Error Codes

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 when 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:
Goal: Here we use *users* table to pull a list of user email addresses. Edit the query to pull email addresses, but only for non-deleted users.

Starter Code:

```
SELECT *
FROM dsv1069.users
```

```
SELECT id AS user_id,
email_address
FROM dsv1069.users
WHERE deleted_at is NULL
```

Exercise 2:
--Goal: Use the *items* table to count the number of items for sale in each category
Starter Code: (none)

```
SELECT category,
count(id) as item_count
FROM dsv1069.items
GROUP BY category
ORDER BY item_count DESC
```

Exercise 3:
--Goal: Select all of the columns from the result when you JOIN the *users* table to the *orders* table

Starter Code: (none)

```
SELECT *
FROM dsv1069.users  JOIN dsv1069.orders on  user_id = id
```

Exercise 4:
--Goal: Check out the query below. This is not the right way to count the number of viewed_item events. Determine what is wrong and correct the error.

Starter Code:

```
SELECT
```

```
SELECT COUNT(distinct event_id)
FROM dsv1069.events
WHERE event_name = 'view_item'
```

COUNT(event_id) AS events
FROM dsv1069.events
WHERE event_name = 'view_item'

Exercise 5:

--Goal:Compute the number of items in the items table which have been ordered. The query below runs, but it isn't right. Determine what is wrong and correct the error or start from scratch.
Starter Code:

```
SELECT
  COUNT(item_id) as item_count
FROM dsv1069.orders
INNER JOIN dsv1069.items
ON orders.item_id = items.id
```

```
SELECT COUNT(DISTINCT orders.item_id) as
item_count
FROM dsv1069.orders
JOIN dsv1069.items
ON  items.id = id
```

--Error: This query runs but the number isn't right

Exercise 6:

--Goal: For each user figure out IF a user has ordered something, and when their first purchase was. The query below doesn't return info for any of the users who haven't ordered anything.

Starter Code:

```
SELECT
  users.id  AS user_id,
  MIN(orders.paid_at)  AS min_paid_at
FROM
  dsv1069.orders
INNER JOIN
  dsv1069.users
ON
  orders.user_id = users.id
GROUP BY
  users.id
```

```
SELECT users.id as user_id,
MIN(orders.paid_at) as min_paid_at
FROM dsv1069.users
LEFT OUTER join
dsv1069.orders
on orders.user_id = users.id
GROUP BY users.id
```

Exercise 7:

--Goal: Figure out what percent of users have ever viewed the user profile page, but this query isn't right. Check to make sure the number of users adds up, and if not, fix the query.
Starter Code:

```sql
SELECT
(CASE WHEN first_view IS NULL THEN false
     ELSE true END) AS has_viewed_profile_page,
COUNT(user_id) as users
FROM
  (SELECT
    users.id  AS user_id,
    MIN(event_time) as first_view
  FROM
    dsv1069.users
  LEFT OUTER JOIN
    dsv1069.events
  ON
    events.user_id = users.id
  WHERE
    event_name = 'view_user_profile'
  GROUP BY
    users.id
  ) first_profile_views
GROUP BY
  (CASE WHEN first_view IS NULL THEN false
     ELSE true END)
```

```sql
SELECT
  ( CASE WHEN first_view IS NULL THEN false
    ELSE True END) AS has_viewed_profile_page
  , COUNT(id)
FROM
  (SELECT
  id
  , MIN(event_time) AS FIRST_VIEW
  FROM dsv1069.users u LEFT OUTER JOIN dsv1069.events e
  ON
  e.user_id = u.id
  AND
  e.event_name = 'view_user_profile'
  GROUP BY
  id
  )first_profile_view
  GROUP BY
  (CASE WHEN First_view IS NULL THEN FALSE
  ELSE TRUE END)
```