Module 3 Lab

Part 1

All of the Part 1 exercises can be solved using the `tidyverse` and `completejourney` packages. The `completejourney` package is an R data package that has been created so the full suite of Complete Journey datasets can be loaded as a library. You can find details about the data and the variables at http://bit.ly/completejourney. In order to use the data you must first install the package following these steps:

```
install.packages('completejourney')
```

Go ahead and load the `tidyverse` and `completejourney` packages:

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

The exercises that follow will use various data sets included in the `completejourney` package to include:

```
transactions <- transactions_sample  # just using a sample of the entire data
transactions
## # A tibble: 75,000 x 11
##    household_id store_id basket_id   product_id quantity sales_value retail_disc
##    <chr>        <chr>    <chr>       <chr>         <dbl>       <dbl>       <dbl>
##  1 2261         309      31625220889 940996            1        3.86        0.43
##  2 2131         368      32053127496 873902            1        1.59        0.9
##  3 511          316      32445856036 847901            1        1           0.69
##  4 400          388      31932241118 13094913          2       11.9         2.9
##  5 918          340      32074655895 1085604           1        1.29        0
##  6 718          324      32614612029 883203            1        2.5         0.49
##  7 868          323      32074722463 9884484           1        3.49        0
##  8 1688         450      34850403304 1028715           1        2           1.79
##  9 467          31782    31280745102 896613            2        6.55        4.44
## 10 1947         32004    32744181707 978497            1        3.99        0
## # i 74,990 more rows
## # i 4 more variables: coupon_disc <dbl>, coupon_match_disc <dbl>, week <int>,
## #   transaction_timestamp <dttm>

products
## # A tibble: 92,331 x 7
##    product_id manufacturer_id department    brand  product_category product_type
##    <chr>      <chr>           <chr>         <fct>  <chr>            <chr>
##  1 25671      2               GROCERY       Natio~ FRZN ICE         ICE - CRUSH~
##  2 26081      2               MISCELLANEOUS Natio~ <NA>             <NA>
##  3 26093      69              PASTRY        Priva~ BREAD            BREAD:ITALI~
##  4 26190      69              GROCERY       Priva~ FRUIT - SHELF S~ APPLE SAUCE
```

```
##  5 26355      69              GROCERY       Priva~ COOKIES/CONES    SPECIALTY C~
##  6 26426      69              GROCERY       Priva~ SPICES & EXTRAC~ SPICES & SE~
##  7 26540      69              GROCERY       Priva~ COOKIES/CONES    TRAY PACK/C~
##  8 26601      69              DRUG GM       Priva~ VITAMINS         VITAMIN - M~
##  9 26636      69              PASTRY        Priva~ BREAKFAST SWEETS SW GDS: SW ~
## 10 26691      16              GROCERY       Priva~ PNT BTR/JELLY/J~ HONEY
## # i 92,321 more rows
## # i 1 more variable: package_size <chr>
```

Exercise 1

Fill in the blanks to create three new variables named `regular_price`, `loyalty_price`, and `coupon_price` according to the following logic:

```
regular_price = (sales_value + retail_disc + coupon_match_disc) / quantity
loyalty_price = (sales_value + coupon_match_disc) / quantity
coupon_price  = (sales_value - coupon_disc) / quantity
```

. . . and then perform the following analyses:

1. Identify the five households with the largest `loyalty_price` transactions. What is unique about the transaction with the largest `loyalty_price` value?
2. Now filter for only those observations where quantity was greater than 0. Now which household(s) have the largest `loyalty_price` transaction?
3. Using the first transaction in the result from #2, filter the `products` data based on the `product_id` to find out which product the largest `loyalty_price` transaction is associated with.

```
# Q0: Create three new variables named `regular_price`, `loyalty_price`, and
#     `coupon_price` according to the logic shown above

transactions <- transactions %>%
   mutate(
     regular_price     = ____________,
     loyalty_price     = ____________,
     coupon_price      = ____________
   ) %>%
   select(regular_price, loyalty_price, coupon_price, product_id, everything())

# Q1. Identify the five households with the largest `loyalty_price` transactions. What
#     is unique about the transaction with the largest `loyalty_price` value?
transactions %>%
   slice_max(order_by = loyalty_price, n = __)
```

```
# Q2. Now filter for only those observations where quantity was greater than 0. Now which
#     household(s) have the largest `loyalty_price` transaction?
transactions %>%
   filter(quantity __ 0) %>%
   slice_max(order_by = ______, n = __)
```

```
# Q3. Using the first transaction in the result from #2, filter the `products` data based
#     on the `product_id` to find out which product the largest `loyalty_price` transaction
#     is associated with.
products %>%
   filter(product_id == _______)
```

Exercise 2

`transactions` includes 20,902 unique product IDs. How many of these products (not transactions!) had a regular price of one dollar or less? How many of these products had a loyalty price of one dollar or less? How about a coupon price of one dollar or less?

Hint: After filtering, select the `product_id` column and count unique products using the `n_distinct()` function.

```
# how many products had a regular price of $1 or less
transactions %>%
  filter(regular_price <= __) %>%
  select(product_id) %>%
  n_distinct()

# how many products had a loyalty price of $1 or less
transactions %>%
  filter(_________) %>%
  select(product_id) %>%
  ---------

# how many products had a coupon price of $1 or less
transactions %>%
  filter(_________) %>%
  --------- %>%
  ---------
```

Exercise 3

What proportion of baskets are over $10 in sales value? What proportion of baskets are over $20 in sales value?

Hint: You need to use `group_by()` and `summarize()`. Depending on your approach you may or may not use `mutate()`.

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

Exercise 4

Which stores had the largest total `sales_value` (hint: `sum(sales_value, na.rm = TRUE)`? Which stores had the largest average loyalty discount as defined below?

Hint: You can calculate loyalty discount as a percentage of regular price using the following logic:

```
pct_loyalty_disc = 1 - (loyalty_price / regular_price)
```

```
# Which stores had the largest total `sales_value`
transactions %>%
   group_by(_____) %>%
   summarize(total_sales_value = __________) %>%
   arrange(desc(total_sales_value))
```

```
# Which stores had the largest average loyalty discount
transactions %>%
   mutate(pct_loyalty_disc = _______________) %>%
   group_by(_______) %>%
   summarize(avg_pct_loyalty_disc = mean(________, na.rm = TRUE)) %>%
   arrange(desc(avg_pct_loyalty_disc))
```

Part 2

For this part of the lab we'll work through the mbta.xlsx data. The Massachusetts Bay Transportation Authority ("MBTA") manages America's oldest subway, as well as Greater Boston's commuter rail, ferry, and bus systems. It's your first day on the job as the T's data analyst and you've been tasked with analyzing average ridership through time. Complete the following data cleaning tasks and be sure to use proper code styling and commenting throughout your notebook.

Exercise 1 - *Import the data*

1. What spreadsheets exist in the workbook?
2. Import mbta.xlsx[1]
3. When importing the data, interpret the string 'NA' as missing values. You may need to look at the help docs of `read_excel()`.

```
library(readxl)

excel_sheets(path = _________)
mbta <- read_excel(path = _______, skip = __, na = __)
```

Exercise 2 - *Exam the data*

The first step when cleaning a data set is to explore it a bit. Pay particular attention to how the rows and columns are organized and to the locations of missing values.

1. View the structure of mbta.
2. View the first 6 rows of mbta.
3. View a summary of mbta.
4. How many missing values are in each column? If you see zero that means you didn't complete exercise 1.3 correctly.

[1] You may need to skip a row or two.

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

Exercise 3 - *Removing unnecessary rows and columns*

It appears that the data are organized with observations stored as columns rather than as rows. You can fix that.

First, though, you can address the missing data. All of the `NA` values are stored in the first row. This row really belongs in a different data frame; it is a quarterly average of weekday MBTA ridership. Since this data set tracks monthly average ridership, you'll remove that row. Similarly, the 7th row (Pct Chg / Yr) and the 11th row (TOTAL) are not really observations as much as they are analysis. Go ahead and remove the 7th and 11th rows as well. The first column also needs to be removed because it's just listing the row numbers.

1. Remove the first, seventh, and eleventh rows of mbta.
2. Remove the first column.
3. Now what is the dimensions of this new data frame?

```
mbta <- mbta %>%
   slice(-c(______)) %>%  # Remove the first, seventh, and eleventh rows of mbta.
   select(__)             # Remove the first column.

____(mbta)                # Now what is the dimensions of this new data frame?
```

Exercise 4 - *Observations are stored in columns*

In this data, variables are stored in rows instead of columns. The different modes of transportation (commuter rail, bus, subway, boat, ...) are variables, providing information about each month's average ridership. The months themselves are observations.[2] You can tell which is which because as you go through time, the month changes, but the modes of transport offered by the T do not.

As is customary, you want to represent variables in columns rather than rows.

1. Pivot the rows and columns of the mbta data so that all columns are variables of the data. This should result in 3 columns - `mode`, `date`, and number of riders in thousands (`thou_riders`).

2. What are the new dimensions of this data?

```
mbta <- mbta %>%
   pivot_longer(__________)   # Pivot the rows and columns of the mbta data so that all
                              # columns are variables of the data. This should result in
                              # 3 columns - `mode`, `date`, and number of riders in
                              # thousands (`thou_riders`).

____(mbta)                    # Now what is the dimensions of this new data frame?
```

[2] Currently, months are listed as variable names; rather, they should be in their own column.

Exercise 5 - *Separating columns*

Your data set is already looking much better! Your boss saw what a great job you're doing and now wants you to do an analysis of the T's ridership during certain months across all years.

Your data set has months in it, so that analysis will be a piece of cake. There's only one small problem: if you want to look at ridership on the T during every January (for example), the month and year are together in the same column, which makes it a little tricky. You'll need to separate the month column into distinct month and year columns to make life easier.

1. Split the month column of mbta at the dash and create a new month column with only the month and a year column with only the year.
2. View the head of this new mbta data set.

```
mbta <- mbta %>%
   separate(__________)    # Split the month column of mbta at the dash and create a new
                           # month column with only the month and a year column with only
                           # the year.

____(mbta)                 #  View the head of this new mbta data set.
```

Exercise 6 - *Dealing with entry error*

Every month, average weekday commuter boat ridership was around 4,000. Then, one month it jumped to 40,000 without warning? Unless the Olympics were happening in Boston that month (they weren't), this value is certainly an error. You can assume that whoever was entering the data that month accidentally typed 40 instead of 4.

1. Locate the row and column of the incorrect value.
2. Replace the incorrect value with 4.

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

Congrats, your data is now clean and ready for analysis.

Exercise 7 - *Performing descriptive analysis*

1. Compute the average ridership per mode.

```
mbta %>%
   group_by(_______) %>%
   summarize(avg_ridership = _______)
```

2. Compute the average ridership per mode for the month of January.

```
mbta %>%
   filter(month == _____) %>%
   group_by(_______) %>%
   summarize(avg_ridership = _______)
```

3. Which year had the largest total ridership for the boat mode?

```
mbta %>%
   filter(_______) %>%
   group_by(year) %>%
   summarize(total_riders = ____(thou_riders))
```

4. On average, which month experiences the greatest number of passengers on the Heavy Rail mode?

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