---

title: "forage-quantium"

Quarto

```{r setup}

library(tidyverse)

library(skimr)

```

```{r read-in-files}

dfile = list.files('data', full.names = T)

cust_info_raw = read_csv(dfile[1])

prod_info_raw = readxl::read_xlsx(dfile[2])

```

There is a lot of information kept in one column, the PROD_NAME column. I split that column into three columns to disentangle the pack_size and the prod_brand from the prod_name. Now we can see what sizes of packs are included and how many brands are in the dataset.

```{r data-cleaning}

cust_info=
```

```
 cust_info_raw %>%
 janitor::clean_names() %>%
 mutate(across(lylty_card_nbr:premium_customer, factor))


prod_info=
 prod_info_raw %>%
 janitor::clean_names() %>%
 mutate(date = janitor::excel_numeric_to_date(date)) %>%
 mutate(prod_name = str_remove_all(prod_name, "[[:punct:]]"),
      pack_size = parse_number(prod_name),
      prod_name = str_remove(prod_name, "\\d{1,3}g|\\d{1,3}G")) %>%
 separate_wider_delim(prod_name, " ",
                 names = c("prod_brand","prod_flavor"),
                 too_many = 'merge',
                 cols_remove = F) %>%
 mutate(prod_brand =
       case_when(prod_brand=="WW" ~ "Woolworths",
              prod_brand=="Dorito" ~ "Doritos",
              prod_brand=="Snbts" ~ "Sunbites",
              prod_brand=="Smith" ~ "Smiths",
              prod_brand=="Infzns" ~ "Infuzions",
              prod_brand=="Old" ~ "Old El Paso",
              prod_brand=="Natural" ~ "Natural Chip Co",
              prod_brand=="National" ~ "National Chip Co",
              prod_brand %in% c("GrnWves","Grain") ~ "Grain Waves",
              TRUE ~ prod_brand),
```

```
    prod_name = str_squish(prod_name),
    across(contains("_nbr"), as.factor),
    across(contains("_id"), as.factor))
```

Some of the substitutions I made need to be checked with the customer. For example: is "NCC" the same as "Natural Chip Co"? I left them as a seperate category but this could be better handled with more information.

The **skimr** package in R will allow a quick overview of the completeness of the data and other high-level summaries of the data type and distribution.

```{r skim}
skim(prod_info)
```

We can tell from the tables above that there is a date column ranging from July 2018 to June 2019, three character variables (including the ones I added) with 23 unique brands and 114 unique products, and three numeric variables. There is an average of 2 packs in each product purchase (rounded up) and the average cost of a purchase is $7.30. The average pack size is around 180g. There are also four categorical variables (factors) including a tax id number which is unique for each purchase, a store number, a loyalty card number, and a product number. The loyalty card number will be useful for cross-referencing the cust_info dataframe that I created from the purchase behaviour dataset that was provided.

Below is a histogram of the pack size by brand to show the distribution of pack sizes.

```{r pack-hist}
prod_info %>%
 ggplot() +
 geom_histogram(aes(x = pack_size, fill = prod_brand)) +
```

```
 theme_minimal() +
 labs(title = "Pack size frequency by brand",
     subtitle = "Most pack sizes are around 150g",
     x = "Pack size",
     y = NULL,
     fill = "Brand name") +
 scale_y_continuous(labels = scales::number_format(big.mark = ","))+
 scale_x_continuous(labels = scales::number_format(suffix = "g"),
                    breaks = seq(100,400,50))
```

Now for the cust_info. I will pass this to **skimr** like before.

```{r skim2}
skim(cust_info)
```

All three variables of this dataset are categorical, though there are many loyalty card numbers. There are 7 categories of lifestage and 3 premium customer categories. Once again, the data is complete (no missing observations).

One quick way to make this dataset more useful is to join the two tables together by loyalty number to prepare for analysis on customer category and lifestage for each purchase. The join is performed in the code below. We pass to **skimr** again just to make sure the join did not produce any NAs.

```{r join}
full_df =
```

```
  prod_info %>%
  left_join(cust_info,
            by = join_by(lylty_card_nbr))


skim(full_df)
```