Create a Test Metric

Log onto Mode Analytics and from the home page, create a new report by clicking on the green plus sign button in the upper right-hand corner. Enter the starter code where provided for each exercise. You may want to create a new tab for each exercise.

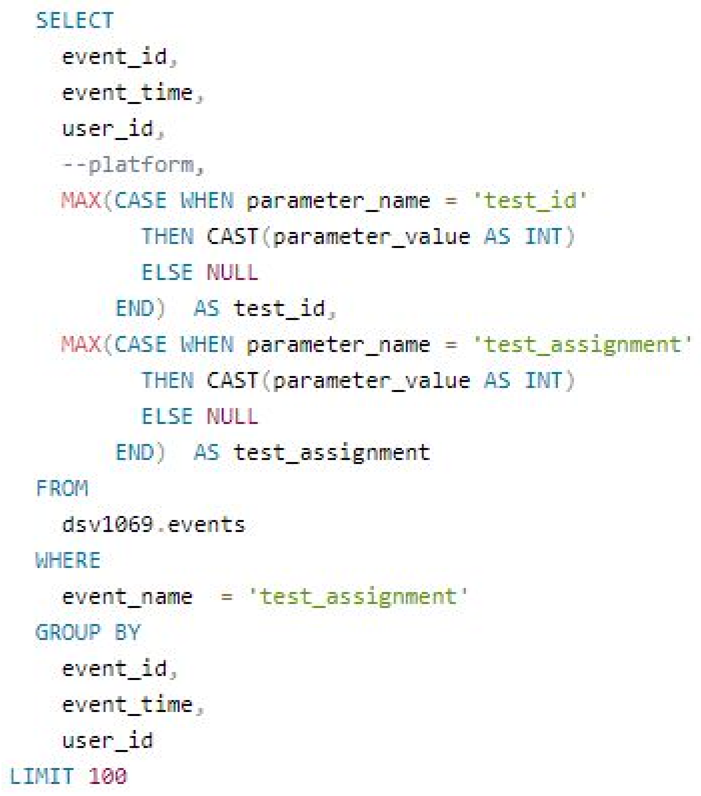
Please use the discussion forum for any questions and/or comments you might have. Once you have tried the exercises, feel free to watch the solutions video. Good luck with your practice!

Note: *When querying a table, remember to prepend dsv1069, which is the schema, or folder that contains the course data.*

Exercise 1:
--Using the table from Exercise 4.3 and compute a metric that measures
--Whether a user created an order after their test assignment

--Requirements: Even if a user had zero orders, we should have a row that counts
-- their number of orders as zero
--If the user is not in the experiment they should not be included

Starter Code:



```
  SELECT
    event_id,
    event_time,
    user_id,
    --platform,
    MAX(CASE WHEN parameter_name = 'test_id'
          THEN CAST(parameter_value AS INT)
          ELSE NULL
        END)  AS test_id,
    MAX(CASE WHEN parameter_name = 'test_assignment'
          THEN CAST(parameter_value AS INT)
          ELSE NULL
        END)  AS test_assignment
  FROM
    dsv1069.events
  WHERE
    event_name  = 'test_assignment'
  GROUP BY
    event_id,
    event_time,
    user_id
LIMIT 100
```

Exercise 2:
--Using the table from the previous exercise, add the following metrics
--1) the number of orders/invoices
--2) the number of items/line-items ordered
--3) the total revenue from the order after treatment

Starter Code:

```sql
SELECT
  test_events.test_id,
  test_events.test_assignment,
  test_events.user_id,
  COUNT(DISTINCT (CASE WHEN orders.created_at > test_events.event_time THEN invoice_id ELSE NULL END))
    AS orders_after_assignment,
  COUNT(DISTINCT (CASE WHEN orders.created_at > test_events.event_time THEN line_item_id ELSE NULL END))
    AS items_after_assignment,
  SUM((CASE WHEN orders.created_at > test_events.event_time THEN price ELSE 0 END))
    AS total_revenue
FROM
  (
  SELECT
    event_id,
    event_time,
    user_id,
    --platform,
    MAX(CASE WHEN parameter_name = 'test_id'
          THEN CAST(parameter_value AS INT)
          ELSE NULL
        END)  AS test_id,
    MAX(CASE WHEN parameter_name = 'test_assignment'
          THEN CAST(parameter_value AS INT)
          ELSE NULL
        END)  AS test_assignment
  FROM
    dsv1069.events
  WHERE
    event_name  = 'test_assignment'
  GROUP BY
    event_id,
    event_time,
    user_id
  ) test_events
LEFT JOIN
  dsv1069.orders
ON
  orders.user_id = test_events.user_id
GROUP BY
  test_events.test_id,
  test_events.test_assignment,
  test_events.user_id
```