Order Now

Data Mining

Evan, Robby, and Tyler

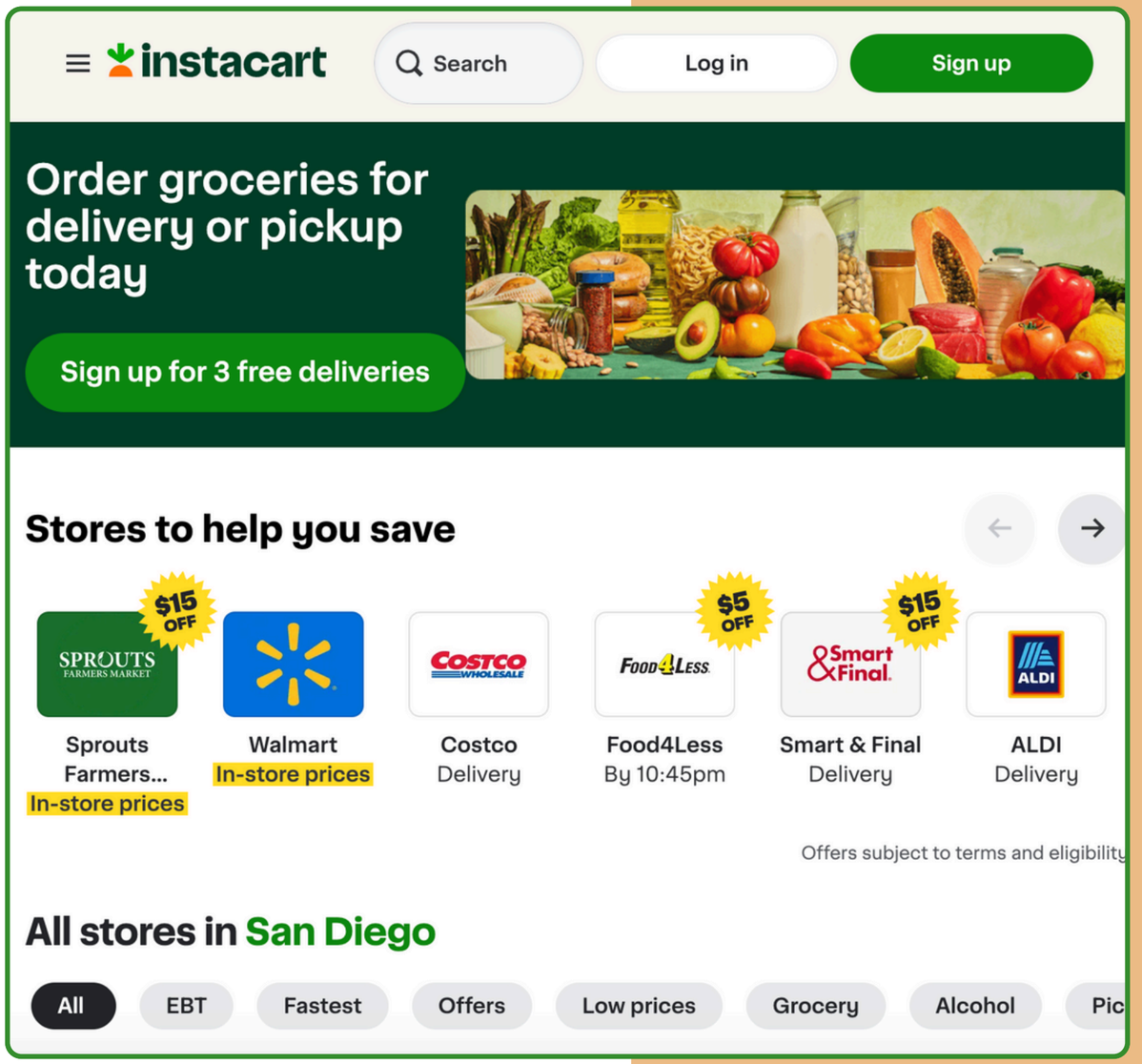
let’s go.

Outline

1. The Dataset
2. Visualizations
3. Queries

Instacart

- **Grocery Delivery**: Shop online and get groceries delivered same-day or schedule pickup.
- **Local Stores**: Access items from multiple nearby stores in one order.
- **Personalized Shopping**: Real-time updates and substitutions via the app.



EXECUTIVE SUMMARY

Objective

- Gain insights based on users' orders

Approach

- Analyzing for correlations and patterns

Tools

- Using ggplot2 and SQLDF in RStudio

Metric	Value
Orders	3214874.00
Days Between Orders (Avg)	11.11
Customers	206209.00
Order Size (Avg)	10.55
Products	49688.00
Departments	21.00
Aisles	134.00

Tables

	department_id	department
1	1	frozen
2	2	other
3	3	bakery
4	4	produce
5	5	alcohol
6	6	international
7	7	beverages
8	8	pets
9	9	dry goods pasta
10	10	bulk
11	11	personal care
12	12	meat seafood
13	13	pantry
14	14	breakfast
15	15	canned goods
16	16	dairy eggs
17	17	household
18	18	babies
19	19	snacks
20	20	deli
21	21	missing

	order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order
1	2539329	1	prior	1	2	8	*NA*
2	2398795	1	prior	2	3	7	15
3	473747	1	prior	3	3	12	21

	aisle_id	aisle
1	1	prepared soups salads
2	2	specialty cheeses
3	3	energy granola bars
4	4	instant foods
5	5	marinades meat preparation
6	6	other
7	7	packaged meat
8	8	bakery desserts
9	9	pasta sauce

	product_id	product_name	aisle_id	department_id
1	1	Chocolate Sandwich Cookies	61	19
2	2	All-Seasons Salt	104	13
3	3	Robust Golden Unsweetened Oolong Tea	94	7
4	4	Smart Ones Classic Favorites Mini Rigatoni With Vodk…	38	1
5	5	Green Chile Anytime Sauce	5	13
6	6	Dry Nose Oil	11	11
7	7	Pure Coconut Water With Orange	98	7
8	8	Cut Russet Potatoes Steam N' Mash	116	1

	order_id	product_id	add_to_cart_order	reordered
1	1	49302	1	1
2	1	11109	2	1
3	1	10246	3	0

Cleaning the Data

Step One: Dropping

- Removed irrelevant tables
- Handled null values

Step Two: Merging

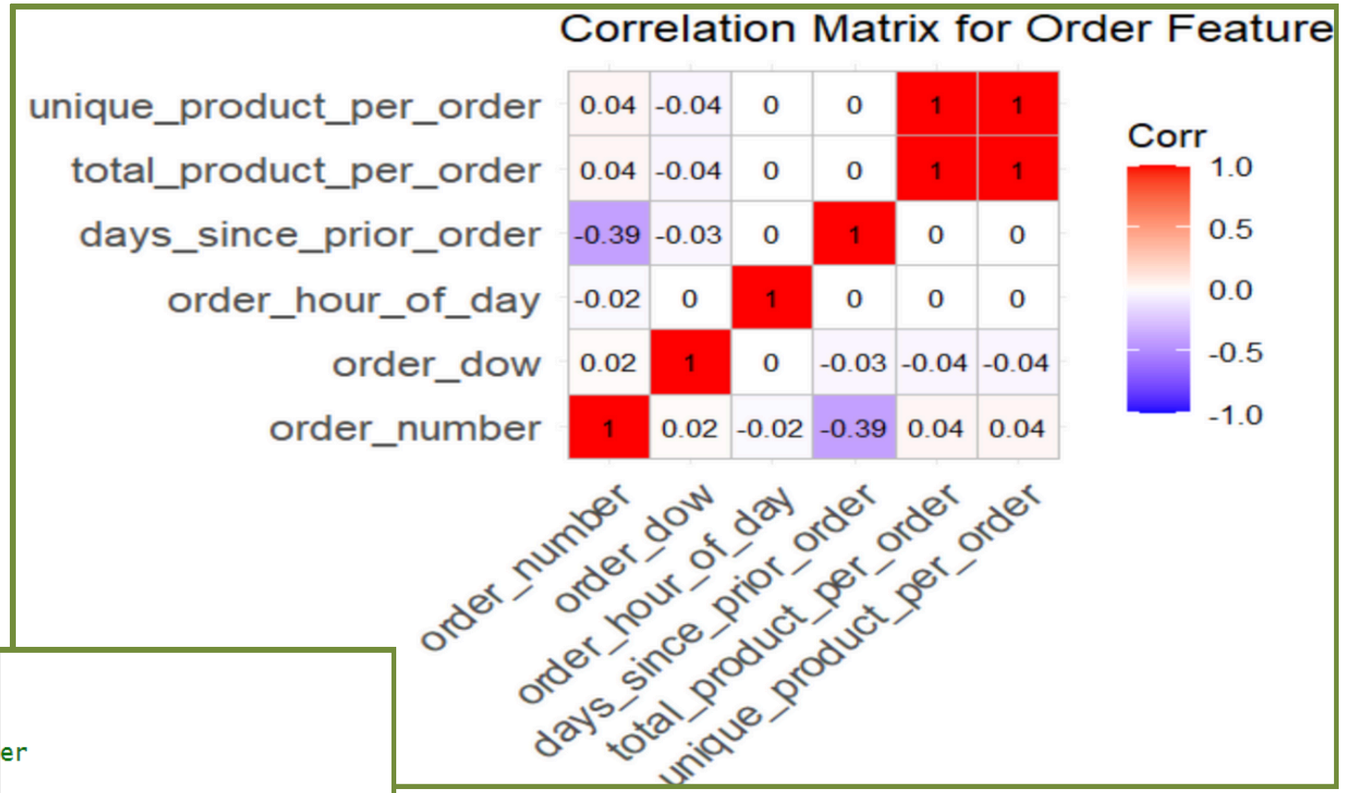
- joined tables for organization and reduction of table references

Step Three: Modifcation

- Removed columns unnecessary to our descriptive analytics
- Added columns for greater interpretability

Variable Correlation

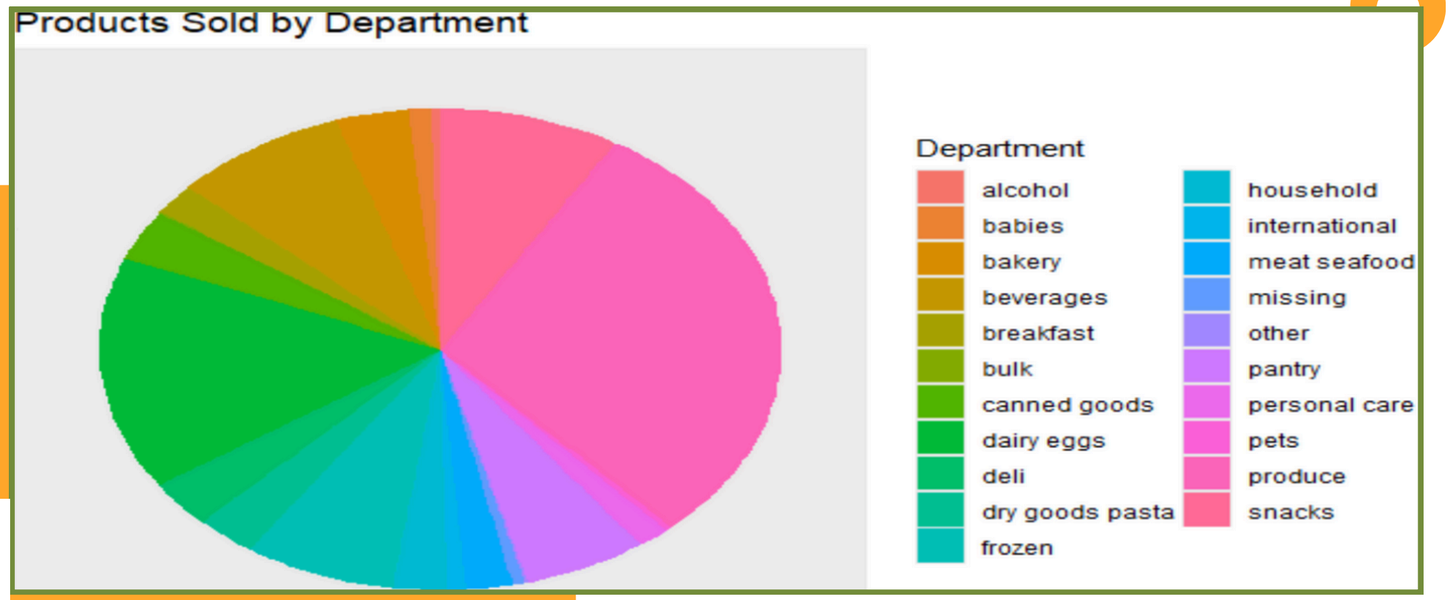
- Categorical Plots
- Clustering
- Customer Behavior modeling




```
customer_order_sum <- sqldf('
                            SELECT order_id,
                            COUNT(*) AS total_product_per_order,
                            COUNT(DISTINCT product_id) AS unique_product_per_order
                            FROM order_products_train
                            GROUP BY order_id'
                            )

# Combine numeric columns from orders_cleaned and order_products_train to visualize
# any linear relationships via correlation
corr_numeric_data <- sqldf('
                           SELECT o.order_number, o.order_dow, o.order_hour_of_day,
                           o.days_since_prior_order, s.total_product_per_order, s.unique_product_per_order
                           FROM orders_cleaned o
                           INNER JOIN customer_order_sum s ON o.order_id = s.order_id'
                           )
```

```
corr_matrix <- cor(corr_numeric_data)

ggcorrplot(corr_matrix, lab = TRUE, lab_size = 3, title = 'Correlation Matrix for Order Features',
           colors = c('blue', 'white','red'))
```

Department Popularity

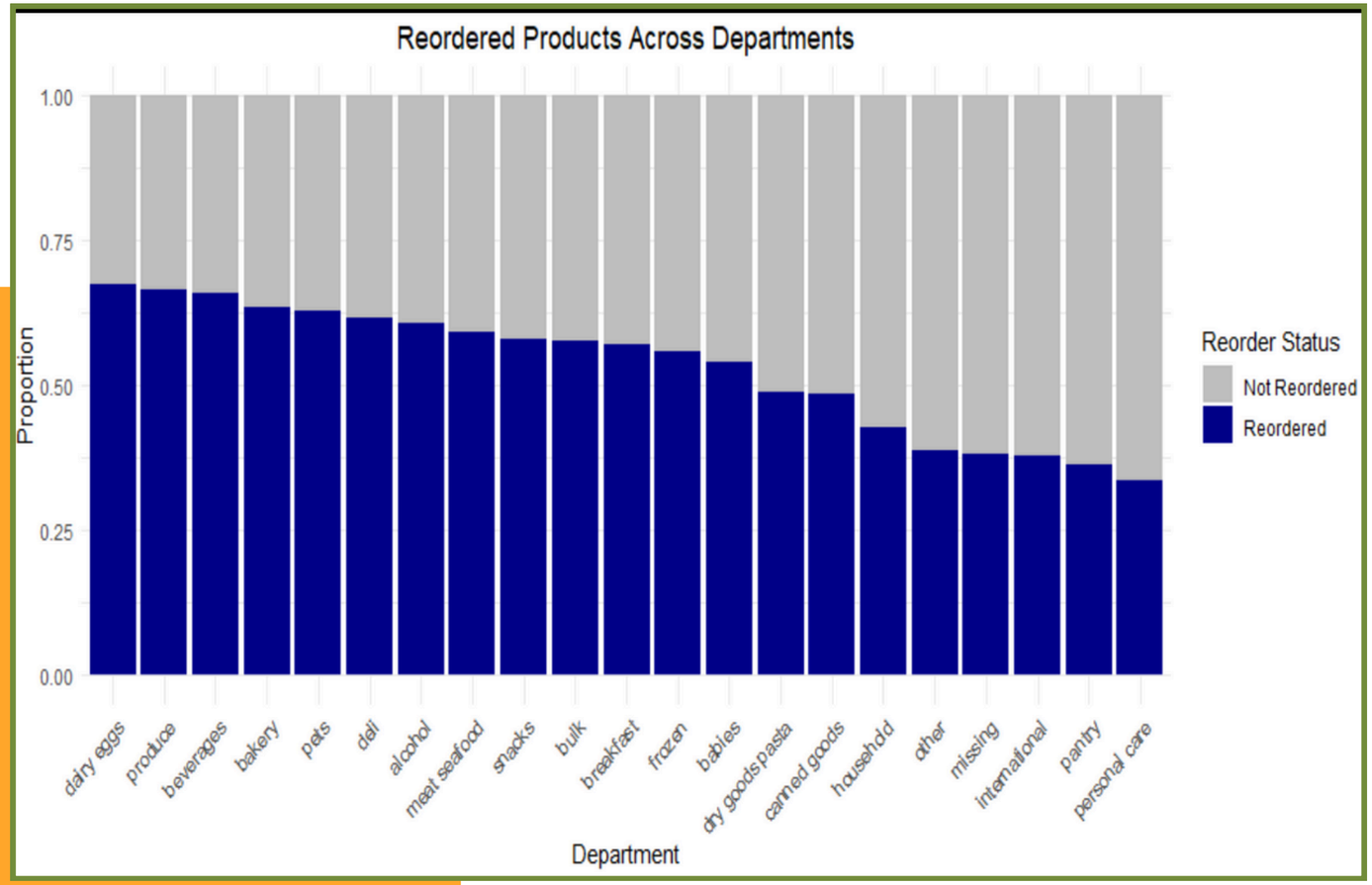



```r
#ggplot2 Treemap creation
library(treemapify)

department_sales <- products_cleaned %>% #New table for plotting purposes
  left_join(order_products_train, by = "product_id") %>%
  count(department)
ggplot(department_sales, aes(area = n, fill = department, label = department)) +
  geom_treemap() +
  geom_treemap_text(fontface = "bold", color = "white", place = "centre", grow = TRUE) +
  labs(title = "Department Sales") +
  theme_minimal()
```


```r
ggplot(pie_data, aes(x = "", y = proportion, fill = department)) +
  geom_bar(stat = "identity") +
  coord_polar("y") +
  labs(
    title = "Products Sold by Department",
    x = NULL,
    y = NULL,
    fill = "Department"
  ) +
  theme(
    axis.text.x = element_blank(),
    axis.text.y = element_blank(),
    panel.grid = element_blank()
  )
```

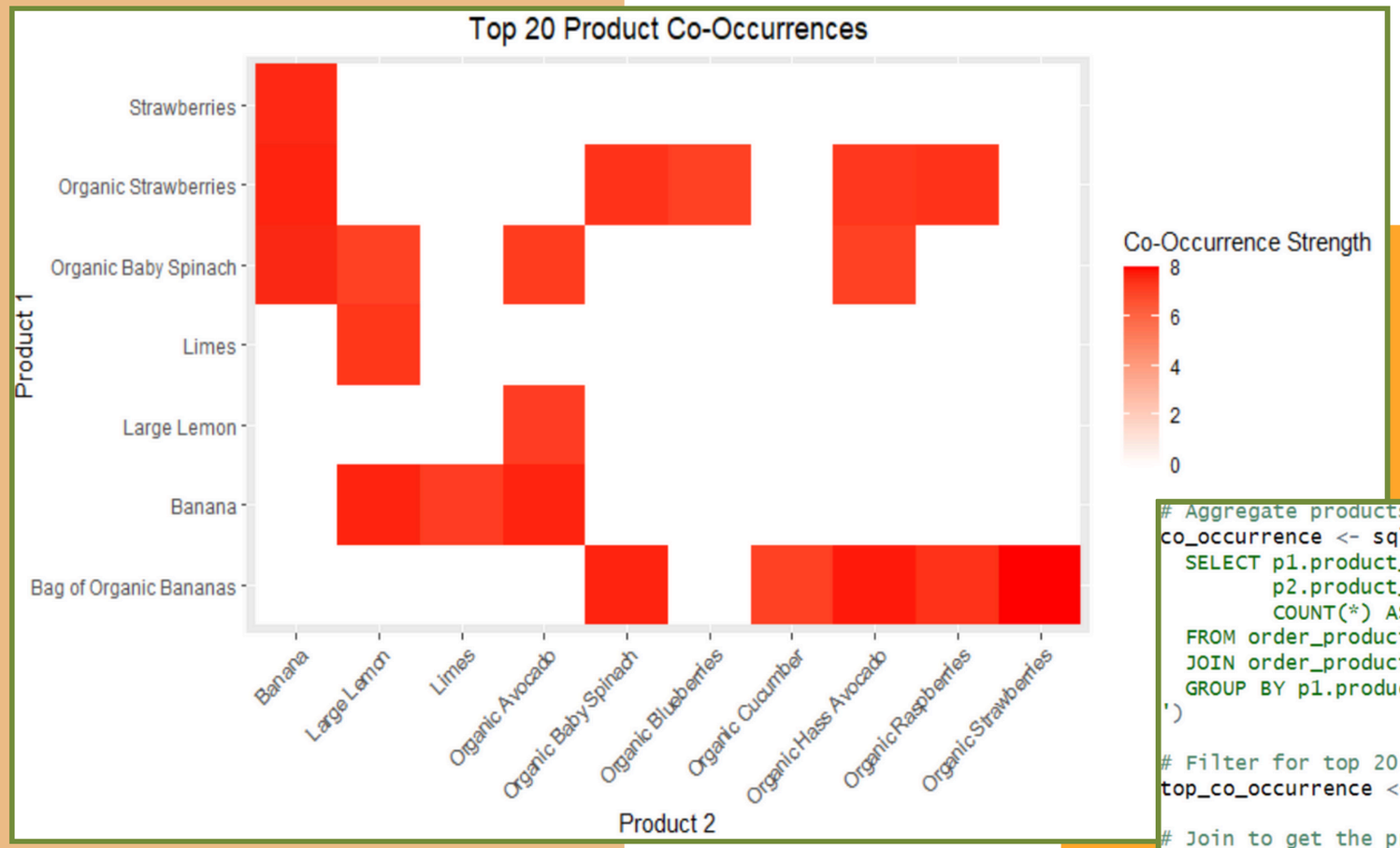
Reordering Rates

- Consumption and Product Turnover
- Inventory planning
- Incentives




```
ggplot(barchart_data, aes(x = reorder(department, -proportion * (reordered == 1)), y = proportion, fill = factor(reordered))
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("0" = "grey", "1" = "darkblue"), labels = c("Not Reordered", "Reordered")) +
  labs(
    title = "Reordered Products Across Departments",
    x = "Department",
    y = "Proportion",
    fill = "Reorder Status"
  ) +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```

Common Combos




```r
# Aggregate products that occur in together in an order
co_occurrence <- sqldf('
  SELECT p1.product_id AS product_id_1,
         p2.product_id AS product_id_2,
         COUNT(*) AS co_occurrence
  FROM order_products_train p1
  JOIN order_products_train p2 ON p1.order_id = p2.order_id AND p1.product_id < p2.product_id
  GROUP BY p1.product_id, p2.product_id
')

# Filter for top 20 co-occurrences
top_co_occurrence <- co_occurrence[order(-co_occurrence$co_occurrence), ][1:20, ]

# Join to get the product names for plot visualization interpretability
top_co_occurrence <- sqldf('
  SELECT c.*, p1.product_name AS product_name_1, p2.product_name AS product_name_2
  FROM top_co_occurrence c
  JOIN products_cleaned p1 ON c.product_id_1 = p1.product_id
  JOIN products_cleaned p2 ON c.product_id_2 = p2.product_id
')
```

SQL Query to see the most sold item in each Department

```
# SQL query to find the top product per department (Histogram)
top_item_per_department <- "SELECT department, product_name
FROM (
  SELECT
  p.department,
  p.product_name,
  COUNT(o.product_id) AS total_orders,
  ROW_NUMBER() OVER (PARTITION BY p.department ORDER BY COUNT(o.product_id) DESC) AS rank
  FROM
  order_products_train o
  JOIN
  products_cleaned p
  ON
  o.product_id = p.product_id
  GROUP BY
  p.department, p.product_name
) AS subquery
WHERE rank = 1"
sqldf(top_item_per_department)
```

	department	product_name
1	alcohol	Sauvignon Blanc
2	babies	Baby Food Stage 2 Blueberry Pear & Purple Carrot
3	bakery	100% Whole Wheat Bread
4	beverages	Sparkling Water Grapefruit
5	breakfast	Honey Nut Cheerios
6	bulk	Dried Mango
7	canned goods	Organic Black Beans
8	dairy eggs	Organic Whole Milk
9	deli	Original Hummus
10	dry goods pasta	Organic Tomato Basil Pasta Sauce
11	frozen	Blueberries
12	household	100% Recycled Paper Towels
13	international	Taco Seasoning
14	meat seafood	Boneless Skinless Chicken Breasts
15	missing	Organic Riced Cauliflower
16	other	Roasted Almond Butter
17	pantry	Extra Virgin Olive Oil
18	personal care	Lavender Hand Soap
19	pets	Double Duty Advanced Odor Control Clumping Cat Litter
20	produce	Banana
21	snacks	Lightly Salted Baked Snap Pea Crisps

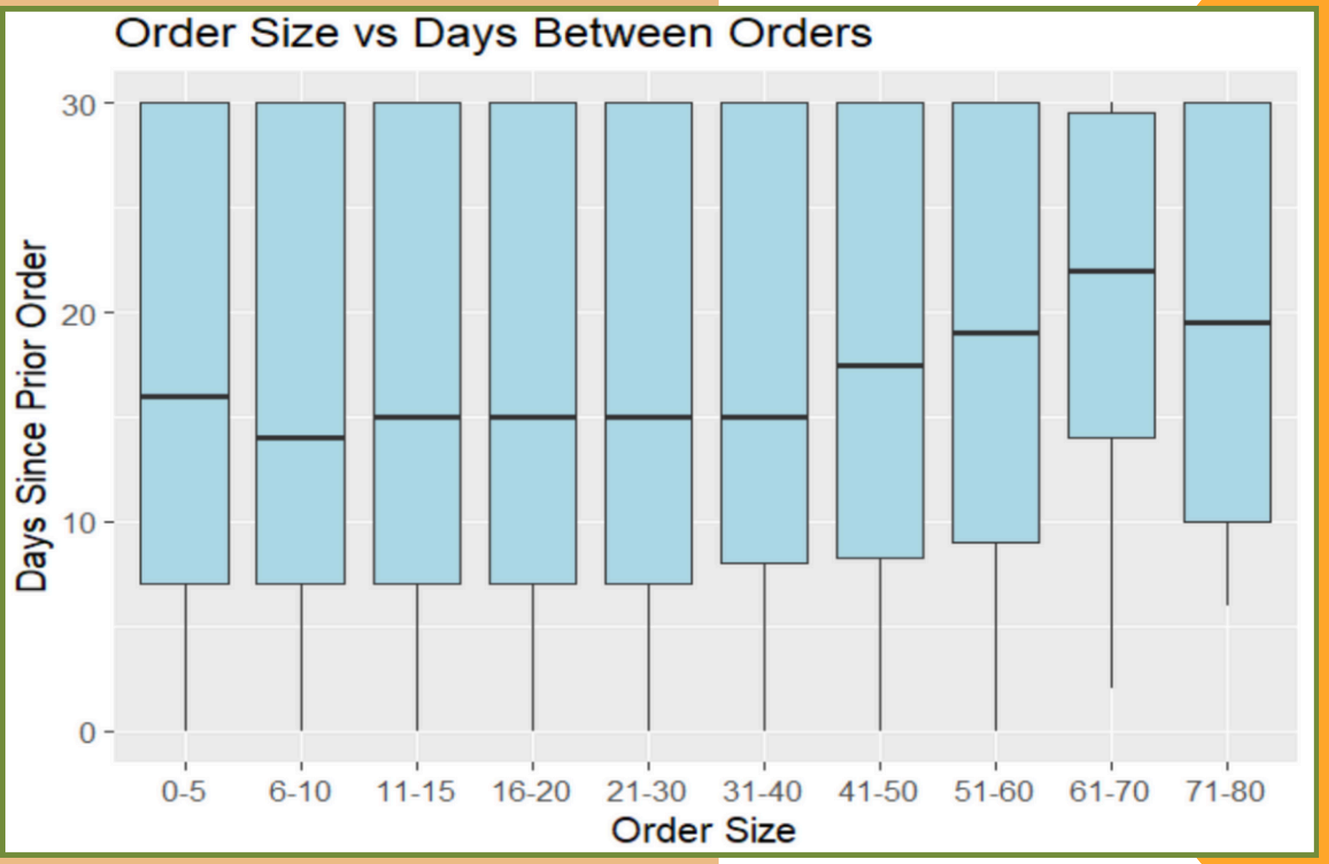
>

Top 10 Selling Items by Department

- Clear stand-alone performer in most Departments

```r
#ggplot2 histogram visualization
top_items_by_department <- order_products_train %>%
  group_by(product_id) %>%
  summarise(total_orders = n()) %>%
  left_join(products_cleaned, by = "product_id") %>%
  group_by(department) %>%
  arrange(department, desc(total_orders)) %>%
  slice_head(n = 10) %>%  #top 10 products per department
  ungroup()
#facet-wrapped plot by department, showing most sold items by department
ggplot(top_items_by_department, aes(x = reorder(product_name, total_orders), y = total_orders)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +
  facet_wrap(~ department, scales = "free_y") +  #Allow independent y-axis scales
  labs(
    title = "Top 10 Selling Products in Each Department",
    x = "Product Name",
    y = "Number of Orders"
  ) +
  theme_minimal() +
  theme(
    axis.text.y = element_text(size = 6),   #Adjust product name text size
    axis.text.x = element_text(size = 8),   #Adjust x-axis tick label size
    strip.text = element_text(face = "bold", size = 10))  #Department labels styling
```

Order Size & Order Activity

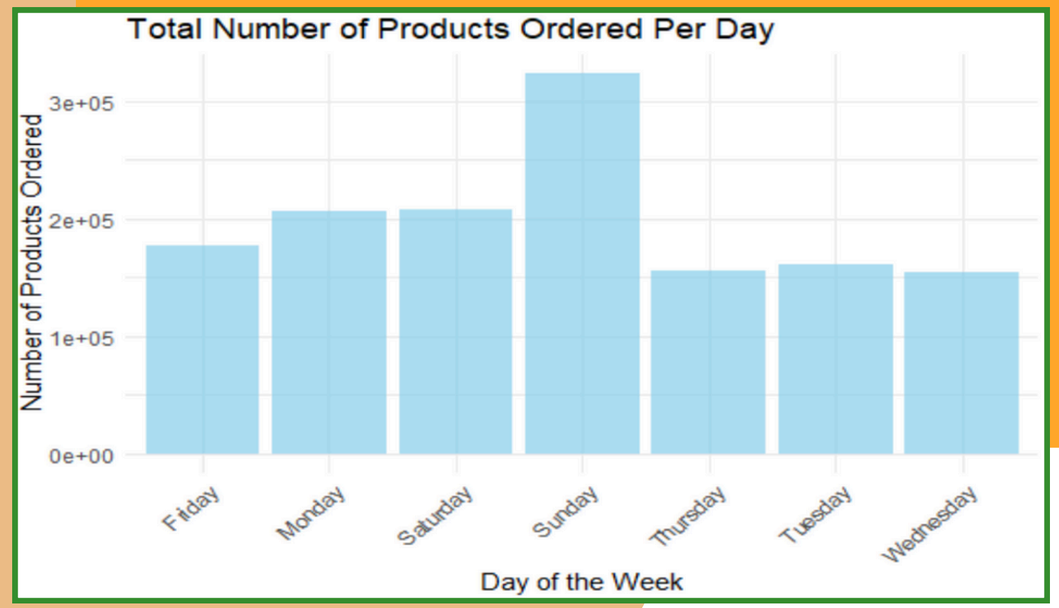


- Tailor marketing

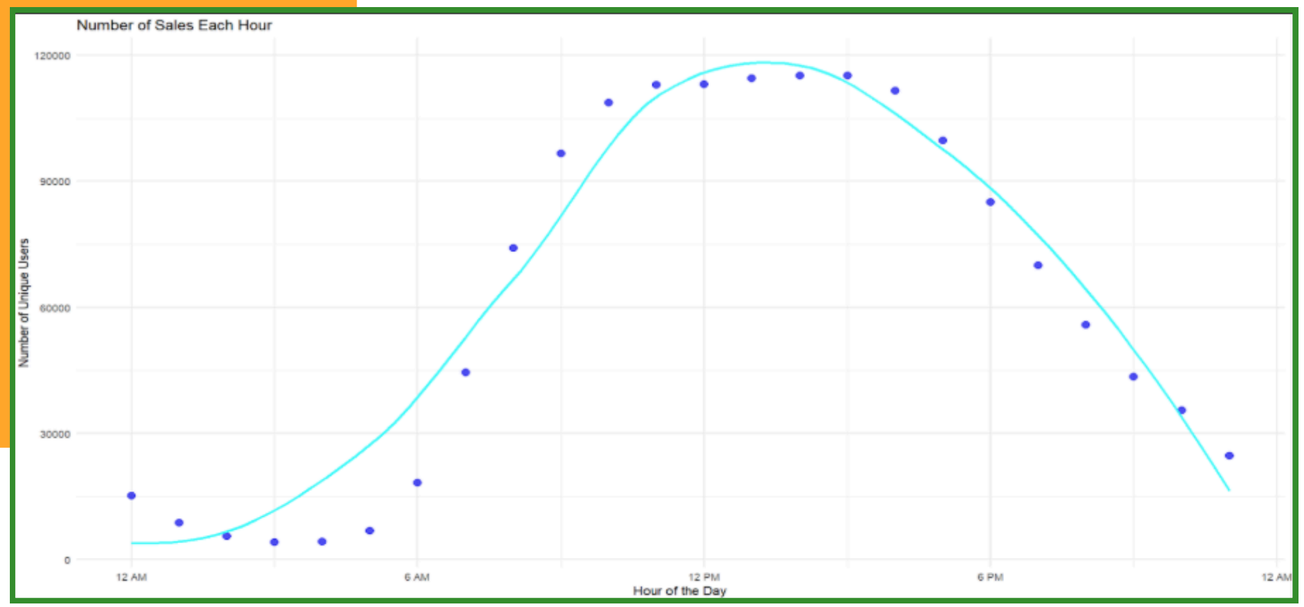
```
# Get the order size by counting the number of products per order
# Join the data together for analysis in the boxplot between order sizes and time gaps between orders
boxplot_data <- sqldf("
                      SELECT o.user_id, o.order_number, o.days_since_prior_order,
                      CASE
                        WHEN s.order_size BETWEEN 0 AND 5 THEN '0-5'
                        WHEN s.order_size BETWEEN 6 AND 10 THEN '6-10'
                        WHEN s.order_size BETWEEN 11 AND 15 THEN '11-15'
                        WHEN s.order_size BETWEEN 16 AND 20 THEN '16-20'
                        WHEN s.order_size BETWEEN 21 AND 30 THEN '21-30'
                        WHEN s.order_size BETWEEN 31 AND 40 THEN '31-40'
                        WHEN s.order_size BETWEEN 41 AND 50 THEN '41-50'
                        WHEN s.order_size BETWEEN 51 AND 60 THEN '51-60'
                        WHEN s.order_size BETWEEN 61 AND 70 THEN '61-70'
                        WHEN s.order_size BETWEEN 71 AND 80 THEN '71-80'
                        ELSE '80+'
                      END AS order_size_in_buckets
                      FROM orders_cleaned o
                      JOIN (
                        SELECT order_id, COUNT(product_id) AS order_size
                        FROM order_products_train
                        GROUP BY order_id
                      ) s ON o.order_id = s.order_id"
                      )
# Order the buckets for ggplot
boxplot_data$order_size_in_buckets <- factor(
  boxplot_data$order_size_in_buckets,
  levels = c('0-5', '6-10', '11-15', '16-20', '21-30',
             '31-40', '41-50', '51-60', '61-70', '71-80', '80+'),
  ordered = TRUE
)
```

Total Orders (by Hour of the Day)

```
ggplot(bar_data, aes(x = order_day, y = num_products)) +
  geom_bar(stat = "identity", fill = "skyblue", alpha = 0.7) +
  labs(title = "Total Number of Products Ordered Per Day", x = "Day of the Week", y = "Number of Products Ordered") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

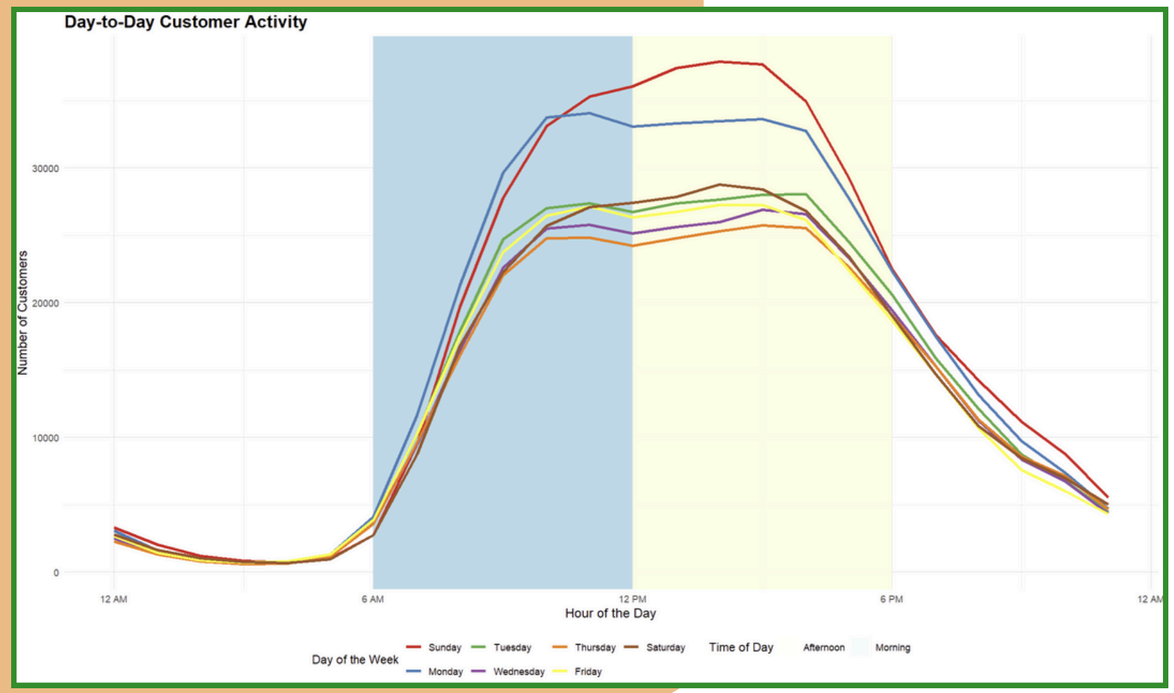



```
#ggplot2 Scatterplot Visualization
#Finding amount of sales per hour each day
hourly_orders <- orders_cleaned %>%
  group_by(order_hour_of_day, order_dow) %>%
  summarise(unique_users = n_distinct(user_id)) %>%

ggplot(hourly_orders, aes(x = order_hour_of_day, y = unique_users)) +
  geom_point(color = "blue", size = 3, alpha = 0.7) +
  geom_smooth(method = "loess", color = "cyan", se = FALSE) +  #Smoothed trend lin
  scale_x_continuous(
    breaks = c(0, 6, 12, 18, 24),
    labels = c("12 AM", "6 AM", "12 PM", "6 PM", "12 AM")
  ) +
  labs(
    title = "Number of Sales Each Hour",
    x = "Hour of the Day",
    y = "Number of Unique Users"
  ) +
  theme_minimal()
```



Day to Day Orders by the Hour

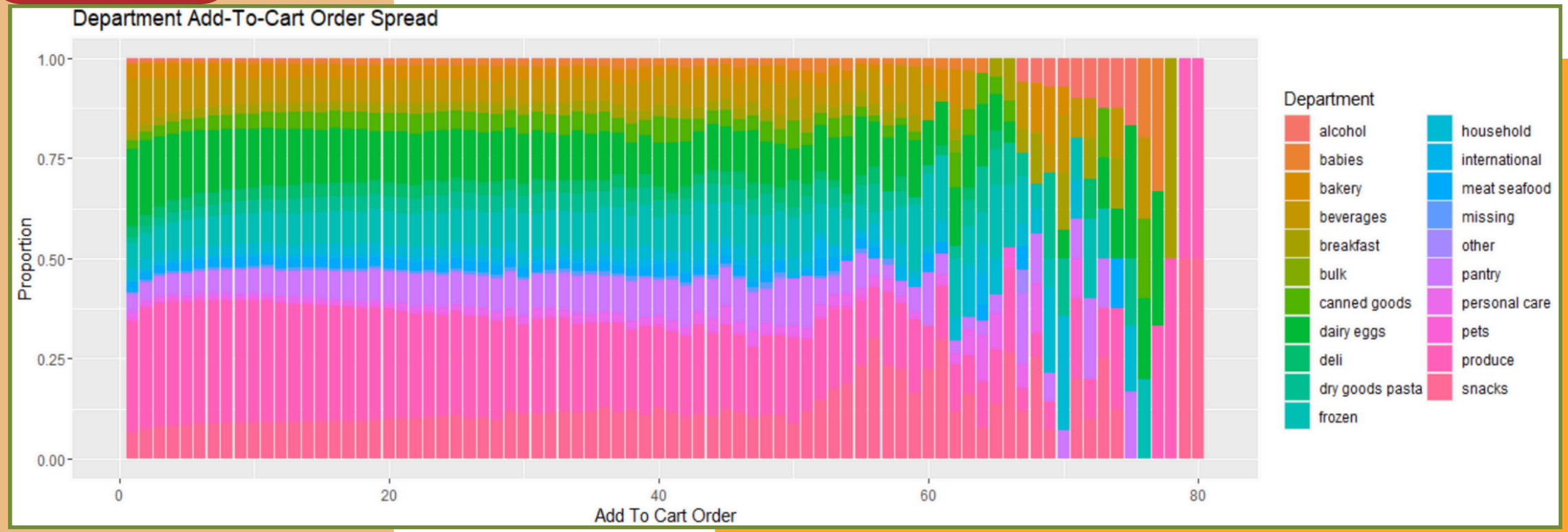
```r
#Finding amount of customers per hour each day
hourly_customers <- orders_cleaned %>%
  group_by(order_dow, order_hour_of_day) %>%
  summarise(unique_users = n_distinct(user_id))


#Line Chart creation
ggplot(hourly_customers, aes(x = order_hour_of_day, y = unique_users, color = factor(order_dow))) +
  #Add transparent shaded areas
  geom_rect(aes(xmin = 6, xmax = 12, ymin = -Inf, ymax = Inf, fill = "Morning"), alpha = 0.1, inherit.aes = FALSE) +  #Morning
  geom_rect(aes(xmin = 12, xmax = 18, ymin = -Inf, ymax = Inf, fill = "Afternoon"), alpha = 0.1, inherit.aes = FALSE) +  #Afternoo
  geom_line(size = 1.2) +  # Customer activity lines
  scale_fill_manual(
    name = "Time of Day",
    values = c("Morning" = "lightblue", "Afternoon" = "lightyellow")
  ) +
  scale_x_continuous(
    breaks = c(0, 6, 12, 18, 24),
    labels = c("12 AM", "6 AM", "12 PM", "6 PM", "12 AM")
  ) +
  labs(
    title = "Day-to-Day Customer Activity",
    x = "Hour of the Day",
    y = "Number of Customers",
    color = "Day of the Week"
  ) +
  scale_color_brewer(palette = "Set1", labels = c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday")) +
  theme_minimal() +
  theme(
    legend.position = "bottom",
    plot.title = element_text(size = 16, face = "bold"),
    plot.subtitle = element_text(size = 12),
    axis.title = element_text(size = 12))
```




```r
# Query to find the top 2 hours(for sake of output length) with the most unique customers for each day of the week
top_hours_by_day <- "SELECT day_of_week, order_hour_of_day, unique_customers
  FROM (
    SELECT
      CASE order_dow
        WHEN 0 THEN 'Sunday'
        WHEN 1 THEN 'Monday'
        WHEN 2 THEN 'Tuesday'
        WHEN 3 THEN 'Wednesday'
        WHEN 4 THEN 'Thursday'
        WHEN 5 THEN 'Friday'
        WHEN 6 THEN 'Saturday'
      END AS day_of_week,
      order_hour_of_day,
      COUNT(DISTINCT user_id) AS unique_customers,
      ROW_NUMBER() OVER (PARTITION BY order_dow ORDER BY COUNT(DISTINCT user_id) DESC) AS rank
    FROM
      orders_cleaned
    GROUP BY
      order_dow, order_hour_of_day
  ) AS ranked_hours
  WHERE rank <= 2"
sqldf(top_hours_by_day)
```

	day_of_week	order_hour_of_day	unique_customers
1	Sunday	14	37888
2	Sunday	15	37708
3	Monday	11	34082
4	Monday	10	33767
5	Tuesday	16	28037
6	Tuesday	15	28006
7	Wednesday	15	26913
8	Wednesday	16	26565
9	Thursday	15	25744
10	Thursday	16	25541
11	Friday	14	27269
12	Friday	15	27267
13	Saturday	14	28778
14	Saturday	15	28426

Add-To-Cart Positional Trends




```
combined_df <- sqldf('
               SELECT o.add_to_cart_order, p.department
               FROM order_products_train o
               LEFT JOIN products_cleaned p ON o.product_id = p.product_id')

# Calculate the proportion departments are added to a customer's cart at specific spots
stacked_hist_data <- sqldf('
               SELECT add_to_cart_order, department, COUNT(*) AS count,
               COUNT(*) * 1.0 / SUM(COUNT(*)) OVER (PARTITION BY add_to_cart_order) as proportion
               FROM combined_df
               GROUP BY add_to_cart_order, department
               ORDER BY add_to_cart_order, department'
               )
```

Conclusion

Opportunities for future work

Large-scale data issues

Thank You

Order Now