EIU

TRƯỜNG ĐẠI HỌC
QUỐC TẾ
MIỀN ĐÔNG

EASTERN
INTERNATIONAL
UNIVERSITY

FINAL PROJECT

Course: MIS 443 – Business Data Management

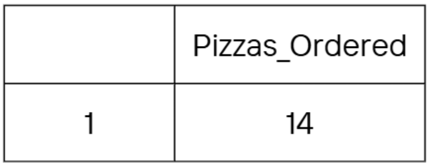
Lecturer: Mr. Dang Thai Doan

Student's Name	IRN
Trần Phương Thảo	2132300447
Thái Gia Huy	2132300332
Trần Tiến Thảo Hiếu Ngân	2032300513

Quarter 2, 2025-2026

A. Pizza Metrics

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



	Pizzas_Ordered
1	14

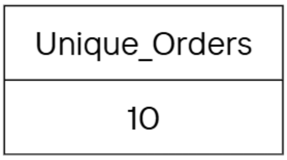
Understanding the Question: We need to find the total number of pizza orders recorded in the CUSTOMER_ORDERS table.

Steps:

- Count: All rows in the CUSTOMER_ORDERS table are counted.
- Select: The total count is selected and displayed as PIZZAS_ORDERED.

Insight: Pizza Runner received a total of 14 pizza orders.

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



Unique_Orders
10

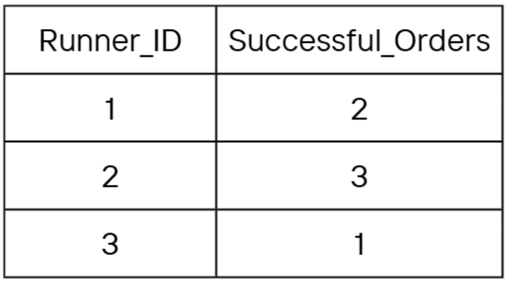
Understanding the Question: We need to find the number of distinct order IDs in the CUSTOMER_ORDERS table, indicating the number of separate orders placed.

Steps:

- Distinct: Unique values of ORDER_ID are identified in the CUSTOMER_ORDERS table.
- Count: The number of unique ORDER_ID values is counted.
- Select: The count of unique orders is selected and displayed as UNIQUE_CUSTOMER_ORDERS.

Insight: Pizza Runner received 10 unique customer orders. This means that while there were 14 pizzas ordered in total (as we saw in the previous question), some customers placed more than one order.

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



Runner_ID	Successful_Orders
1	2
2	3
3	1

Understanding the question:

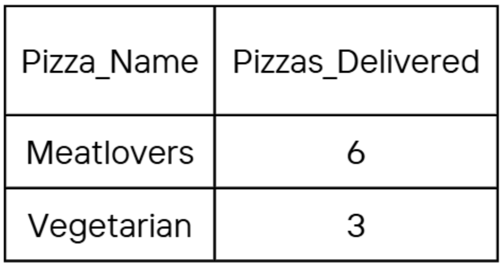
- For this question, we need to find out how many orders each runner has successfully delivered.
- "Successful" is defined by the presence of a PICKUP_TIME. If a runner has a PICKUP_TIME, we consider it a successful delivery.
- We need to group the results by RUNNER_ID to see the count for each runner individually.

Steps:

- Filter: Rows with PICKUP_TIME equal to 'NULL' are removed.
- Group: Remaining rows are grouped by RUNNER_ID.
- Count: The number of ORDER_ID values is counted for each group.
- Select: RUNNER_ID and the counts are selected and displayed as Successful_Orders.

Insight: The result shows the distribution of successful deliveries among the three runners. Runner 2 had the highest number of successful deliveries (3), followed by Runner 1 (2), and Runner 3 (1).

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



Pizza_Name	Pizzas_Delivered
Meatlovers	6
Vegetarian	3

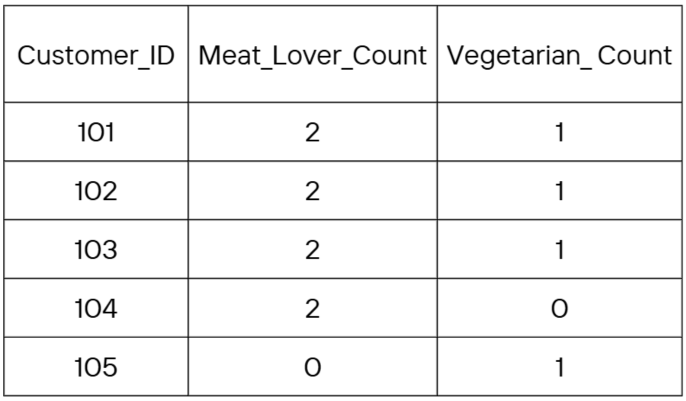
Understanding the Question: We need to determine the count of successful deliveries for each type of pizza (Meatlovers and Vegetarian). We will need to join multiple tables to get the pizza names and filter for successful deliveries.

Steps:

- Join: Join RUNNER_ORDERS, CUSTOMER_ORDERS, and PIZZA_NAMES tables.
- Filter: Filter for successful deliveries (PICKUP_TIME not 'null').
- Group: Group by PIZZA_NAME.
- Count: Count PIZZA_ID for each group.
- Select: Select PIZZA_NAME and the counts as PIZZAS_DELIVERED.

Insight: Meatlovers was the most delivered pizza type (6), double the number of Vegetarian pizzas delivered (3), suggesting a preference for Meatlovers.

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



Customer_ID	Meat_Lover_Count	Vegetarian_ Count
101	2	1
102	2	1
103	2	1
104	2	0
105	0	1

Understanding the Question: We need to find the number of Meatlovers and Vegetarian pizzas ordered by each individual customer. We need to join the CUSTOMER_ORDERS and PIZZA_NAMES tables to get the pizza names.

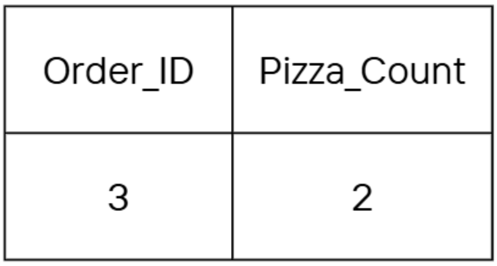
Steps:

- Join: Join CUSTOMER_ORDERS and PIZZA_NAMES tables using PIZZA_ID.
- Conditional Sum: Use SUM(CASE WHEN ...) to count Meatlovers and Vegetarian pizzas for each row.
- Group: Group by CUSTOMER_ID.
- Select: Select CUSTOMER_ID, Meatlovers counts as MEAT_LOVERS_COUNT, and Vegetarian counts as VEGETARIAN_COUNT.

Insight:

- ➢ Customers 101, 102, and 103 all ordered 2 Meatlovers and 1 Vegetarian pizza, indicating a consistent order pattern among these customers.
- ➢ Customer 104 exclusively ordered Meatlovers pizzas (2), showing a preference for meat-based options.
- ➢ Customer 105 ordered only a Vegetarian pizza, highlighting a preference for vegetarian options.
- ➢ This data reveals customer preferences and can be used for targeted marketing or menu adjustments.

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



Order_ID	Pizza_Count
3	2

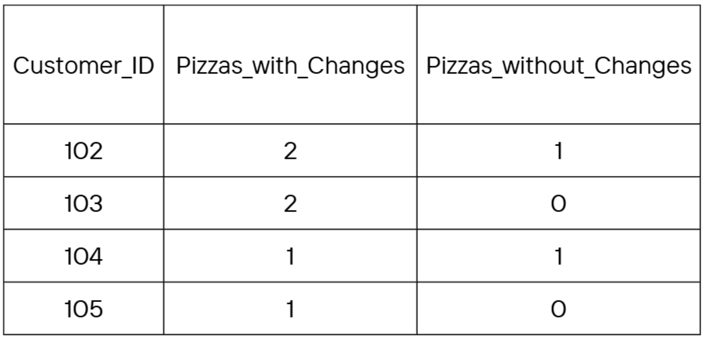
Understanding the Question: We need to find the order ID that had the highest number of pizzas ordered. We will need to count the pizzas for each order and then find the maximum count.

Steps:

- Filter: Filter CUSTOMER_ORDERS table to exclude rows where PIZZA_ID is NULL.
- Group: Group the remaining rows by ORDER_ID.
- Count: Count the PIZZA_ID values for each ORDER_ID group.
- Order: Order the results in descending order based on the PIZZA_COUNT.
- Limit: Limit the result to the first row (the row with the highest PIZZA_COUNT).
- Select: Select the ORDER_ID and the PIZZA_COUNT.

Insight: Order ID 3 had the maximum number of pizzas ordered in a single order, with a total of 2 pizzas.

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



Customer_ID	Pizzas_with_Changes	Pizzas_without_Changes
102	2	1
103	2	0
104	1	1
105	1	0

Understanding the Question: We need to find, for each customer, how many of their delivered pizzas had changes (exclusions or extras) and how many had no changes. We will need to join CUSTOMER_ORDERS and RUNNER_ORDERS, filter for delivered orders, and then use conditional aggregation based on the EXCLUSIONS and EXTRAS columns.

Steps:

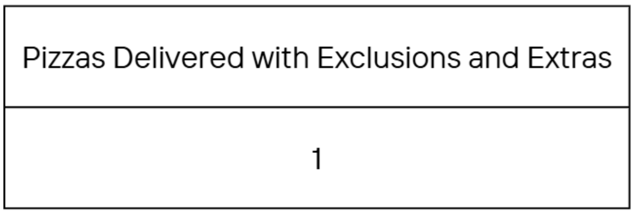
- Join: Join CUSTOMER_ORDERS and RUNNER_ORDERS tables on ORDER_ID.
- Filter: Filter for delivered orders by checking PICKUP_TIME is not 'null'.

- Conditional Sum (Changes): For each customer, sum 1 if EXCLUSIONS or EXTRAS is not null, not 'null', and has length > 0, else sum 0.
- Conditional Sum (No Changes): For each customer, sum 1 if EXCLUSIONS and EXTRAS are both null, 'null', or have length 0, else sum 0.
- Group: Group by CUSTOMER_ID.
- Select: Select CUSTOMER_ID, sum of changes as Pizzas_with_Changes, and sum of no changes as Pizzas_without_Changes.

Insight:

➢ Customer 102 had 2 pizzas with changes and 1 without.
➢ Customer 103 had 2 pizzas with changes and none without.
➢ Customer 104 had 1 pizza with changes and 1 without.
➢ Customer 105 had 1 pizza with changes and none without.

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



Pizzas Delivered with Exclusions and Extras
1

Understanding the Question: We need to find the total number of pizzas that were delivered and had both exclusions and extras specified. We will need to join CUSTOMER_ORDERS and RUNNER_ORDERS, filter for delivered orders, and then filter further for rows where both EXCLUSIONS and EXTRAS are not null, not 'null', and have a length greater than 0.

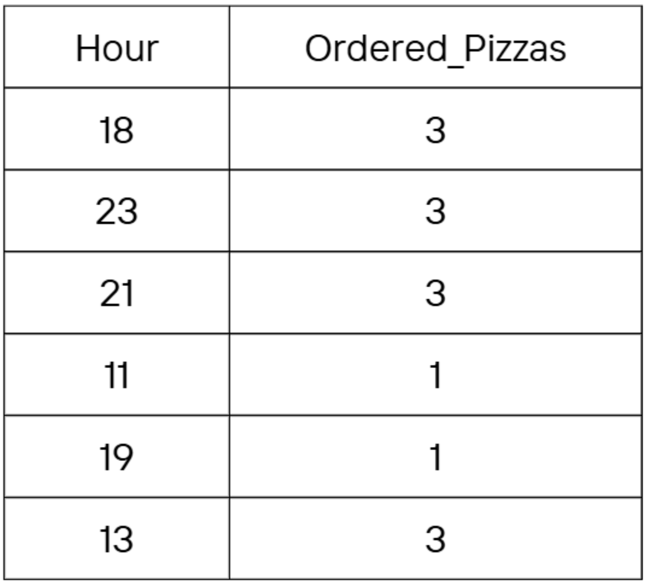
Steps:

- Join: Join CUSTOMER_ORDERS and RUNNER_ORDERS tables on ORDER_ID.
- Filter (Delivery): Filter for delivered orders by checking PICKUP_TIME is not 'null'.
- Filter (Exclusions): Filter for rows where EXCLUSIONS is not null, not 'null', and has length > 0.

- Filter (Extras): Filter for rows where EXTRAS is not null, not 'null', and has length > 0.
- Count: Count the PIZZA_ID values from the filtered rows.
- Select: Select the count as PIZZAS_DELIVERED_WITH_EXCLUSIONS_AND_EXTRAS.

Insight: Only 1 pizza was delivered that had both exclusions and extras specified. This indicates that it's relatively rare for customers to request both exclusions and extras on their pizzas.

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



Hour	Ordered_Pizzas
18	3
23	3
21	3
11	1
19	1
13	3

Understanding the Question: We need to find the total number of pizzas ordered during each hour of the day. We will need to extract the hour from the ORDER_TIME column and then count the number of orders for each hour.

Steps:

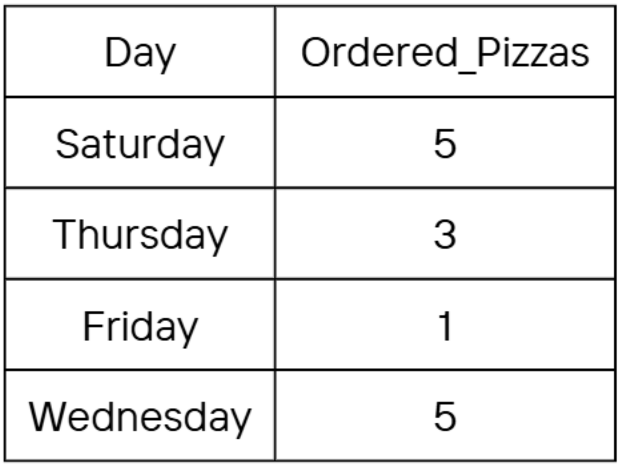
- Extract Hour: Extract the hour from the ORDER_TIME column using DATE_PART('HOUR', ORDER_TIME).
- Group: Group the rows by the extracted hour.
- Count: Count the number of rows (pizzas) in each hour group.
- Select: Select the extracted hour as HOUR and the counts as ORDERED_PIZZAS.

Insight:

➢ The highest order volumes occurred at hours 13, 18, 21, and 23, with 3 pizzas ordered each.

➢ The lowest order volumes occurred at hours 11 and 19, with 1 pizza ordered each.

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



Day	Ordered_Pizzas
Saturday	5
Thursday	3
Friday	1
Wednesday	5

Understanding the Question: We need to find the total number of pizza orders placed on each day of the week. We will need to extract the day of the week from the ORDER_TIME column and then count the number of orders for each day.

Steps:

- Extract Day: Extract the day of the week from the ORDER_TIME column using TO_CHAR(ORDER_TIME, 'DAY'). This will return the full name of the day.
- Group: Group the rows by the extracted day.
- Count: Count the number of rows (pizzas) in each day group.
- Select: Select the extracted day as DAY and the counts as ORDERED_PIZZAS.

Insight:

➢ Saturday and Wednesday had the highest order volumes, with 5 pizzas ordered each.

➢ Thursday had a moderate order volume of 3 pizzas.

- ➢ Friday had the lowest order volume, with only 1 pizza ordered.

Overall Conclusion for Pizza Metrics

Pizza Runner is in its early stages of operation, characterized by a relatively low order volume. A total of 14 pizzas were ordered across 10 unique customer orders, indicating some repeat business.

Order Distribution and Preferences:

- **Meatlovers is the most popular pizza**, with twice as many deliveries as Vegetarian.
- Customers tend to **order a mix of Meatlovers and Vegetarian**, with a few exceptions showing strong preferences for one or the other.
- **Most orders involve some form of customization**, as seen by the frequency of exclusions and extras. However, orders with both exclusions and extras are rare.

Operational Insights:

- **Order ID 3 represents the largest single order**, with 2 pizzas.
- **Runner 2 is the most active and efficient**, delivering the most orders. Runner 3's low delivery count suggests potential issues.
- **Saturday and Wednesday are the busiest days**, while Friday is the slowest.
- **Peak order hours are 13:00, 18:00, 21:00, and 23:00**, suggesting evening and late-night demand.

B. Runner and Customer Experience

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

Registration_Week	Runners_Registerd
Week 01	2
Week 02	1
Week 03	1

Understanding the Question: We need to determine the number of runners who registered within each weekly period, starting from the first week of 2021. We will use the WEEK() function to extract the week number from the registration date. We will then count the number of runners for each extracted week.

Steps:

- Filter: Filter the runners table to include only registrations on or after '2021-01-01'.
- Extract Week: Extract the week number from the registration_date using WEEK(registration_date, 1) + 1. The 1 as second argument in week function, means that week starts on monday. We add 1 to the week number because week number started from 0.
- Group: Group the results by the extracted registration_week.
- Count: Count the number of runner_id within each week group.
- Select: Select the registration_week and the runner_signup count.
- Order: Order the results by registration_week.

Insight:

➢ Week 01 (starting 2021-01-01) saw the highest number of runner sign-ups, with 2 runners registering.

➢ Weeks 02 and 03 each saw 1 runner sign up.

➢ There appears to be a decrease in runner sign-ups after the first week of 2021.

2. What was the average time in minutes it took for each runner to arrive at the Pizza Runner HQ to pick up the order?

Runner_ID	Avg_Minutes
3	10
2	20
1	14

Understanding the Question: We need to calculate the average time, in minutes, that each runner took to travel from the customer's order time to the pickup time, using the runner_orders and customer_orders tables, and the TO_TIMESTAMP(), EXTRACT(), AVG(), ROUND(), and CASE WHEN THEN ELSE END functions. We will use the pickup_time from the runner_orders table and the order_time from the customer_orders table. We will convert the time difference from seconds to minutes and handle any null values.

Steps:

- Clean Pickup Time: Transform the pickup_time column to a timestamp, handling 'null' or empty strings.
- Calculate Time in Seconds: Join runner_orders and customer_orders on order_id to get both pickup_time and order_time. Then, calculate the difference between these times in seconds.
- Convert to Minutes: Convert the time difference from seconds to minutes.
- Calculate Average: Calculate the average time in minutes for each runner.
- Round: Round the average time to the nearest whole minute.
- Group: Group the results by runner ID.

Insight:

- ➢ Runner 2 took the longest time on average to arrive at the Pizza Runner HQ for pickup, with an average of 20 minutes.
- ➢ Runner 3 had the shortest average pickup time, taking 10 minutes.
- ➢ Runner 1's average pickup time was 14 minutes, placing them in the middle.

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

Pizza_Ordered	Avg_Seconds
1	741.4000000000000000
2	1102.5000000000000000
3	1757.0000000000000000

Understanding the Question: We want to find out if there's a connection between the number of pizzas ordered and the time it takes to prepare the order, using the runner_orders and customer_orders tables, and the TO_TIMESTAMP(), EXTRACT(), AVG(), COUNT(), MIN() and CASE WHEN THEN ELSE END functions. We will calculate the preparation time by finding the difference between pickup_time and order_time.

Steps:

- Clean and Prepare runner_orders: Clean the pickup_time column and prepare runner order data.
- Count Pizzas and Calculate Order Time: Count the number of pizzas ordered for each order_id in customer_orders and get the minimum order time.
- Calculate Time Difference: Join the cleaned runner_orders data with the pizza counts and order times. Calculate the time difference between pickup_time and order_time in seconds.
- Calculate Average Time: Calculate the average preparation time (in seconds) for each number of pizzas ordered.
- Group: Group the results by the number of pizzas ordered.
- Display: Show the number of pizzas ordered and the corresponding average preparation time.
- Order: Sort the results by the number of pizzas ordered.

Insight:

- ➢ 1 pizza: 741.4 seconds (approx. 12.35 minutes).
- ➢ 2 pizzas: 1102.5 seconds (approx. 18.38 minutes).
- ➢ 3 pizzas: 1757 seconds (approx. 29.28 minutes).
- ➢ Preparation time increases with pizza count.

4. What was the average distance travelled for each customer?

Customer_ID	Avg_Distance_Travelled
101	20.00000
102	16.73333
103	23.40000
104	10.00000
105	25.00000

Understanding the Question: We need to find the average distance travelled by each customer, using the runner_orders and customer_orders tables, and the AVG(), REPLACE(), and type casting functions. We will extract the numeric distance from the DISTANCE column and calculate the average for each customer.

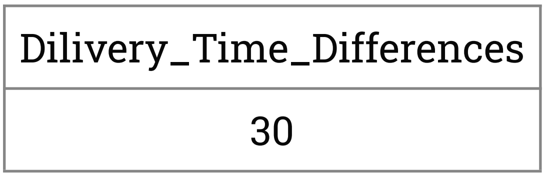
Steps:

- Join Tables: Join the runner_orders and customer_orders tables on order_id.
- Extract Numeric Distance: Remove the 'KM' from the DISTANCE column and convert the remaining value to a numeric type.
- Filter Null Distances: Filter out rows where the distance is 'NULL'.
- Calculate Average: Calculate the average distance for each customer.
- Group: Group the results by customer_id.

Insight:

- Customer 105 traveled an average distance of 25.0 kilometers.
- Customer 103 traveled an average distance of 23.4 kilometers.
- Customer 101 traveled an average distance of 20.0 kilometers.
- Customer 102 traveled an average distance of 16.73 kilometers.
- Customer 104 traveled an average distance of 10.0 kilometers.
- Distance varies significantly by customer.

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



Dilivery_Time_Differences
30

Understanding the Question: We need to find the difference between the maximum and minimum delivery times, using the runner_orders table, and the MAX() and MIN() functions. We will find the maximum and minimum values of the DURATION column and subtract them.

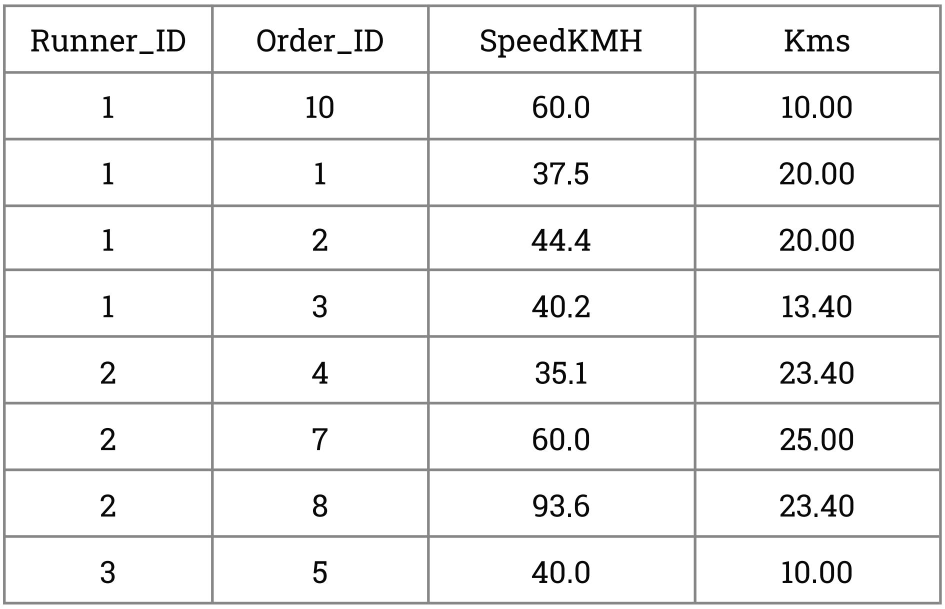
Steps:

- Filter Null Durations: Filter out rows where the DURATION is NULL.
- Find Maximum Duration: Find the maximum value of the DURATION column.
- Find Minimum Duration: Find the minimum value of the DURATION column.
- Calculate Difference: Subtract the minimum duration from the maximum duration.

Insight:

- ➢ The difference between the longest and shortest delivery times is 30 minutes.

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



Runner_ID	Order_ID	SpeedKMH	Kms
1	10	60.0	10.00
1	1	37.5	20.00
1	2	44.4	20.00
1	3	40.2	13.40
2	4	35.1	23.40
2	7	60.0	25.00
2	8	93.6	23.40
3	5	40.0	10.00

Understanding the Question: We want to calculate the average speed for each runner for each delivery, using the runner_orders table, and the ROUND(), SUM() and basic arithmetic functions. We will calculate the speed (in km/h) using the DISTANCE and DURATION columns.

Steps:

- Filter Non-Zero Distances: Filter out deliveries where the distance is 0.
- Calculate Speed: Calculate the speed in km/h using the formula (distance * 60 / duration).
- Round Speed: Round the calculated speed to one decimal place.
- Calculate total distance: Sum the distance for each runner and order.
- Group: Group the results by runner_id and order_id.
- Display: Show the runner_id, order_id, calculated speed, and total distance.
- Order: Sort the results by runner_id.

Insight:

➢ Runner 2 shows the widest range of speeds, with a notably high speed of 93.6 km/h for order 8.

➢ Runner 1's speeds are more consistent, falling within a narrower range.

➢ Runner 3 only has one recorded delivery, making it difficult to identify trends.

➢ There is no direct correlation between distance and speed, as some shorter distances have high speed and vice versa.

7. What is the successful delivery percentage for each runner?

Runner_ID	Successful_Dilivery
1	0.5
2	0.75
3	0.5

Understanding the Question: We need to calculate the percentage of successful deliveries for each runner, using the runner_orders table, and the COUNT(), SUM(), ROUND() and CASE WHEN THEN ELSE END functions. We will define a successful delivery as one where the distance is not 0.

Steps:

- Count Total Orders and Successful Deliveries: Count the total number of orders and the number of successful deliveries (distance != 0) for each runner.
- Calculate Percentage: Calculate the successful delivery percentage using the formula (successful deliveries / total orders) * 100.
- Round Percentage: Round the percentage to the nearest whole number.
- Group: Group the results by runner_id.

Insight:

➢ Runner 2 has the highest successful delivery rate at 75% (0.75).

➢ Runners 1 and 3 have the same successful delivery rate of 50% (0.5).

➢ Runner 2 is more reliable in completing deliveries compared to the other runners.

Overall Conclusion for Runner and Customer Experience

Pizza Runner's runner and customer experience data reveals insights into runner performance, delivery efficiency, and customer service. While the data set is limited, it provides a foundation for understanding operational dynamics.

Runner Performance and Sign-ups:

- Runner Sign-up Trends: The initial week of 2021 saw the highest number of runner sign-ups, with a subsequent decrease in the following weeks. This suggests a potential initial surge in interest, followed by a stabilization or decline.
- Delivery Efficiency:
  - Runner 2 demonstrates the longest average pickup time, while Runner 3 is the quickest. This suggests variations in runner efficiency or possible differences in the distances they need to travel.
  - Runner 2 also has the highest delivery succes rate, and the most variable delivery speeds.

- Delivery Speed:
  - Runner 2 exhibits the widest range of delivery speeds, including a notably high speed, indicating potential data anomalies or highly variable delivery conditions.
  - Runner 1 has the most consistent delivery speeds.

Customer Experience and Order Dynamics:

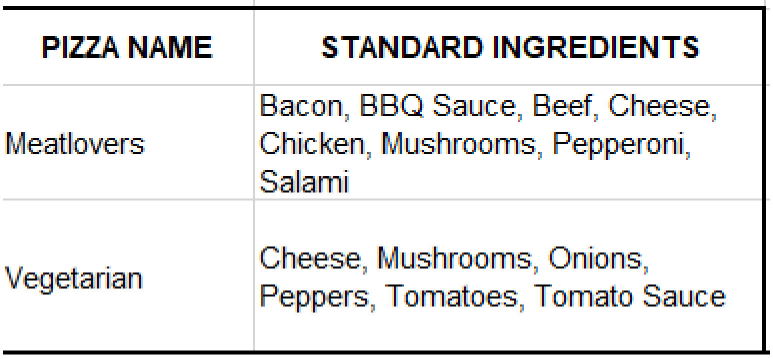
- Order Preparation Time: There is a clear positive correlation between the number of pizzas ordered and the preparation time. Larger orders take significantly longer to prepare, which is consistent with expectations.
- Customer Distance: Customer distances vary considerably, indicating a wide service area or diverse customer locations. Customer 105 lives the furthest away, and customer 104 lives the closest.
- Delivery Time Variation: There is a 30 minute difference between the fastest and slowest delivery times, indicating a wide range of delivery times.

Operational Insights:

- The data suggests a need for further investigation into **Runner 2's delivery speed variations** and **Runner 3's consistent efficiency**.
- The correlation between pizza order size and preparation time highlights the need for **efficient kitchen operations to handle larger orders**.
- The wide range of customer distances may require the business to **examine delivery zones and runner assignments**.
- The delivery success rate of each runner should be monitored, and the cause of the **lower success rates of Runner 1 and 3** should be investigated.

C. Ingredient Optimisation

1. What are the standard ingredients for each pizza?



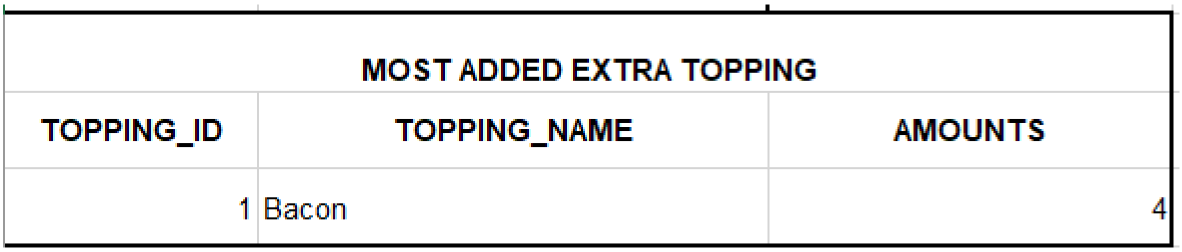
PIZZA NAME	STANDARD INGREDIENTS
Meatlovers	Bacon, BBQ Sauce, Beef, Cheese, Chicken, Mushrooms, Pepperoni, Salami
Vegetarian	Cheese, Mushrooms, Onions, Peppers, Tomatoes, Tomato Sauce

Understanding the question: we need to find the standard ingredients for each pizza in order to see what ingredients are they made of

Step:

Insight: As said in their name, both food items have unique ingredients as meatlovers are meant for meat-enjoyers and vegetarian is for plant-eater. However, Cheese and Mushrooms are included in both food items, meaning that these 2, potentially, can play an important role

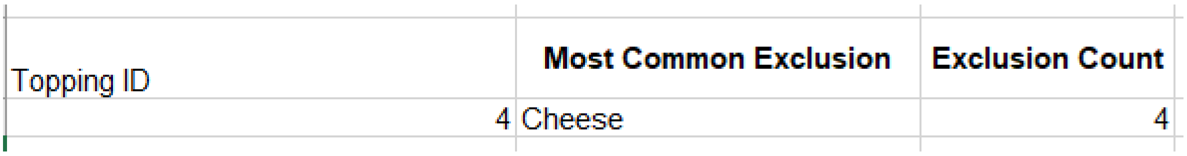
2. What was the most commonly added extra?



	MOST ADDED EXTRA TOPPING	
TOPPING_ID	TOPPING_NAME	AMOUNTS
1	Bacon	4

Insight: The topping that is most commonly used by customers is Bacon that been ordered 4 times

3. What was the most common exclusion?

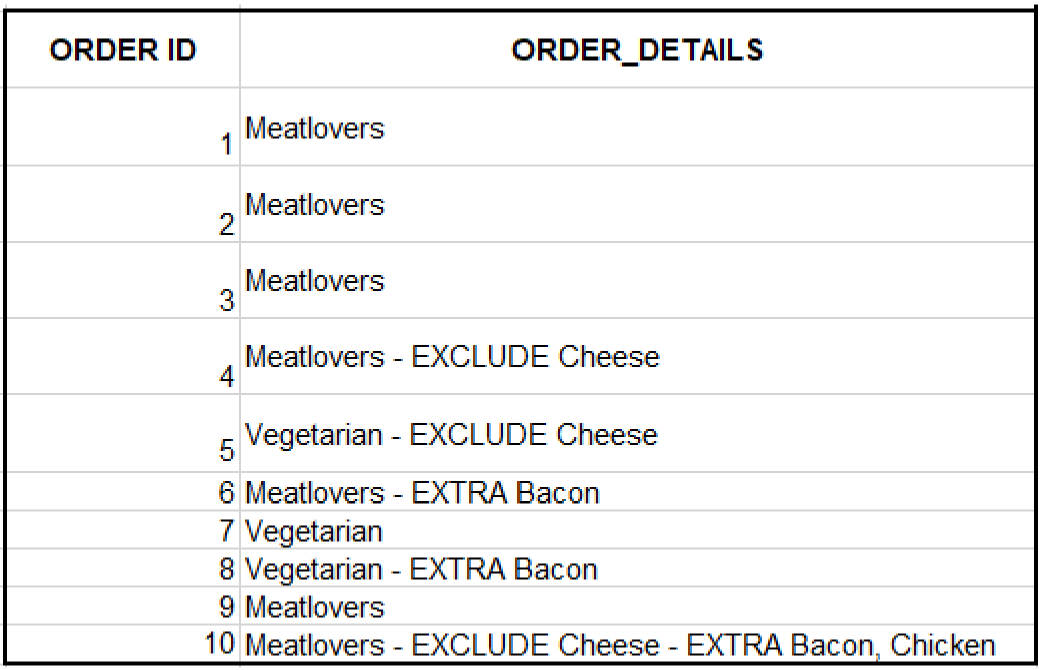


Topping ID	Most Common Exclusion	Exclusion Count
4	Cheese	4

Insight: Among ingredients, cheese is the one with the highest exclusion.

4. Generate an order item for each record in the customers_orders table in the format of one of the following:

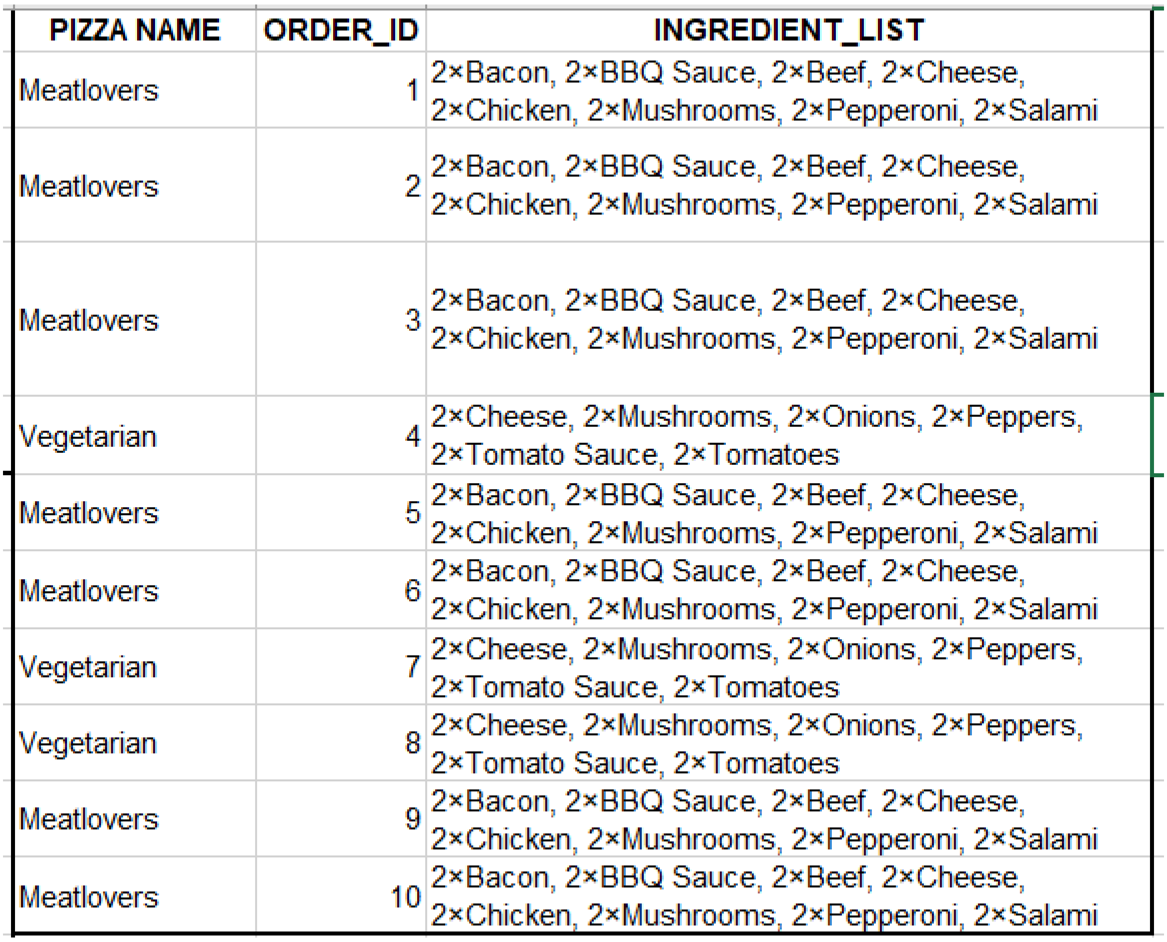
- ○ Meat Lovers
- ○ Meat Lovers - Exclude Beef
- ○ Meat Lovers - Extra Bacon
- ○ Meat Lovers - Exclude Cheese, Bacon - Extra Mushroom, Peppers



ORDER ID	ORDER_DETAILS
1	Meatlovers
2	Meatlovers
3	Meatlovers
4	Meatlovers - EXCLUDE Cheese
5	Vegetarian - EXCLUDE Cheese
6	Meatlovers - EXTRA Bacon
7	Vegetarian
8	Vegetarian - EXTRA Bacon
9	Meatlovers
10	Meatlovers - EXCLUDE Cheese - EXTRA Bacon, Chicken

Insight: This table shows each order 's exclusion and extras by including notes after the pizzas' name. Clearly seen that only half of total orders were on excluding and extraing. Cheese were the only excluded topping from both meatlovers and vegetarian. On the other hand, bacon was added even for vegetarian (order id 8)

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"



PIZZA NAME	ORDER_ID	INGREDIENT_LIST
Meatlovers	1	2×Bacon, 2×BBQ Sauce, 2×Beef, 2×Cheese, 2×Chicken, 2×Mushrooms, 2×Pepperoni, 2×Salami
Meatlovers	2	2×Bacon, 2×BBQ Sauce, 2×Beef, 2×Cheese, 2×Chicken, 2×Mushrooms, 2×Pepperoni, 2×Salami
Meatlovers	3	2×Bacon, 2×BBQ Sauce, 2×Beef, 2×Cheese, 2×Chicken, 2×Mushrooms, 2×Pepperoni, 2×Salami
Vegetarian	4	2×Cheese, 2×Mushrooms, 2×Onions, 2×Peppers, 2×Tomato Sauce, 2×Tomatoes
Meatlovers	5	2×Bacon, 2×BBQ Sauce, 2×Beef, 2×Cheese, 2×Chicken, 2×Mushrooms, 2×Pepperoni, 2×Salami
Meatlovers	6	2×Bacon, 2×BBQ Sauce, 2×Beef, 2×Cheese, 2×Chicken, 2×Mushrooms, 2×Pepperoni, 2×Salami
Vegetarian	7	2×Cheese, 2×Mushrooms, 2×Onions, 2×Peppers, 2×Tomato Sauce, 2×Tomatoes
Vegetarian	8	2×Cheese, 2×Mushrooms, 2×Onions, 2×Peppers, 2×Tomato Sauce, 2×Tomatoes
Meatlovers	9	2×Bacon, 2×BBQ Sauce, 2×Beef, 2×Cheese, 2×Chicken, 2×Mushrooms, 2×Pepperoni, 2×Salami
Meatlovers	10	2×Bacon, 2×BBQ Sauce, 2×Beef, 2×Cheese, 2×Chicken, 2×Mushrooms, 2×Pepperoni, 2×Salami

??

6. What is the total quantity of each ingredient used in all delivered pizzas sorted by most frequent first?

…