# Analyzing Results

Exercise 1: Use the order_binary metric from the previous exercise, count the number of users per treatment group for test_id = 7, and count the number of users with orders (for test_id 7).

```sql
select
  test_assignment,
  count(user_id)    as users,
  sum(order_binary) as order_binary
from
  (
  select
    assignments.user_id,
    assignments.test_id,
    assignments.test_assignment,
    max(case when orders.created_at > assignments.event_time then 1 else 0 end)  as order_binary

  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_big
    where
      event_name  = 'test_assignment'
    group by
      event_id,
      event_time,
      user_id
    order by event_id
    ) assignments
  left outer join
    dsv1069.orders_big orders
  on
    assignments.user_id = orders.user_id
  group by
    assignments.user_id,
    assignments.test_id,
    assignments.test_assignment
  ) user_level
where
  test_id = 7
group by
  test_assignment
```

Exercise 2: Create a new tem view binary metric. Count the number of users per treatment group, and count the number of users with views (for test_id 7).

```sql
select
  test_assignment,
  count(user_id)    as users,
  sum(view_binary) as views
from
```

```sql
(
  select
    assignments.user_id,
    assignments.test_id,
    assignments.test_assignment,
    max(case when views.event_time > assignments.event_time then 1 else 0 end)  as view_binary
  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_with_testing
    where
      event_name  = 'test_assignment'
    group by
      event_id,
      event_time,
      user_id
    order by event_id
    ) assignments
  left outer join
    (
    select *
    from
      dsv1069.events
    where
      event_name = 'view_item'
    and
      parameter_name = 'item_id'
    ) views
  on
    assignments.user_id = views.user_id

  group by
    assignments.user_id,
    assignments.test_id,
    assignments.test_assignment
) user_level
where
  test_id = 7
group by
  test_assignment
```

Exercise 3: Alter the result from EX 2, to compute the users who viewed an item WITHIN 30 days of their treatment event.

```sql
select
  test_assignment,
  count(user_id)    as users,
  sum(view_binary) as views,
  sum(view_binary_30d) as views_30d
from
(
  select
    assignments.user_id,
    assignments.test_id,
    assignments.test_assignment,
    max(case when views.event_time > assignments.event_time then 1 else 0 end)  as view_binary,

    max(case when (views.event_time > assignments.event_time
                  and date_part('day', views.event_time - assignments.event_time ) <= 30)
             then 1 else 0 end)  as view_binary_30d
  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_with_testing
    where
      event_name  = 'test_assignment'
    group by
      event_id,
      event_time,
      user_id
    order by event_id
    ) assignments
  left outer join
    (
    select *
    from
      dsv1069.events
    where
      event_name = 'view_item'
    and
      parameter_name = 'item_id'
    ) views
  on
    assignments.user_id = views.user_id
  group by
    assignments.user_id,
    assignments.test_id,
    assignments.test_assignment
) user_level
where
  test_id = 7
group by
  test_assignment
```

Exercise 4: Create the metric invoices (this is a mean metric, not a binary metric) and for test_id = 7

----The count of users per treatment group

----The average value of the metric per treatment group

----The standard deviation of the metric per treatment group

```
select
  test_id,
  test_assignment,
  count(user_id) as users,
  avg(total_revenue)  as metric_average,
  stddev(total_revenue) as metric_stddev
from
  (
select
```

```sql
  assignments.user_id,
  assignments.test_id,
  assignments.test_assignment,
  count(distinct case when orders.created_at > assignments.event_time then orders.invoice_id else null end)
    as invoices,
  count(distinct case when orders.created_at > assignments.event_time then orders.line_item_id else null end)
    as line_items,
  coalesce(sum(case when orders.created_at > assignments.event_time then orders.price else 0 end), 0)
    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
  order by event_id
  ) assignments
left outer join
  dsv1069.orders
on
  assignments.user_id = orders.user_id
group by
  assignments.user_id,
  assignments.test_id,
  assignments.test_assignment
  ) user_level
where test_id = 6
group by
  test_id,
  test_assignment
limit 100
```