AB Test

01 - Compare the final_assignments_qa table to the assignment events
we captured for user_level_testing. Write an answer to the following question:
Does this table have everything you need to compute metrics like 30-day view-binary?

ANSWER: No, we need the "created_at" date to analyze.

```
SELECT
    *
FROM
  dsv1069.final_assignments_qa;
```

01

	item_id	test_a	test_b	test_c	test_d	test_e	test_f
1	2512	1	0	1	1	0	1
2	482	0	1	1	1	0	0
3	2446	0	1	1	0	1	0
4	1312	0	0	0	0	0	1
5	3556	1	1	0	1	0	0
6	131	0	0	0	0	1	1
7	1178	1	0	1	0	1	1
8	110	0	1	1	1	1	0
9	47	0	0	1	0	1	1
10	1696	0	0	1	1	1	1
11	3196	0	0	0	1	0	1
12	1578	1	0	0	1	0	1

02 - Write a query and table creation statement to make final_assignments_qa
look like the final_assignments table. If you discovered something missing in part 1,
you may fill in the value with a place holder of the appropriate data type.

ANSWER: The code below combines data from multiple columns ('test_a', 'test_b',....)
into a single result set, where each row represents a unique item assignment for a test,
along with additional information about the test assignment, test_number, and test start date.

```
SELECT item_id,
       test_a AS test_assignment,
       (CASE
            WHEN test_a IS NOT NULL then 'test_a'
            ELSE NULL
        END) AS test_number,
       (CASE
            WHEN test_a IS NOT NULL then '2013-01-05 00:00:00'
            ELSE NULL
        END) AS test_start_date
FROM dsv1069.final_assignments_qa
UNION
SELECT item_id,
       test_b AS test_assignment,
       (CASE
            WHEN test_b IS NOT NULL then 'test_b'
            ELSE NULL
        END) AS test_number,
       (CASE
            WHEN test_b IS NOT NULL then '2013-01-05 00:00:00'
            ELSE NULL
        END) AS test_start_date
FROM dsv1069.final_assignments_qa
UNION
SELECT item_id,
       test_c AS test_assignment,
       (CASE
            WHEN test_c IS NOT NULL then 'test_c'
            ELSE NULL
        END) AS test_number,
       (CASE
            WHEN test_c IS NOT NULL then '2013-01-05 00:00:00'
            ELSE NULL
        END) AS test_start_date
FROM dsv1069.final_assignments_qa
UNION
SELECT item_id,
       test_d AS test_assignment,
       (CASE
            WHEN test_d IS NOT NULL then 'test_d'
            ELSE NULL
        END) AS test_number,
       (CASE
            WHEN test_d IS NOT NULL then '2013-01-05 00:00:00'
            ELSE NULL
        END) AS test_start_date
FROM dsv1069.final_assignments_qa
UNION
SELECT item_id,
       test_e AS test_assignment,
       (CASE
            WHEN test_e IS NOT NULL then 'test_e'
            ELSE NULL
        END) AS test_number,
       (CASE
            WHEN test_e IS NOT NULL then '2013-01-05 00:00:00'
            ELSE NULL
        END) AS test_start_date
FROM dsv1069.final_assignments_qa
UNION
SELECT item_id,
       test_f AS test_assignment,
       (CASE
            WHEN test_f IS NOT NULL then 'test_f'
            ELSE NULL
        END) AS test_number,
       (CASE
            WHEN test_f IS NOT NULL then '2013-01-05 00:00:00'
            ELSE NULL
        END) AS test_start_date
FROM dsv1069.final_assignments_qa;
```

02

item_id COUNT DISTINCT: 57	test_assignment SUM: 43 AVG: 0.45 MIN: 0 MAX: 1	test_number COUNT DISTINCT: 6	test_start_date COUNT DISTINCT: 1
3462	1	test_c	2013-01-05 00:00:00
3514	1	test_d	2013-01-05 00:00:00
3540	1	test_e	2013-01-05 00:00:00
7	0	test_a	2013-01-05 00:00:00
354	0	test_c	2013-01-05 00:00:00
1965	1	test_f	2013-01-05 00:00:00
1056	1	test_d	2013-01-05 00:00:00
2133	0	test_a	2013-01-05 00:00:00

Page 1 of 1 Showing rows: 1-100 of 100

03 - Use the final_assignments table to calculate the order binary for the
30 day window after the test assignment for item_test_2
(You may include the day the test started)

ANSWER: The code generates a summary of metrics related to item assignments
for the 'item_test_2' test, including the number of items assigned to each
test and the number of items ordered within 30 days of the test start date.

```
SELECT test_assignment,
    COUNT(DISTINCT item_id) AS number_of_items,
    SUM(order_binary) AS items_ordered_30d
FROM
 (SELECT item_test_2.item_id,
         item_test_2.test_assignment,
         item_test_2.test_number,
         item_test_2.test_start_date,
         item_test_2.created_at,
         MAX(CASE
                 WHEN (created_at > test_start_date
                       AND DATE_PART('day', created_at - test_start_date) <= 30) THEN 1
                 ELSE 0
             END) AS order_binary
  FROM
    (SELECT final_assignments.*,
            DATE(orders.created_at) AS created_at
     FROM dsv1069.final_assignments AS final_assignments
     LEFT JOIN dsv1069.orders AS orders
       ON final_assignments.item_id = orders.item_id
       WHERE test_number = 'item_test_2') AS item_test_2
  GROUP BY item_test_2.item_id,
           item_test_2.test_assignment,
           item_test_2.test_number,
           item_test_2.test_start_date,
           item_test_2.created_at) AS order_binary
GROUP BY test_assignment;
```

03

test_assignment SUM: 1 AVG: 0.50 MIN: 0 MAX: 1	number_of_items SUM: 2198 AVG: 1099 MIN: 1068 MAX: 1130	items_ordered_30d SUM: 749 AVG: 374.50 MIN: 363 MAX: 386
0	1130	386
1	1068	363

Page 1 of 1 Showing rows: 1-2 of 2

04 - Use the final_assignments table to calculate the view binary,
and average views for the 30 day window after the test assignment for item_test_2.
(You may include the day the test started)

```
SELECT
test_assignment,
COUNT(item_id) AS items,
SUM(view_binary_30d) AS viewed_items,
CAST(100*SUM(view_binary_30d)/COUNT(item_id) AS FLOAT) AS viewed_percent,
SUM(views) AS views,
SUM(views)/COUNT(item_id) AS average_views_per_item
FROM
(
 SELECT
  f.test_assignment,
  f.item_id,
  MAX(CASE WHEN item_views.event_time > f.test_start_date THEN 1 ELSE 0 END)  AS view_binary_30d,
  COUNT(item_views.event_id) AS views
 FROM
  dsv1069.final_assignments f
 LEFT OUTER JOIN
  (
   SELECT
    event_time,
    event_id,
    CAST(parameter_value AS INT) AS item_id
   FROM
    dsv1069.events
   WHERE
    event_name = 'view_item'
   AND
    parameter_name = 'item_id'
  ) item_views
 ON
  f.item_id = item_views.item_id
 AND
  item_views.event_time >= f.test_start_date
 AND
  DATE_PART('day', item_views.event_time - f.test_start_date ) <= 30
 WHERE
  f.test_number= 'item_test_2'
 GROUP BY
  f.test_assignment,
  f.item_id
) item_orders
GROUP BY
 test_assignment
```

04

test_assignment SUM: 1 AVG: 0.50 MIN: 0 MAX: 1	items SUM: 2198 AVG: 1099 MIN: 1068 MAX: 1130	viewed_items SUM: 1808 AVG: 904 MIN: 890 MAX: 918	viewed_percent SUM: 164 AVG: 82 MIN: 81 MAX: 83	views SUM: 3778 AVG: 1889 MIN: 1862 MAX: 1916	average_views_per_item SUM: 3.44 AVG: 1.72 MIN: 1.74 MAX: 1.74
0	1130	918	81	1916	1.6956
1	1068	890	83	1862	1.7434

Page 1 of 1 Showing rows: 1-2 of 2

05 - Use the https://thumbtack.github.io/abba/demo/abba.html
to compute the lifts in metrics and the p-values for the binary metrics
(30 day order binary and 30 day view binary) using a interval 95% confidence.

05

test_assignment SUM: 1 AVG: 0.50 MIN: 0 MAX: 1	number_of_items SUM: 2198 AVG: 1099 MIN: 1068 MAX: 1130	view_binary_30d SUM: 1819 AVG: 909.50 MIN: 894 MAX: 925
0	1130	925
1	1068	894

Page 1 of 1 Showing rows: 1-2 of 2