```
-- Week 2 (Pizza Metrics)

-- Following are the 3 databases that coreespons to the below case study

-- 1. Customer_orders

/*
Customer pizza orders are captured in the customer_orders table with 1 row for each individual pizza that is
part of the order.

The pizza_id relates to the type of pizza which was ordered whilst the exclusions are the ingredient_id
values which should be removed from the pizza and the extras are the ingredient_id values which need to be
added to the pizza.

Note that customers can order multiple pizzas in a single order with varying exclusions and extras values
even if the pizza is the same type!

The exclusions and extras columns will need to be cleaned up before using them in your queries.
*/

SELECT * FROM customer_orders;

/* Output
-------------------------------------------------------------------------------
order_id | customer_id |pizza_id| exclusions | extras    |     order_time
-------------------------------------------------------------------------------
   1     |    101      |   1    |            |           |  2020-01-01 18:05:02
   2     |    101      |   1    |            |           |  2020-01-01 19:00:52
   3     |    102      |   1    |            |           |  2020-01-02 23:51:23
   3     |    102      |   2    |            |           |  2020-01-02 23:51:23
   4     |    103      |   1    |      4     |           |  2020-01-04 13:23:46
   4     |    103      |   1    |      4     |           |  2020-01-04 13:23:46
   4     |    103      |   2    |      4     |           |  2020-01-04 13:23:46
   5     |    104      |   1    |    null    |     1     |  2020-01-08 21:00:29
   6     |    101      |   2    |    null    |    null   |  2020-01-08 21:03:13
   7     |    105      |   2    |    null    |     1     |  2020-01-08 21:20:29
   8     |    102      |   1    |    null    |    null   |  2020-01-09 23:54:33
   9     |    103      |   1    |      4     |    1, 5   |  2020-01-10 11:22:59
  10     |    104      |   1    |    null    |    null   |  2020-01-11 18:34:49
  10     |    104      |   1    |    2, 6    |    1, 4   |  2020-01-11 18:34:49
-------------------------------------------------------------------------------
*/

-- 2. pizza_names

-- At the moment - Pizza Runner only has 2 pizzas available the Meat Lovers or Vegetarian!

SELECT * FROM pizza_names;

 /* Output
 ---------------------
 pizza_id | pizza_name
 ---------------------
    1     | Meatlovers
    2     | Vegetarian
 ---------------------
 */

-- 3. pizza_recipes

-- Each pizza_id has a standard set of toppings which are used as part of the pizza recipe.

SELECT * FROM pizza_recipes;

/* Output
----------------------------------
pizza_id |        toppings
----------------------------------
    1    |   1, 2, 3, 4, 5, 6, 8, 10
    2    |   4, 6, 7, 9, 11, 12
----------------------------------
*/

-- 4. pizza_toppings

-- This table contains all of the topping_name values with their corresponding topping_id value

SELECT * FROM pizza_toppings;

/* Output
--------------------------------
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
--------------------------------
*/
```

```sql
-- 5. runner_orders

/*
After each orders are received through the system - they are assigned to a runner - however not all orders are fully completed and can be cancelled by the restaurant or the customer.

The pickup_time is the timestamp at which the runner arrives at the Pizza Runner headquarters to pick up the freshly cooked pizzas. The distance and duration fields are related to how far and long the runner had to travel to deliver the order to the respective customer.

There are some known data issues with this table so be careful when using this in your queries - make sure to check the data types for each column in the schema SQL!
*/

SELECT * FROM runner_orders;

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

*/

-- 6. runners

-- The runners table shows the registration_date for each new runner
SELECT * FROM runners;

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

                                        -- Data Cleaning

-- Cleaning the customer_orders table

UPDATE customer_orders
   SET exclusions = null
 WHERE exclusions = ''
    OR exclusions = 'null';

UPDATE customer_orders
   SET extras = null
 WHERE extras = ''
    OR extras = 'null';

-- Table after cleaning
SELECT * FROM customer_orders;

/* Output
-------------------------------------------------------------------------------------------------
order_id | customer_id |pizza_id| exclusions | extras    |     order_time
-------------------------------------------------------------------------------------------------
   1     |    101      |   1    |            |           |  2020-01-01 18:05:02
   2     |    101      |   1    |            |           |  2020-01-01 19:00:52
   3     |    102      |   1    |            |           |  2020-01-02 23:51:23
   3     |    102      |   2    |            |           |  2020-01-02 23:51:23
   4     |    103      |   1    |     4      |           |  2020-01-04 13:23:46
   4     |    103      |   1    |     4      |           |  2020-01-04 13:23:46
   4     |    103      |   2    |     4      |           |  2020-01-04 13:23:46
   5     |    104      |   1    |            |     1     |  2020-01-08 21:00:29
   6     |    101      |   2    |            |           |  2020-01-08 21:03:13
   7     |    105      |   2    |            |     1     |  2020-01-08 21:20:29
   8     |    102      |   1    |            |           |  2020-01-09 23:54:33
   9     |    103      |   1    |     4      |    1, 5   |  2020-01-10 11:22:59
  10     |    104      |   1    |            |           |  2020-01-11 18:34:49
  10     |    104      |   1    |    2, 6    |    1, 4   |  2020-01-11 18:34:49
-------------------------------------------------------------------------------------------------
*/

-- Cleaning the runner_orders table

UPDATE runner_orders
   SET cancellation = null
 WHERE cancellation IN ('null','');

UPDATE runner_orders
   SET distance = null
 WHERE distance = 'null';

UPDATE runner_orders
   SET duration = null
 WHERE duration = 'null';


UPDATE runner_orders
   SET distance = REPLACE(distance,'km','');

UPDATE runner_orders
   SET duration = REPLACE(duration,'minutes','');

UPDATE runner_orders
   SET duration = REPLACE(duration,'minute','');

UPDATE runner_orders
   SET duration = REPLACE(duration,'mins','');

-- Changing the datatype of some columns
  ALTER TABLE runner_orders
MODIFY COLUMN distance DECIMAL(3,1);

  ALTER TABLE runner_orders
MODIFY COLUMN duration INT;

-- Table after cleaning the data

SELECT * FROM runner_orders;

/* Output
-------------------------------------------------------------------------------------------------------------
 order_id | runner_id  |     pickup_time        |  distance  |   duration   |     cancellation
-------------------------------------------------------------------------------------------------------------
     1    |    1       | 2020-01-01 18:15:34    |      20    |     32       |
     2    |    1       | 2020-01-01 19:10:54    |      20    |     27       |
     3    |    1       | 2020-01-03 00:12:37    |     13.4   |     20       |
     4    |    2       | 2020-01-04 13:53:03    |     23.4   |     40       |
     5    |    3       | 2020-01-08 21:10:57    |       10   |     15       |
     6    |    3       |                        |            |              |   Restaurant Cancellation
     7    |    2       | 2020-01-08 21:30:45    |      25    |     25       |
     8    |    2       | 2020-01-10 00:15:02    |     23.4   |     15       |
     9    |    2       |                        |            |              |    Customer Cancellation
    10    |    1       | 2020-01-11 18:50:20    |       10   |     10       |
-------------------------------------------------------------------------------------------------------------

*/
```

```
                                -- There are 4 focus areas in Week 2
                                        -- 1.Pizza Metric

-- 1. How many pizzas were ordered?

SELECT COUNT(order_id) Pizzas_ordered
       FROM customer_orders;

/* Output
_______________
Pizzas_ordered
_______________
      14
_______________
*/

-- 2 How many unique customer orders were made?

SELECT COUNT(DISTINCT order_id,customer_id) AS unique_customer_orders
  FROM customer_orders;

/* Output
________________________
unique_customer_orders
________________________
          10
________________________
*/

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

    SELECT R.runner_id,
           COUNT(order_id) AS Orders_delivered
      FROM runners R
 LEFT JOIN runner_orders RO
        ON R.runner_id = RO.runner_id
     WHERE cancellation IS NULL
  GROUP BY R.runner_id;

/* Output
_____________________________
runner_id | Orders_delivered
_____________________________
    1     |       4
    2     |       3
    3     |       1
    4     |       0
_____________________________
*/

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

    SELECT P.pizza_id,
           P.pizza_name,
           COUNT(P.pizza_id) AS pizzas_delivered
      FROM customer_orders C
INNER JOIN runner_orders R
        ON C.order_id = R.order_id
INNER JOIN pizza_names P
        ON P.pizza_id = C.pizza_id
     WHERE cancellation IS NULL
  GROUP BY pizza_id;

/* Output
_____________________________________________
pizza_id | pizza_name | pizzas_delivered
_____________________________________________
    1    | Meatlovers  |        9
    2    | Vegetarian  |        3
_____________________________________________
*/

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

    SELECT customer_id,pizza_name,
           COUNT(pizza_name) AS No_of_pizza
      FROM customer_orders C
INNER JOIN pizza_names P
        ON C.pizza_id = P.pizza_id
  GROUP BY customer_id,pizza_name;

/* Output
_____________________________________
customer_id | pizza_name | No_of_pizza
_____________________________________
    101     | Meatlovers |       2
    102     | Meatlovers |       2
    102     | Vegetarian |       1
    103     | Meatlovers |       3
    103     | Vegetarian |       1
    104     | Meatlovers |       3
    101     | Vegetarian |       1
    105     | Vegetarian |       1
_____________________________________
*/
```

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

    SELECT C.order_id,
           COUNT(pizza_id) AS Max_delivered
      FROM customer_orders C
 LEFT JOIN runner_orders R
        ON C.order_id = R.order_id
     WHERE cancellation IS NULL
  GROUP BY order_id
  ORDER BY 2 DESC
     LIMIT 1;

/* Output
_____________________________
order_id | Max_pizzas_delivered
_____________________________
    4    |          3
_____________________________
*/

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

   SELECT customer_id,
          SUM(CASE
              WHEN exclusions IS NOT NULL OR extras IS NOT NULL THEN 1
              ELSE 0
              END) AS Atleast_one_change,
          SUM(CASE
              WHEN exclusions IS NULL AND extras IS NULL THEN 1
              ELSE 0
              END) as No_change
     FROM customer_orders C LEFT JOIN runner_orders R ON C.order_id = R.order_id WHERE cancellation IS
NULL  GROUP BY customer_id;

/* Ouput
_____________________________________________
customer_id | Atleast_one_change | No_change
_____________________________________________
    101     |         0          |    2
    102     |         0          |    3
    103     |         3          |    0
    104     |         2          |    1
    105     |         1          |    0
_____________________________________________
*/

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

    SELECT COUNT(pizza_id) AS Both_exclusion_extras
      FROM customer_orders C
INNER JOIN runner_orders R
        ON C.order_id= R.order_id
     WHERE (cancellation IS NULL) AND (exclusions IS NOT NULL) AND (extras IS NOT NULL);

/* Output
_____________________
Both_exclusion_extras
_____________________
           1
_____________________
*/

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

  SELECT DAY(order_time),
         HOUR(order_time),
         COUNT(*) AS Pizzas_ordered
    FROM customer_orders
GROUP BY DAY(order_time),HOUR(order_time);

/* Output
_____________________________________________________
DAY(order_time) | HOUR(order_time) | Pizzas_ordered
_____________________________________________________
        1       |        18        |      1
        1       |        19        |      1
        2       |        23        |      2
        4       |        13        |      3
        8       |        21        |      3
        9       |        23        |      1
       10       |        11        |      1
       11       |        18        |      2
_____________________________________________________
*/

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

  SELECT dayname(order_time) AS Day_of_week,
         COUNT(order_id) AS Day_of_week
    FROM customer_orders
GROUP BY Day_of_week;

/* Output
_____________________
Day_of_week | Day_of_week
_____________________
 Wednesday  |     5
 Thursday   |     3
 Saturday   |     5
 Friday     |     1
 _____________________
 */
```