Module 4 Lab

For this module we will be using the `completejourney` data sets. However, rather than use the sample transaction data (`transactions_sample`) we will be using the entire transaction data set provided by `get_transactions()` and the entire promotions data set provided by `get_promotions()`.

```
library(tidyverse)
library(lubridate)
library(completejourney)

transactions <- get_transactions()
dim(transactions)
## [1] 1469307      11

promotions <- get_promotions()
dim(promotions)
## [1] 20940529        5
```

1. Fill in the blanks with the correct join operations to answer the following questions. Using the `transactions` and `demographics` data, how many of the 1,469,307 transactions do we have demographic information for? How many of the transactions do we not have demographic information on?

```
# how many transaction do we have demographics on?
transactions %>%
   _____join(demographics, by = "household_id") %>%
   tally()

# how many transaction do we NOT have demographics on?
transactions %>%
   _____join(demographics, by = "household_id") %>%
   tally()
```

2. Fill in the blanks to perform an inner join with the `transactions` and `demographics` data. Then, compute the total `sales_value` by `age` category to identify which age group generates the most sales.

```
transactions %>%
   _____join(demographics, by = "household_id") %>%
   group_by(_____) %>%
   summarise(total_sales = _______) %>%
   arrange(desc(total_sales))
```

3. Identify all households that have total sales (`sales_value`) of $1000 or more. To do this, fill in the blanks to compute total sales by household ID and then filter for those household IDs that have `total_sales` equal to or greater than $1000.

```
# Identify households with $1000 or more in total sales
hshld_1000 <- transactions %>%
   group_by(______) %>%
   ________(total_sales = ____(sales_value, na.rm = TRUE)) %>%
   ________(total_sales >= 1000)
```

Now, join the above results with the demographics data to determine:

- How many of these households do we have demographic data on?
- How many do we not have demographic on?
- For those that we do have demographics on, which `income` range produces the most households that spend $1000 or more?

```
# How many of these households do we have demographic data on?
hshld_1000 %>%
   _____join(demographics, by = "household_id") %>%
   tally()

# How many do we not have demographic on?
hshld_1000 %>%
   _____join(demographics, by = "household_id") %>%
   tally()

# Which income range produces the most households that spend \$1000 or more?
hshld_1000 %>%
   inner_join(demographics, by = _______) %>%
   ________
```

4. Using the `promotions` and `transactions` data, compute the total sales for **all** products that were in a display in the front of the store (`display_location = 1`).

```
# join transactions and filtered promotions data
front_display_trans <- promotions %>%
   filter(__________) %>%
   inner_join(transactions, by = c('product_id', 'store_id', 'week'))

# total sales for all products displayed in the front of the store
front_display_trans %>%
   summarize(total_sales = __________)
```

Now compute the total sales for each product (`product_id`) displayed in the front of the store and identify the `product_id` that had the largest total sales.

```
# Identify the product displayed in the front of the store that had the
# largest total sales
front_display_trans %>%
   group_by(__________) %>%
   summarize(total_front_display_sales = __________) %>%
   __________
```

5. Fill in the blanks to identify which `product_category` is related to the coupon where `campaign_id` is equal to 18 and `coupon_upc` is equal to 10000089238?

```
coupons %>%
   _______(campaign_id == ____, coupon_upc == _____) %>%
   inner_join(products, by = "product_id")
```

6. Identify all different products that contain "pizza" in their `product_type` description. Which of these products produces the greatest amount of total sales (compute total sales by product ID and product type)?

```
# test your ability to right this code from scratch rather than just
# filling in the blanks :)
-------
```

7. Fill in the blanks to identify all products that are categorized (`product_category`) as "pizza" but are considered a "snack" or "appetizer" (via `product_type`). **Hint**: the simplest way to do this is to filter first for pizza products and then second for products that are snacks or appetizers.

```
relevant_products <- products %>%
   filter(
      str_detect(product_category, regex(_______, ignore_case = TRUE)),
      str_detect(product_type, regex(_______, ignore_case = TRUE))
   )
```

Now fill in the blanks to join the above relevant pizza products with the transactions data, compute the total quantity of items sold by product ID. Which of these products (`product_id`) have the most number of sales (which we are measuring by total quantity)?

```
relevant_products %>%
   inner_join(transactions, by = 'product_id') %>%
   __________ %>%
   __________(total_qty = sum(quantity)) %>%
   arrange(desc(_______))
```

8. Identify all products that contain "peanut butter" in their `product_type`. How many unique products does this result in?

```
pb <- products %>%
   filter(_______(product_type, regex(_______, ignore_case = TRUE)))

tally(pb)
```

For these products, compute the total `sales_value` by month based on the `transaction_timestamp`. Which month produces the most sales value for these products? Which month produces the least sales value for these products?

```
pb %>%
   inner_join(transactions, by = "product_id") %>%
   group_by(month = month(_________, label = TRUE)) %>%
   summarize(total_sales = _______) %>%
   arrange(desc(_______))
```

9. Using the `coupon_redemptions` data, filter for the coupon associated with `campaign_id` 18 and `coupon_upc` "10000085475". How many households redeemed this coupon? Now, using this coupon, identify the total `sales_value` for all transactions associated with the `household_id`s that redeemed this coupon on the same day they redeemed the coupon. To do this you will want to:

    a. filter `coupon_redemptions` data for `campaign_id` == "18" and `coupon_upc` == "10000085475",
    b. join with the transactions data so that you only include households that redeemed the coupon,
    c. filter for those transactions where the `redemption_date` was made on the same day as the `transaction_timestamp` (hint: `yday()`), and
    d. then compute the total `sales_value` across all these transactions.

```
# test your ability to right this code from scratch rather than just
# filling in the blanks :)
-------
```

10. Let's build onto #9. Using the same redeemed coupon (campaign_id == "18" & coupon_upc == "10000085475"). In this problem we are going to calculate the total `sales_value` for each `product_type` that this coupon was applied to so that we can identify which `product_type` resulted in the greatest sales when associated with this coupon.

To do this you will want to:

a. filter `coupon_redemptions` data for `campaign_id` == "18" and `coupon_upc` == "10000085475",
b. perform an inner join this with the `coupons` data so that we only retain the coupon information for the relevant coupon. This step will provide us with the necessary `product_id` information so we can link the coupon to the products purchased.
c. perform an inner join with the `products` data so that we can get the product information for each product associated with the redeemed coupons.
d. Filter for only those products where "vegetables" is in the 'product_category' description.
e. Now perform an inner join with the transactions data using the `household_id` and `product_id` keys.
f. Filter the data so that the day of year of the `redemption_date` is equal to the day of year of the `transaction_timestamp` (hint: `yday()`).
g. Now you can group by `product_type`,
h. compute the total `sales_value`, and
i. arrange the data to identify the `product_type` with the largest total sales value.

```
# test your ability to right this code from scratch rather than just
# filling in the blanks :)
-------
```