN dannys\_diner.menu m

ON s.product\_id = m.product\_id

GROUP BY customer\_id, m.product\_id, s.product\_id) as y

GROUP BY customer\_id ;

|  |  |
| --- | --- |
| **customer\_id** | **points** |
| B | 940 |
| C | 360 |
| A | 860 |

15. In the first week after a customer joins the program (including their join date) they earn 2x points on all items, not just sushi - how many points do customer A and B have at the end of January?

SELECT customer\_id, SUM(total\_points)

FROM (

SELECT customer\_id , order\_date, y.week,

CASE

WHEN y.week = 'Week1' THEN SUM(me.price)\*2

WHEN y.product\_id =1 AND y.week ='Not' THEN SUM(me.price)\*2

WHEN y.product\_id <> 1 AND y.week ='Not' THEN SUM(me.price)

END as total\_points

FROM (

SELECT order\_date , s.customer\_id, product\_id,

CASE

WHEN order\_date BETWEEN join\_date AND join\_date + 6

THEN 'Week1'

ELSE 'Not'

END AS week

FROM dannys\_diner.sales s JOIN dannys\_diner.members m

ON s.customer\_id = m.customer\_id

) y

JOIN

dannys\_diner.menu me ON

y.product\_id = me.product\_id

WHERE order\_date BETWEEN '2021-01-01' AND '2021-01-31'

GROUP BY y.customer\_id, y.order\_date,y.product\_id, y.week

) as kk

GROUP BY customer\_id;

|  |  |
| --- | --- |
| **customer\_id** | **sum** |
| B | 82 |
| A | 137 |

### **A. Pizza Metrics**

1. How many pizzas were ordered?

SELECT COUNT( \*) FROM pizza\_runner.customer\_orders;

|  |
| --- |
| **count** |
| 14 |

1. How many unique customer orders were made?

SELECT COUNT(\*) as unique\_orders

FROM (

SELECT DISTINCT pizza\_id,exclusions,extras

FROM pizza\_runner.customer\_orders) AS y;

|  |
| --- |
| **unique\_orders** |
| 10 |

1. How many successful orders were delivered by each runner?

SELECT COUNT(\*)

FROM (

SELECT

CASE WHEN cancellation IS NULL THEN 'None'

WHEN cancellation = 'null' THEN 'None'

WHEN cancellation ='' THEN 'None'

ELSE 'Cancelled'

END as Cancelled

FROM pizza\_runner.runner\_orders) as y

WHERE cancelled <> 'Cancelled';

|  |
| --- |
| **count** |
| 8 |

1. How many of each type of pizza was delivered?
2. How many Vegetarian and Meatlovers were ordered by each customer?
3. What was the maximum number of pizzas delivered in a single order?
4. For each customer, how many delivered pizzas had at least 1 change and how many had no changes?
5. How many pizzas were delivered that had both exclusions and extras?
6. What was the total volume of pizzas ordered for each hour of the day?
7. What was the volume of orders for each day of the week?

### **B. Runner and Customer Experience**

1. How many runners signed up for each 1 week period? (i.e. week starts 2021-01-01)
2. What was the average time in minutes it took for each runner to arrive at the Pizza Runner HQ to pickup the order?
3. Is there any relationship between the number of pizzas and how long the order takes to prepare?
4. What was the average distance travelled for each customer?
5. What was the difference between the longest and shortest delivery times for all orders?
6. What was the average speed for each runner for each delivery and do you notice any trend for these values?
7. What is the successful delivery percentage for each runner?

### **C. Ingredient Optimisation**

1. What are the standard ingredients for each pizza?
2. What was the most commonly added extra?
3. What was the most common exclusion?
4. 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
5. Generate an alphabetically ordered comma separated ingredient list for each pizza order from the customer\_orders table and add a 2x in front of any relevant ingredients
   * For example: "Meat Lovers: 2xBacon, Beef, ... , Salami"
6. What is the total quantity of each ingredient used in all delivered pizzas sorted by most frequent first?

### **D. Pricing and Ratings**

1. 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?
2. What if there was an additional $1 charge for any pizza extras?
   * Add cheese is $1 extra
3. The Pizza Runner team now wants to add an additional ratings system that allows customers to rate their runner, how would you design an additional table for this new dataset - generate a schema for this new table and insert your own data for ratings for each successful customer order between 1 to 5.
4. Using your newly generated table - can you join all of the information together to form a table which has the following information for successful deliveries?
   * customer\_id
   * order\_id
   * runner\_id
   * rating
   * order\_time
   * pickup\_time
   * Time between order and pickup
   * Delivery duration
   * Average speed
   * Total number of pizzas
5. If a Meat Lovers pizza was $12 and Vegetarian $10 fixed prices with no cost for extras and each runner is paid $0.30 per kilometre traveled - how much money does Pizza Runner have left over after these deliveries?

### **E. Bonus Questions**

If Danny wants to expand his range of pizzas - how would this impact the existing data design? Write an INSERT statement to demonstrate what would happen if a new Supreme pizza with all the toppings was added to the Pizza Runner menu?