**Case Study #2 - Pizza Runner**

**A. Pizza Metrics**

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

SELECT COUNT(order\_id) AS pizza\_order\_count

FROM customer\_orders

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

SELECT COUNT(DISTINCT order\_id) AS unique\_customer\_count

FROM customer\_orders

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

SELECT runner\_id,

COUNT(pickup\_time) AS count

FROM runner\_orders

WHERE pickup\_time != 'null'

GROUP BY runner\_id

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

WITH cte\_order\_id\_success\_delivered as (

SELECT order\_id

FROM runner\_orders

WHERE pickup\_time != 'null'

)

SELECT

pizza\_id,

COUNT(order\_id)

FROM customer\_orders

WHERE order\_id IN ( SELECT order\_id FROM cte\_order\_id\_success\_delivered)

GROUP BY pizza\_id

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

SELECT

customer\_id,

pizza\_name,

COUNT(pn.pizza\_id) AS pizza\_count

FROM customer\_orders as co

JOIN pizza\_names AS pn ON pn.pizza\_id = co.pizza\_id

GROUP BY customer\_id, pizza\_name

ORDER BY customer\_id

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

SELECT

order\_id,

COUNT(pizza\_id) AS pizza\_count

FROM customer\_orders

GROUP BY order\_id

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

WITH cte\_order\_id\_success\_delivered as (

SELECT order\_id

FROM runner\_orders

WHERE pickup\_time != 'null'

)

SELECT

customer\_id,

COUNT(CASE

WHEN (exclusions not in ('null','') and exclusions is not NULL)

or ( extras not in ('null','') and extras is not NULL) THEN 1

END) as at\_least\_1\_change,

count(CASE

WHEN (exclusions in ('null','') or exclusions is NULL) and ( extras in ('null','') or extras is NULL) THEN 1

END) as no\_change

FROM customer\_orders

WHERE order\_id in (SELECT order\_id FROM cte\_order\_id\_success\_delivered)

GROUP BY customer\_id

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

SELECT

count(CASE

WHEN (exclusions not in ('null','') and exclusions is not NULL) and ( extras not in ('null','') and extras is not NULL) THEN 1

END) as pizza\_count\_w\_exclusions\_extras

FROM customer\_orders

WHERE order\_id in (SELECT order\_id FROM cte\_order\_id\_success\_delivered)

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

SELECT

HOUR(order\_time) AS hour\_of\_day,

COUNT(\*) AS order\_count

FROM

customer\_orders

GROUP BY

hour\_of\_day

ORDER BY

hour\_of\_day;

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

-- ADD 1 to adjust 1st day of the week as monday

SELECT

ELT(DAYOFWEEK(order\_time + INTERVAL 1 day),'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday','Sunday') AS date\_of\_week,

COUNT(\*) AS order\_count

FROM

customer\_orders

GROUP BY

date\_of\_week

ORDER BY order\_count desc

**B. Runner and Customer Experience**

**1 - How many runners signed up for each 1 week period? (i.e. week starts 2021-01-01)**

SELECT

WEEKOFYEAR(registration\_date + INTERVAL 1 week) AS registraion\_week,

COUNT(\*) AS runner\_signup

FROM runners

GROUP BY registraion\_week

**2 - What was the average time in minutes it took for each runner to arrive at the Pizza Runner HQ to pickup the order?**

SELECT AVG(avg\_each\_order) AS avg\_pickup\_minutes

FROM (

SELECT

DISTINCT ro.order\_id,

TIMESTAMPDIFF(MINUTE,order\_time,pickup\_time) AS avg\_each\_order

FROM runner\_orders AS ro

JOIN customer\_orders AS co ON ro.order\_id = co.order\_id

) AS subquery

**3 - Is there any relationship between the number of pizzas and how long the order takes to prepare?**

SELECT pizza\_count,

AVG(avg\_pickup\_time) AS avg\_pickup\_time

FROM

(SELECT

ro.order\_id,

COUNT(pizza\_id) AS pizza\_count,

AVG(TIMESTAMPDIFF(MINUTE,order\_time,pickup\_time)) AS avg\_pickup\_time

FROM runner\_orders AS ro

JOIN customer\_orders AS co ON ro.order\_id = co.order\_id

GROUP BY ro.order\_id) AS subquery

GROUP BY pizza\_count

**4 - What was the average distance travelled for each customer?**

SELECT

customer\_id,

AVG(distance) AS avg\_distance

FROM runner\_orders AS ro

JOIN customer\_orders AS co ON ro.order\_id = co.order\_id

WHERE pickup\_time != 'null'

GROUP BY customer\_id

**5 - What was the difference between the longest and shortest delivery times for all orders?**

SELECT

MAX(CAST(duration AS FLOAT)) - MIN(CAST(duration AS FLOAT)) as delivery\_time\_difference

FROM runner\_orders

WHERE pickup\_time != 'null'

**6 - What was the average speed for each runner for each delivery and do you notice any trend for these values?**

SELECT

runner\_id,

ROUND(CAST(distance AS FLOAT)/CAST(duration AS FLOAT )\*60,2) as avg\_speed

FROM runner\_orders

WHERE pickup\_time != 'null'

**7 - What is the successful delivery percentage for each runner?**

SELECT

runner\_id,

ROUND(SUM(CASE

WHEN pickup\_time != 'null' THEN 1

ELSE 0 END

)/COUNT(\*)\*100,0) as success\_rate

FROM runner\_orders

GROUP BY runner\_id

**C. Ingredient Optimization**

**1 - What are the standard ingredients for each pizza?**

SELECT

pizza\_id,

topping\_name

FROM pizza\_recipes

JOIN pizza\_toppings

WHERE FIND\_IN\_SET(topping\_id,REPLACE(toppings,' ','')) != 0

ORDER BY pizza\_id

**2 - What was the most commonly added extra?**

SELECT

topping\_id,

topping\_name,

SUM(CASE

WHEN FIND\_IN\_SET(topping\_id,REPLACE(extras,' ','')) != 0 THEN 1

ELSE 0 END) as topping\_count

FROM pizza\_toppings

JOIN customer\_orders

WHERE (extras not in ('null','') and extras is not NULL)

GROUP BY topping\_id,topping\_name

HAVING topping\_count > 0

**3 - What was the most common exclusion?**

SELECT

topping\_id,

topping\_name,

SUM(CASE

WHEN FIND\_IN\_SET(topping\_id,REPLACE(exclusions,' ','')) != 0 THEN 1

ELSE 0 END) as topping\_count

FROM pizza\_toppings

JOIN customer\_orders

WHERE (exclusions not in ('null','') and exclusions is not NULL)

GROUP BY topping\_id,topping\_name

HAVING topping\_count > 0

**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

SELECT

CONCAT(pizza\_names.pizza\_name,

IF(exclusions NOT IN ('null','') AND exclusions IS NOT NULL,

CONCAT(' - Exclude ',

(SELECT GROUP\_CONCAT(topping\_name separator ', ')

FROM pizza\_toppings

WHERE FIND\_IN\_SET(topping\_id,REPLACE(exclusions,' ','')) != 0

GROUP BY pizza\_names.pizza\_name

)

)

, ''),

IF(extras NOT IN ('null','') AND extras IS NOT NULL,

CONCAT(' - Extra ',

(SELECT GROUP\_CONCAT(topping\_name separator ', ')

FROM pizza\_toppings

WHERE FIND\_IN\_SET(topping\_id,REPLACE(extras,' ','')) != 0

GROUP BY pizza\_names.pizza\_name

)

)

, '')) AS order\_item

FROM customer\_orders

JOIN pizza\_names ON customer\_orders.pizza\_id = pizza\_names.pizza\_id

**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?**

SELECT

SUM(CASE

WHEN pizza\_name = 'Meatlovers' THEN 12

ELSE 10 END) as total

FROM customer\_orders AS co

JOIN pizza\_names AS pn ON co.pizza\_id = pn.pizza\_id

JOIN runner\_orders AS ro ON co.order\_id = ro.order\_id

WHERE pickup\_time != 'null'

**2 - What if there was an additional $1 charge for any pizza extras?**

* Add cheese is $1 extra

SELECT

SUM(CASE

WHEN pizza\_name = 'Meatlovers' THEN IF(extras NOT IN ('null','') and extras IS NOT NULL ,LENGTH(extras) - LENGTH(REPLACE(extras, ',', '')) + 1 + 12,12)

ELSE IF(extras NOT IN ('null','') and extras IS NOT NULL ,LENGTH(extras) - LENGTH(REPLACE(extras, ',', '')) + 1 + 10,10) END) as total

FROM customer\_orders AS co

JOIN pizza\_names AS pn ON co.pizza\_id = pn.pizza\_id

JOIN runner\_orders AS ro ON co.order\_id = ro.order\_id

WHERE pickup\_time != 'null'

**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.**

CREATE TABLE ratings (

order\_id int,

rating int,

UNIQUE(order\_id),

CHECK(rating BETWEEN 1 AND 5)

);

INSERT INTO ratings VALUES

(1,1),

(2,3),

(3,5),

(4,4),

(5,3),

(7,2),

(8,5),

(10,3)

**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

SELECT

customer\_id,

co.order\_id,

runner\_id,

rating,

order\_time,

pickup\_time,

TIMESTAMPDIFF(MINUTE, order\_time, pickup\_time) AS time\_between\_order\_pickup,

CAST(duration AS FLOAT) as duration\_min,

ROUND(CAST(distance AS FLOAT)/CAST(duration AS FLOAT )\*60,2) AS avg\_speed,

COUNT(pizza\_id) AS pizza\_count

FROM customer\_orders AS co

JOIN runner\_orders AS ro ON co.order\_id = ro.order\_id

JOIN ratings AS r ON r.order\_id = co.order\_id

GROUP BY customer\_id,

co.order\_id,

runner\_id,

rating,

order\_time,

pickup\_time,

time\_between\_order\_pickup,

duration,

avg\_speed

**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?**

WITH total\_without\_delivery AS

(SELECT

co.order\_id,

SUM(CASE

WHEN pizza\_name = 'Meatlovers' THEN 12

ELSE 10 END) as total

FROM customer\_orders AS co

JOIN pizza\_names AS pn ON co.pizza\_id = pn.pizza\_id

JOIN runner\_orders AS ro ON co.order\_id = ro.order\_id

WHERE pickup\_time != 'null'

GROUP BY co.order\_id)

SELECT

ROUND(SUM(total - 0.3 \* CAST(distance AS FLOAT)) OVER(),2) AS total\_with\_delivery

FROM runner\_orders AS ro

JOIN total\_without\_delivery AS twd ON ro.order\_id = twd.order\_id

LIMIT 1