Ask Questions to Make Data-Driven Decisions

Step 1: Define the Problem and Business Objective

Scenario

You are a junior data analyst working in the marketing analyst team at Cyclistic, a bike-share company in Chicago. The director of marketing believes the company’s future success depends on maximizing the number of annual memberships. Therefore, your team wants to understand how casual riders and annual members use Cyclistic bikes differently. From these insights, your team will design a new marketing strategy to convert casual riders into annual members. But first, Cyclistic executives must approve your recommendations, so they must be backed up with compelling data insights and professional data visualizations.

Characters and teams

● Cyclistic: A bike-share program that features more than 5,800 bicycles and 600 docking stations. Cyclistic sets itself apart by also offering reclining bikes, hand tricycles, and cargo bikes, making bike-share more inclusive to people with disabilities and riders who can’t use a standard two-wheeled bike. The majority of riders opt for traditional bikes; about 8% of riders use the assistive options. Cyclistic users are more likely to ride for leisure, but about 30% use them to commute to work each day.

● Lily Moreno: The director of marketing and your manager. Moreno is responsible for the development of campaigns and initiatives to promote the bike-share program. These may include email, social media, and other channels.

● Cyclistic marketing analytics team: A team of data analysts who are responsible for collecting, analyzing, and reporting data that helps guide Cyclistic marketing strategy. You joined this team six months ago and have been busy learning about Cyclistic’s mission and business goals — as well as how you, as a junior data analyst, can help Cyclistic achieve them.

● Cyclistic executive team: The notoriously detail-oriented executive team will decide whether to approve the recommended marketing program.

### **Discovering connections**

**A third-party logistics company working with another company to get shipments delivered to customers on time is a problem requiring analysts to discover connections. By analyzing the wait times at shipping hubs, analysts can determine the appropriate schedule changes to increase the number of on-time deliveries.**

There are Six main different types of problems that we can look into for the project. However, the problem type that would fit best would be to Discover Connections. In essence we are trying to determine these three things:

Ask Three questions will guide the future marketing program:

1. How do annual members and casual riders use Cyclistic bikes differently?

2. Why would casual riders buy Cyclistic annual memberships?

3. How can Cyclistic use digital media to influence casual riders to become members?

You will produce a report with the following deliverables:

1. A clear statement of the business task

2. A description of all data sources used

3. Documentation of any cleaning or manipulation of data

4. A summary of your analysis

5. Supporting visualizations and key findings

6. Your top three recommendations based on your analysis

### **Business Objective**

Your team will design a new marketing strategy to convert casual riders into annual members

Step 2: Prepare the Data

You will use Cyclistic’s historical trip data to analyze and identify trends. Download the previous 12 months of Cyclistic trip data here. (Note: The datasets have a different name because Cyclistic is a fictional company. For the purposes of this case study, the datasets are appropriate and will enable you to answer the business questions. The data has been made available by Motivate International Inc. under this license.) This is public data that you can use to explore how different customer types are using Cyclistic bikes. But note that data-privacy issues prohibit you from using riders’ personally identifiable information. This means that you won’t be able to connect pass purchases to credit card numbers to determine if casual riders live in the Cyclistic service area or if they have purchased multiple single passes. Now, prepare your data for analysis using the following Case Study Roadmap as a guide:

Case Study Roadmap - Prepare Guiding questions

● Where is your data located?

The data is located within Big Query, they are saved as individual sheets on my desktop as well as the cloud with Google with Big Query

● How is the data organized?

The data is organized into many spreadsheets and is organized by trip or ride ids. The data explains the information about each ride which occurred over the past year.

● Are there issues with bias or credibility in this data? Does your data ROCCC?

To make sure that the types of bikes used were the correct type I ran this code:

SELECT DISTINCT rideable\_type

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`;

It came back with only docked bike, classic bike and an electric bike. All the bikes fall into these three categories that were giving with the data type, so the bike type is reliable.

I ran a search to see where the starting places were for the ride, I used this code to find similar starting places:

SELECT DISTINCT(Start\_station\_name), start\_lat, start\_lng

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE start\_lat BETWEEN 40 AND 43 AND start\_lng BETWEEN -88 AND -84;

And got 1,175,000 results, there are 1 Million 175 thousand starting places between

I compared it to results on this code:

SELECT DISTINCT(Start\_station\_name), start\_lat, start\_lng

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`;

I got 1,175,000 results again

I compared these results (from this code):

SELECT Start\_station\_name, start\_lat, start\_lng

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE start\_lat BETWEEN 40 AND 43 AND start\_lng BETWEEN -88 AND -84;

And got 5,136,261 results

I ran this code and got the same number of results (5,136,261), this shows that all of the starting positions are within the same geographical area, there are no major outliers that would mean that the starting position shouldn’t be included in the data

SELECT Start\_station\_name, start\_lat, start\_lng

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`;

I searched distinct members status with this code:

SELECT DISTINCT(member\_casual)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`;

And I got only two different results, member and casuals are the results and they are what we are comparing so this part of the data is reliable

I ran the code:

SELECT started\_at, ended\_at

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE started\_at >= '2020-10-01 00:00:00' AND ended\_at <='2021-09-30 23:59:59'

ORDER BY started\_at

To make sure that the data is within the date ranges of the past year, I got 5,136,139

I can’t figure out why this number isn’t the same as the total number of results. When I ran the code:

SELECT \*

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE started\_at < '2020-10-01 00:00:00' AND ended\_at >'2021-10-01 00:00:00'

ORDER BY started\_at

I got zero results for the previous code. All the data is current or in the correct time frame for our purposes

It’s hard to determine whether the data is biased because I don’t know the population size (as opposed to this sample). To know whether or not the data has bias we need to know the population size is (compared to the sample size). We also need to know how random the data is when it comes to collection.

● How are you addressing licensing, privacy, security, and accessibility?

I checked the license for the data. I do not plan to use the data for commercial purposes to I believe I am in my rights to use the data. The license is located here [Data License Agreement | Divvy Bikes](https://www.divvybikes.com/data-license-agreement).

In terms of privacy, I will not publish the Rider’s Identification number in the final version of the report. The only potential problem with the data is the fact that there is 122 starting times that are unaccounted for, however, there were no staring time outside the yearly time frame for the data, so we’ll just have to assume that there is something we’re not accounting for that we can’t see at the moment, but the data starting times still adhere to the year that we collected for.

● How did you verify the data’s integrity?

I got the data directly from [Index of bucket "divvy-tripdata"](https://divvy-tripdata.s3.amazonaws.com/index.html) or a primary source. Primary sources are considered to have more integrity than Secondary Sources or Third-Party sources. It is considered fictional, but for our purposes it works.

● How does it help you answer your question?

This makes us sure that the data we are using is correct, if the data isn’t correct, even if we do the calculations correctly, it will not work.

● Are there any problems with the data? Key tasks

1. Download data and store it appropriately.

2. Identify how it’s organized.

3. Sort and filter the data.

I tried pasting all last year’s rows of data into one Excel sheet. It didn’t work. I’m going to use Big Query to merge the data with SQL by using Union All.

SELECT \*

FROM `cycling-project-328223.202010.2020-10`;

SELECT \*

FROM `cycling-project-328223.202011.2020-11`;

I combined two tables from the Datasets given by the cycling company.

SELECT \*

FROM `cycling-project-328223.202010.2020-10`

UNION ALL

SELECT \*

FROM `cycling-project-328223.202010.2020-11`;

This combined the tables as a test. I saved the results as a Big Query Table (it will be too large to save as a single Google Sheet document or a single Excel document) I did this for every single table for the last year in Big Query. I went into each Excel file (except the first one) and deleted the header rows in order to merge them into a new table with just one header row. Working with this file will be too large to be done in Excel or Google Sheets. I had to use R and Big Query.

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2020-11`;

I couldn’t upload the entire CSV file for Trippdata 2021-06 through 2021-09. I had to split them into multiple files (A file and B file) to get around the 100 mb file upload limit on Big Query.

This was the First Semi-Merged Table. It consists of 2020-10 and 2020-11.

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2020-10-no-headers`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2020-11`

This is the Second Semi-Merged Table. It consists of 2020-12, 2021-01, 2021-02.

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2020-12`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-01`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-02`

This is the Third Semi-Merged Table. It includes tables 2021-03, 2021-04, 2021-05, 2021-05, 2021-06-A, 2021-06-B, 2021-07-A, 2021-07-B, 2021-08-A, 2021-08-B, 2021-09-A, 2021-09-B

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-03`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-04`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-05`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-06-A`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-06-B`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-07-A`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-07-B`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-08-A`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-08-B`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-09-A`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Unmerged\_Data\_Sets.2021-09-B`;

I created a new table to replace the old table while changing the int64\_field\_5 and in64\_field\_7 into string data types so that the Semi-Merged Tables could be merged into one single table.

CREATE TABLE `cycling-project-328223.Semi\_Merged\_Data\_Sets.2020-10-Through-2020-11-Formatted` AS

   SELECT string\_field\_0, string\_field\_1, timestamp\_field\_2, timestamp\_field\_3, string\_field\_4, SAFE\_CAST(int64\_field\_5 AS string) AS string\_field\_5, string\_field\_6,SAFE\_CAST(int64\_field\_7 AS string) AS string\_field\_7, double\_field\_8, double\_field\_9, double\_field\_10, double\_field\_11, string\_field\_12

   FROM `cycling-project-328223.Semi\_Merged\_Data\_Sets.2020-10-Through-2020-11`;

Now that all the data types match we merge the data tables in Semi\_Merged\_Data\_Sets using this code to get all the tables into a single merged table

SELECT \*

FROM `cycling-project-328223.Semi\_Merged\_Data\_Sets.2020-10-Through-2020-11-Formatted`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Semi\_Merged\_Data\_Sets.2020-12-Through-2021-02`

UNION ALL

SELECT \*

FROM `cycling-project-328223.Semi\_Merged\_Data\_Sets.2021-03-Through-2021-09-B`

We get a table with more than 5 million records! We’ll save this in the Merged\_Data\_Set as 2020-10\_Through\_2021-09. This code will show our combined table (for checking):

SELECT \*

FROM `cycling-project-328223.Merged\_Data\_Set.2020-10\_Through\_2021-09`;

This changes all the column names except for the last column:

SELECT \*, string\_field\_0 as ride\_id, string\_field\_1 as rideable\_type, timestamp\_field\_2 as started\_at, timestamp\_field\_3 as ended\_at,

string\_field\_4 as start\_station\_name, string\_field\_5 as start\_station\_id, string\_field\_6 as end\_station\_name, string\_field\_7 as end\_station\_id,

double\_field\_8 as start\_lat, double\_field\_9 as start\_lng, double\_field\_10 as end\_lat, double\_field\_11 as end\_lng, string\_field\_12 as member\_casual

FROM `cycling-project-328223.Merged\_Data\_Set.2020-10\_Through\_2021-09`;

Save this table as Biking\_Data and drop the extra columns with the code:

ALTER TABLE `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

DROP COLUMN string\_field\_0,

DROP COLUMN string\_field\_1,

DROP COLUMN timestamp\_field\_2,

DROP COLUMN timestamp\_field\_3,

DROP COLUMN string\_field\_4,

DROP COLUMN string\_field\_5,

DROP COLUMN string\_field\_6,

DROP COLUMN string\_field\_7,

DROP COLUMN double\_field\_8,

DROP COLUMN double\_field\_9,

DROP COLUMN double\_field\_10,

DROP COLUMN double\_field\_11,

DROP COLUMN string\_field\_12;

Check to make sure the final merged table is completed with this code:

SELECT \*

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`;

Step 3: Process the Data

● What tools are you choosing and why?

I used Big Query, RStudio and Tableau. I didn’t make use of Excel because it could not handle 5 million plus rows of data. Big Query, RStudio and Tableau was able to handle more data. Also it requires more input and skill than Excel so I wanted to brush up on the different tools.

● Have you ensured your data’s integrity?

I made sure the data is Reliable and Current with my code in the previous step. This along with the fact the data is from a primary source shows that the data has integrity.

● What steps have you taken to ensure that your data is clean?

I ran some queries in the previous step to make sure that the data lined up as part of ROCCC.

● How can you verify that your data is clean and ready to analyze?

Running queries and checking the sums of those queries and filtered queries against the totals can help verify the data. Also filtering for things such as specific geographical areas can help verify the data too.

● Have you documented your cleaning process so you can review and share those results

[See my code above](#Cleaning_Code).

Create a column called ride\_length as data type interval.

ALTER TABLE `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

ADD COLUMN ride\_length INTERVAL;

This DML (Data Manipulation Language) code creates a calculated column based on the ride times.

UPDATE `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

SET ride\_length = (ended\_at - started\_at)

WHERE TRUE;

Check to see if the computed column works (use limit 10 to speed up the process and save resources on Big Query):

SELECT \*

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

LIMIT 10;

I was not able to get the day of the week shown in the data. It is likely some form of SQL coding that I’ll have to get. The amount of data made it impossible to open in Excel or Google sheets so all of the processing will have to be done by RStudio and Big Query.

Figuring Out Which Columns are Categories and Which are Sub Categories:

This is our table of “Categories” and “Subcategories”. The Categories are measurements that can be measured independently of subcategories and you can use aggregate functions with them. They still may need to be split up based on things like time (however, time is not supposed to be a subcategory) The Subcategories are the measurements that being measured independently of categories or other subcategories, can become a problem, is not descriptive enough or presents a very clear picture of the data .

Subcategories always have to be paired with a category or another subcategory. Categories can be independent of subcategories or they can be paired with other Categories or Subcategories (but it is not necessary to pair them to get a chart/measurement from them or use an aggregate function with them).

Things such as map coordinates, geographical locations or id columns are usually subcategories. IDs cannot be added or subtracted (using aggregate functions on them is meaningless) and max, min and mean would be pretty meaningless to get from ID columns too. You can use count or count distinct, but not much more than that. This is also true for latitude and longitude values.

I moved the data to RStudio instead of Tableau. RStudio is able to handle this much data. If I uploaded 5 million rows of data into tableau it would take an incredibly long time to do it.

I split up the data from one sheet to one column per a sheet for the importing of CSV data into RStudio. If they aren’t split up to one column a sheet it will be impossible to pull only a single column from the CSV file for the data frame. This worked with the capstone project because it was type .xlsx instead of CSV.

I tried uploading the entire file (Biking\_Data.csv), however, this file is too large for RStudio to Process. I split up the data into separate files based on columns. This is the code I used to upload an individual column as a test. It worked.

# installing packages to use code from

Install.packages("tidyverse")

install.packages("reader")

install.packages("readx1")

install.packages("ggplot2")

# Upload the CSV file

tripdata <- read.csv("202109-divvy-tripdata-A.csv")

View(tripdata)

(Clear previously uploaded columns and data in the environment pane with this code:

Delete Global Environment Objects)

rm(list = ls())

This is the code needed to upload the files, however, because of the size of the files it is better to upload the columns as you currently use them from within the console instead of uploading all of them at one time and then using them later (RStudio kept crashing when I uploaded them all before use and I couldn’t get them all uploaded before it crashed and erased the previous uploaded columns).

Install.packages("tidyverse")

install.packages("reader")

install.packages("readx1")

install.packages("ggplot2")

# Upload the CSV file

end\_lat <- read.csv("end\_lat.csv")

# Upload the CSV file

end\_lng <- read.csv("end\_lng.csv")

# Upload the CSV file

end\_station\_id <- read.csv("end\_station\_id.csv")

# Upload the CSV file

end\_station\_name <- read.csv("end\_station\_name.csv")

# Upload the CSV file

ended\_at <- read.csv("ended\_at.csv")

# Upload the CSV file

member\_casual <- read.csv("member\_casual.csv")

# Upload the CSV file

ride\_id <- read.csv("ride\_id.csv")

# Upload the CSV file

ride\_time <- read.csv("ride\_time.csv")

# Upload the CSV file

rideable\_type <- read.csv("rideable\_type.csv")

# Upload the CSV file

start\_lat <- read.csv("start\_lat.csv")

# Upload the CSV file

start\_lng <- read.csv("start\_lng.csv")

# Upload the CSV file

start\_station\_id <- read.csv("start\_station\_id.csv")

# Upload the CSV file

start\_station\_name <- read.csv("start\_station\_name.csv")

# Upload the CSV file

started\_at <- read.csv("started\_at.csv")

Step 4: Analyze the Data

|  |  |
| --- | --- |
| **Categories** | **Sub Categories** |
| rideable\_type | ride\_id |
| started\_at | start\_station\_id |
| ended\_at | end\_station\_id |
| start\_station\_name | start\_lat |
| end\_station\_name | start\_lng |
|  |  |
| member\_casual | end\_lat |
| ride\_time | end\_lng |

|  |  |  |  |
| --- | --- | --- | --- |
| Top Questions | Measurements |  |  |
| What ride types do members vs. non-members use? | rideable\_type | Count Distinct of member\_casual (where member) | Count Distinct of member\_casual (where non-member) |
| Do Members or Non-members have higher average ride times? | member\_casual | Average of all ride\_time |  |
| What months of the year is Casual and Member rides the highest and lowest? | Count of started\_at (by Month) | ride\_id |  |
| Where are our members vs. non-members starting from with their location for their rides? | start\_station\_name | Count Distinct of member\_casual (where member) | Count Distinct of member\_casual (where non-member) |
| Where are our members vs. non-members ending their rides? | start\_station\_name | Count Distinct of member\_casual (where member) | Count Distinct of member\_casual (where non-member) |
| Map of where our Members are starting | start\_lat | start\_lng |  |
| Map of where our Non-Members are starting | start\_lat | start\_lng |  |
| Do Members or Non-Members total more time riding? | Sum of ride\_time (where member causal = member) | Sum of ride\_time (where member causal = non member) |  |

● How should you organize your data to perform analysis on it?

This is the code to get the data for a stacked bar chart into data frames for the first question from the categories and subcategories table. This code does not include the stacked bar visuals (the next section of [code](#Stacked_Bar_Chart_Code) makes use of these data frames).

SELECT COUNT(rideable\_type)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE member\_casual = 'member' AND rideable\_type = 'docked\_bike';

= 270200

SELECT COUNT(rideable\_type)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE member\_casual = 'member' AND rideable\_type = 'classic\_bike';

= 1630116

SELECT COUNT(rideable\_type)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE member\_casual = 'member' AND rideable\_type = 'electric\_bike';

= 877658

SELECT COUNT(rideable\_type)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE member\_casual = 'casual' AND rideable\_type = 'docked\_bike';

= 407780

SELECT COUNT(rideable\_type)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE member\_casual = 'casual' AND rideable\_type = 'classic\_bike';

= 1120715

SELECT COUNT(rideable\_type)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

WHERE member\_casual = 'casual' AND rideable\_type = 'electric\_bike';

= 829792

I used this code and the to use data frames that would make it possible to create a stacked bar chart in RStudio:

# Create the X column, the y column and the group columns as their own datasets

x\_column <- data.frame( x\_column = rep(c('270200', '1630116', '877658', '407780', '1120715', '829792')))

y\_column <- data.frame( y\_column = rep(c('docked\_bike', 'classic\_bike', 'electric\_bike', 'docked\_bike', 'classic\_bike', 'electric\_bike')))

group\_column <- data.frame(group\_column = rep(c('member', 'member', 'member', 'casual', 'casual', 'casual')))

# check the group\_column

View(group\_column)

# Merge the three data frames into a new data frame

three\_columns <- data.frame(x\_column, y\_column, group\_column)

# Check the data frame with all three data frames merged into it

View(three\_columns)

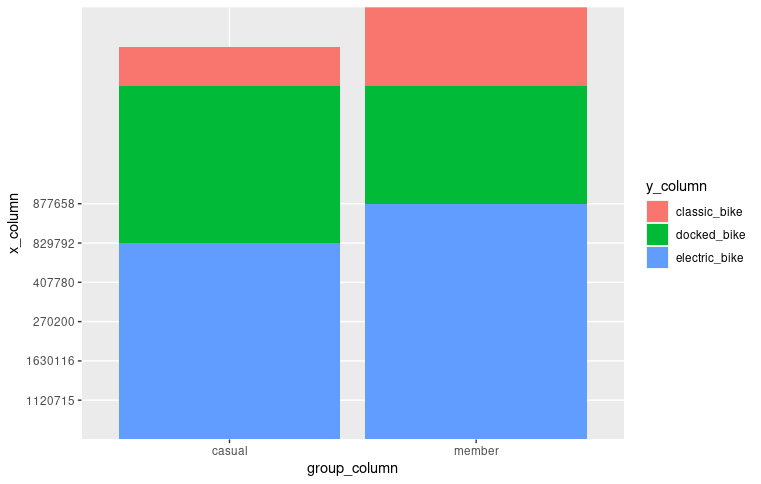
# plotting the stacked bar chart

ggplot(three\_columns, aes(x=group\_column, y=x\_column, fill = y\_column)) + geom\_bar(stat="identity")

# test code

# ggplot(three\_columns, aes(x=group\_column, y=x\_column, fill = y\_column)) + geom\_col()

Look at the [previous code](#Stacked_Bar_Chart_Code) to get this chart.



(“Divvy-Tripdata.”)

This is the code for the month by month riding data of Members and Casuals in the city of Chicago:

install.packages("tidyverse")

install.packages("reader")

install.packages("readx1")

install.packages("ggplot2")

install.packages("dplyr")

library(ggplot2)

library(reader)

library(readxl)

library(dplyr)

# Upload the Excel file to RStudio

brbm <- read\_excel("Bike Riders by Month.xlsx")

# Create a data frame for the Excel file uploaded to RStudio

brdf <- data.frame(brbm)

# Check the brdf data frame from the previous step

View(brdf)

# Select just the Date column from brdf

brdate <- brdf %>%

select(Date)

# Select the Amount column from brdf

bramount <- brdf %>%

select(Amount)

# Select the Status column from brdf

brstatus <- brdf %>%

select(Status)

# turn brdate list data type into numbers

brdate <- unlist(brdate)

# turn bramount list data type into numbers

bramount <- unlist(bramount)

# turn brstatus list data type into numbers

brstatus <- unlist(brstatus)

# Bar graph, month on x-axis, color fill grouped by status -- use position\_dodge()

ggplot(data=brdf, aes(x=brdate, y=bramount, fill=brstatus)) +

geom\_bar(stat="identity", position=position\_dodge())

This is the chart that the [previous code](#Month_By_Month_Code) created:

Chart, bar chart

Description automatically generated

(“Divvy-Tripdata.”)

Unfortunately, because the data is stored in Interval type, the average (using the previous code) cannot be calculated, we’ll have to convert ride time to a time datatype.

Add column for seconds calculation as a numeric data type:

ADD COLUMN seconds\_difference numeric;

Drop the unused columns for the table we’re creating:

ALTER TABLE `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`

DROP COLUMN ride\_length\_time,

DROP COLUMN seconds\_difference;

This code gets the total time of the rides into a table that we can save in big query for later:

SELECT \*, DATE\_DIFF(ended\_at , started\_at, SECOND) AS seconds\_difference

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data`;

This code gets us the average number of seconds the members took on the bike ride:

SELECT AVG(seconds\_difference)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE member\_casual = "member";

= 664.502309956885 Seconds

= 11.083333333333334 Minutes

This code gets us the average number of seconds the casual riders took on the bike ride:

SELECT AVG(seconds\_difference)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE member\_casual = "casual";

= 1912.3095874252874 Seconds

= 31.866666666666667 Minutes

This is code to get the number of unique stations that members start at:

SELECT COUNT(DISTINCT(start\_station\_name))

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE member\_casual = "member";

= 770

This is code to get the number of unique stations that members end at:

SELECT COUNT(DISTINCT(end\_station\_name))

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE member\_casual = "member";

= 766

This gets us rides of members go in a loop (they end where they start):

SELECT COUNT(\*)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE start\_station\_name = end\_station\_name AND member\_casual = "member";

= 109,009

Total rides of members (loops and non-loops):

SELECT COUNT(\*)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE member\_casual = "member"

= 2,777,974

Total Percentage of Members Looping- 3.924%

This is code to get the number of unique stations that casuals start at:

SELECT COUNT(DISTINCT(start\_station\_name))

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE member\_casual = "casual";

=780

This is code to get the number of unique stations that casuals end at:

SELECT COUNT(DISTINCT(end\_station\_name))

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE member\_casual = "casual";

= 780

This gets us rides where casuals go in a loop (they end where they start):

SELECT COUNT(\*)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE start\_station\_name = end\_station\_name AND member\_casual = "casual";

= 231,342

Total rides of casuals (loops and non-loops):

SELECT COUNT(\*)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_2`

WHERE member\_casual = "casual"

= 2,358,287

Total Percentage of Casual’s Looping = 9.809%

● How should you organize your data to perform analysis on it Continued…

I tried importing the Members file into RStudio, but I didn’t have enough RAM or CPU’s to do so (the file was about 500 mb in size). I split them up into smaller files (up to 500,000 results a file) to help with the importing with this code.

SELECT \*

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_Members`

LIMIT 500000 OFFSET 1000000;

Unfortunately, I was not able to combine all these uploaded files into a single file (it required more ram and CPUs than I have access to in RStudio Cloud out of what I pay for). This is the R Code that would be used to combine all 6 files in RStudio into one for the members data:

# Upload the CSV file

biking\_data\_members\_1 <- read.csv("Biking\_Data\_Members\_0.0-0.5.csv")

# Upload the CSV file

biking\_data\_members\_2 <- read.csv("Biking\_Data\_Members\_0.5-1.0.csv")

# Upload the CSV file

biking\_data\_members\_3 <- read.csv("Biking\_Data\_Members\_1.0-1.5.csv")

# Upload the CSV file

biking\_data\_members\_4 <- read.csv("Biking\_Data\_Members\_1.5-2.0.csv")

# Upload the CSV file

biking\_data\_members\_5 <- read.csv("Biking\_Data\_Members\_2.0-2.5.csv")

# Upload the CSV file

biking\_data\_members\_6 <- read.csv("Biking\_Data\_Members\_2.5-2.7.csv")

#combine the files into on stacked file

#combine all the biking\_data\_members

biking\_data\_members\_all <- rbind(biking\_data\_members\_1, biking\_data\_members\_2, biking\_data\_members\_3, biking\_data\_members\_4, biking\_data\_members\_5, biking\_data\_members\_6)

This is the R Code that gets the starting locations of member riders inside the city of Chicago turned into a map in RStudio:

Install.packages("tidyverse")

install.packages("reader")

install.packages("readx1")

install.packages("ggplot2")

library(usmap)

library(ggplot2)

# Upload the CSV file

biking\_data\_members\_1 <- read.csv("Biking\_Data\_Members\_0.0-0.5.csv")

df1 <- data.frame(biking\_data\_members\_1)

# get just the starting latitude and longitude from the biking\_data\_members\_1

starting\_latitude <- df1 %>%

select(start\_lat)

View(starting\_latitude)

starting\_longitude <- df1 %>%

select(start\_lng)

#Check to make sure you got the longitude

#View(starting\_longitude)

# Lat/Lon lists added to code from online

#convert the latitude and longitude to numeric from a list

starting\_longitude <- unlist(starting\_longitude)

starting\_latitude <- unlist(starting\_latitude)

#store latitude and longitude in a data frame nameed test\_data

test\_data <- data.frame(starting\_longitude, starting\_latitude)

# Check to make sure the test\_data looks good

View(test\_data)

#transform the test\_data dataframe into us map data (this will be stored

# as the data in our map function)

transformed\_data <- usmap\_transform(test\_data)

#checking to make sure transformed data looks correct

View(transformed\_data)

#pipe the starting latitude and longitude data into a function to pull out the transformed

# data points

starting\_latitude <- transformed\_data %>%

select(starting\_latitude.1)

starting\_longitude <- transformed\_data %>%

select(starting\_longitude.1)

#change the new latitude and longitude data into numeric (instead of list)

starting\_longitude <- unlist(starting\_longitude.1)

starting\_latitude <- unlist(starting\_latitude.1)

transformed\_data <- transformed\_data %>%

select(starting\_longitude.1, starting\_latitude.1)

#check transformed\_data

View(transformed\_data)

starting\_longitude <- unlist(starting\_longitude)

starting\_latitude <- unlist(starting\_latitude)

#Setting up the Libraries for the Chicago Ma

library(tidyverse)

library(ggmap)

library(RColorBrewer)

library(patchwork)

library(here)

options(digits = 3)

set.seed(1234)

theme\_set(theme\_minimal())

# store bounding box coordinates

chi\_bb <- c(

left = -87.936287,

bottom = 41.679835,

right = -87.447052,

top = 42.000835

)

chicago\_stamen <- get\_stamenmap(

bbox = chi\_bb,

zoom = 11

)

chicago\_stamen

chicago <- chicago\_stamen

ggmap(chicago)

#This shows the map of starting points inside the city of Chicago

ggmap(chicago) +

geom\_point(

data = transformed\_data,

aes(

x = starting\_longitude,

y = starting\_latitude

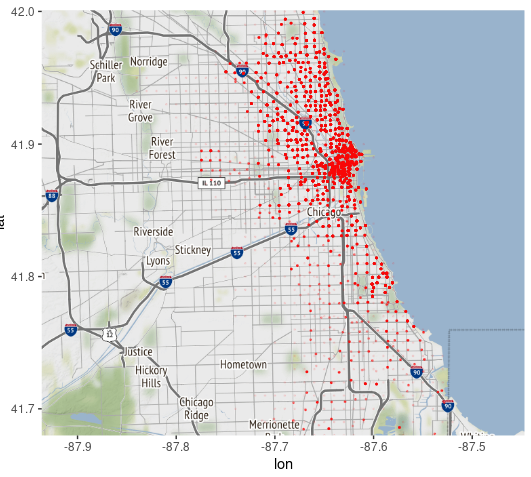
),

size = .3,

alpha = .07,

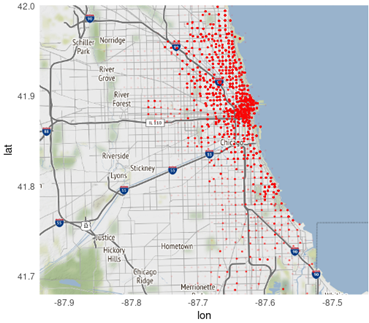
color = "red" )

This is the map that the previous code shows. According to this map it shows that a good amount of our members start riding the bikes in the heart of downtown Chicago as opposed to rural areas.



(“Divvy-Tripdata.”)

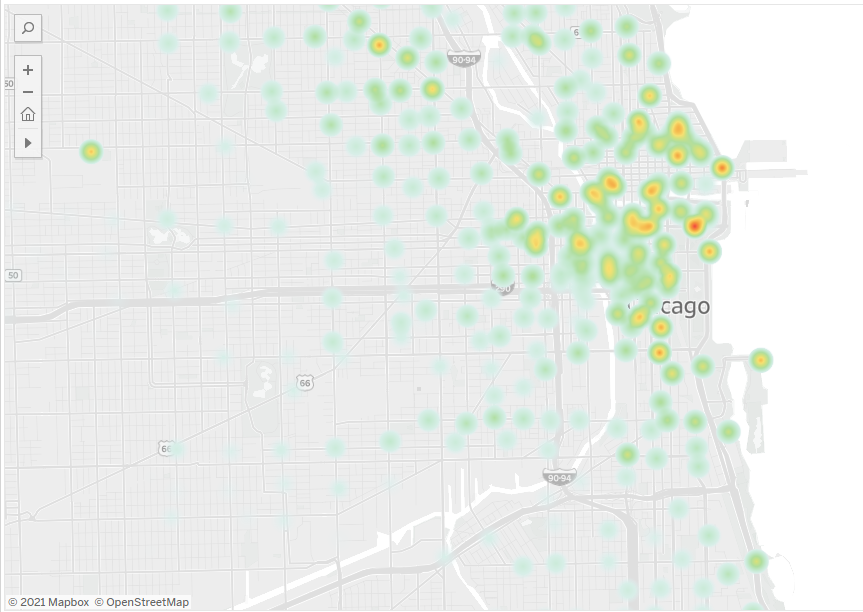
This is the ending point of the Members in Chicago. A lot of people looped back to where they started. The code for this image is very similar to the code for the starting position map right above this one (with the word “start” inside [this code](#Chicago_Map_Code) is replaced by the word “end”).



(“Divvy-Tripdata.”)

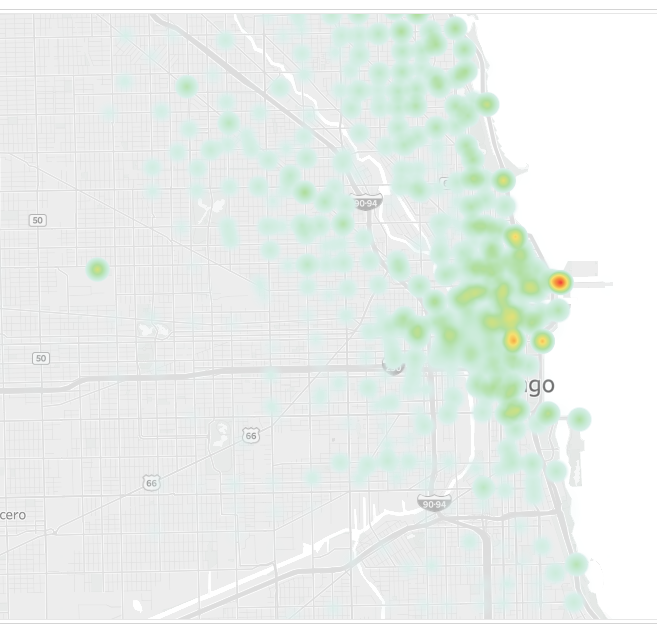
I could not get RStudio to work with Casual data for the map. It kept crashing before I could load the slightly zoomed out area (compared to the members dot data) of Chicago. I uploaded the casual’s data to Tableau Public so I could visualize it on a map.

Casuals: Starting Position (Chicago)



(“Divvy-Tripdata.”)

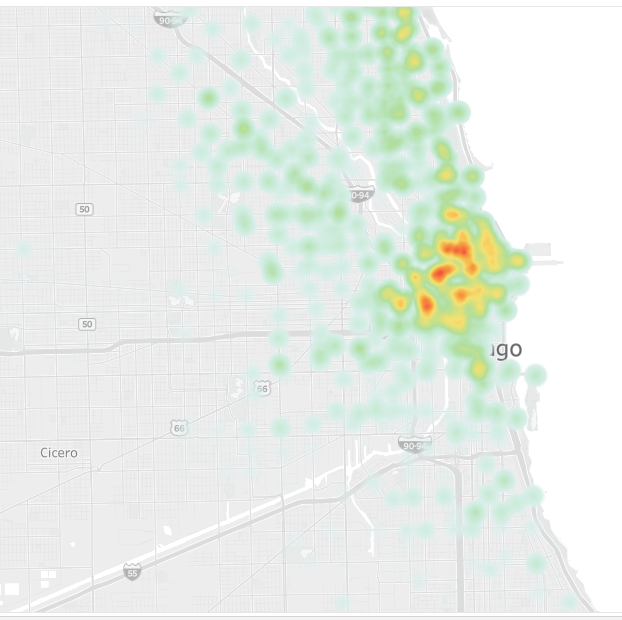
Casuals: Ending Position (Chicago)



(“Divvy-Tripdata.”)

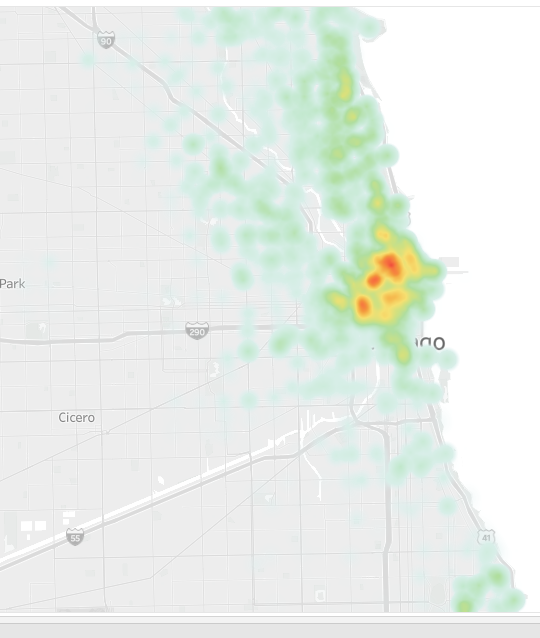
To compare to members heat data on a one to one basis (in the same program), I added the heat map for members starting here

Members: Starting Position (Chicago)



(“Divvy-Tripdata.”)

Members: Ending Position (Chicago)



(“Divvy-Tripdata.”)

This code is used for getting the total ride time by casuals:

SELECT SUM(ride\_time)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_Casuals`;

The total amount of hours of riding time by the casuals is:

1. 0 1252715:14:0

= 52196.458333333336 Days

This code is used for getting the total ride time by members:

SELECT SUM(ride\_time)

FROM `cycling-project-328223.Merged\_Data\_Set.Biking\_Data\_Members`;

The total amount of hours of riding time by the members is:

0-0 0 512769:29:0

= 21365.375 Days

● Has your data been properly formatted?

I had to add one column to allow myself to do math on the time interval between start\_at and end\_at. I could not do math on a data type interval, so I added a numeric column and calculated the time spent in seconds.

● What surprises did you discover in the data?

I discovered that a lot of members are not going on scenic routes (or starting and leaving at the same place and thus completing a loop). They are also spending much less time on their routes on average than casuals. This shows that they’re likely using the bikes to go places they need to be such as work.

● What trends or relationships did you find in the data?

A lot of bikers, regardless of whether they are members or casuals, start and end in downtown areas. There is less in rural areas. Also there are a lot more casual riders during the Summer months. There are a lot more Member riders during the Fall and Winter.

● How will these insights help answer your business questions?

These trends show where our advertisements should go and which bikes we should push in order to get more members out of our casual users.

Step 5: Share

● Were you able to answer the question of how annual members and casual riders use Cyclistic bikes differently?

A. According to Wikipedia an electric bike, also known as an **e-bike or ebik**, is a [bicycle](https://en.wikipedia.org/wiki/Bicycle) with an integrated [electric motor](https://en.wikipedia.org/wiki/Electric_motor) used to assist propulsion. Based on the stacked bar chart made in R the use of Electric Bikes and Classic Bikes rises when people shift from Casual Status to Member Status. We should advertise the use of Electric Bikes and Classic Bikes over the use of Classic Bikes when advertising to Casuals. It appears that electric bikes would be quicker than classic bikes and because our members are taking shorter rides it can be inferred that they are using electric bikes to commute to places they have to be (as opposed to places they might be sight seeing) such as work.

B. Members were averaging 11.083 minutes a ride while casuals is averaging 31.867 minutes a ride. It can be inferred that members are taking shorter rides and because they are using shorter rides are more likely to be commuting to work using the bikes. Casuals are traveling farther distances than members are.

C. 3.924% of Members are stopping in the same exact place that they are starting from while 9.809% of casuals are starting and stopping I the exact same place. If you are a casual, you are roughly 2.5 times more likely (proportion wise) to take a scenic route than a commute where you are trying to go from one location to another.

D. The number of starting vs. stopping stations, along with the heat map, shows that they are starting and stopping in roughly the same number of places (with members being only slightly less). With our heat maps we can see that these areas typically overlap and that most of the riding occurs in downtown Chicago, vs starting in more rural areas of Chicago.

E. The number of casual riders are a lot more during the Summer Months (May – September). During the Fall and Winter months, even though there is less Members riding than in the Summer, the ratio of Members to Casuals are greatest during the Fall and Winter. We should make sure the number of ads we buy is highest during the Summer months (since our Business Objective is to convert Casuals to Members) and less during the Fall and Winter Months. One explanation is that the Casuals are riding for pleasure instead of commuting (since they ride most when the weather is favorable).

● What story does your data tell?

Casuals are taking longer rides than members. They’re also more likely to come from rural areas, are 2.5 times more likely to start and stop in the same place and are less likely to use electric bikes. This shows that people who are casual are more likely to use the bike for fun while people who are members are more likely to be using the bike to get to places they have to be. Also the total number of ride times is higher for casuals (along with their higher average) meaning they’re taking longer routes than the members are.

● How do your findings relate to your original question?

They show why a person would use the bike as a casual vs. why they would use the bike as a member.

● Who is your audience? What is the best way to communicate with them?

Our audience for the presentation is the marketing team and the head of analytics. It would help to show the numbers themselves. It would not be quite as easy to explain the steps used to clean the data with the head of marketing, however the analytics team would want to see the SQL I used as well.

● Can data visualization help you share your findings?

The data visualization map can help show where casuals and members are starting and finishing inside the city of Chicago. The rest of the data is a little less visual (since it was performed; however it is done in Big Query). This would not stop a data analyst from understanding it.

● Is your presentation accessible to your audience?

I need to make use of more visuals to explain the data. Using SQL steps would definitely be a step for data analysts only and would not be accessible to people with non-technical backgrounds. A dashboard would look similar to this-

Dashboard for Presentation

Chart, bar chart

Description automatically generated

(“Divvy-Tripdata.”)

Step 6: Act

1. How do annual members and casual riders use Cyclistic bikes differently?

Members take quicker trips than causal riders and are less likely to go in a loop than casual riders. It is more likely that they are using the bike to commute to places they must go to like work and they prefer the quicker electric bike more than casuals (likely for the speed). Casuals also are more likely to ride when the weather is better (during the Summer and Spring months).

2. Why would casual riders buy Cyclistic annual memberships?

If they make it a part of their commute, then they’ll be more likely to buy a membership. Casuals are more likely to use the bikes as a part of traversing the city just, possibly for a scenic route.

3. How can Cyclistic use digital media to influence casual riders to become members?

Cyclistic can put emphasis on advertising the electric bike and the classic bike (they’re more proportionally for members than casuals). Also advertising the speed of the electric bikes would be suggested as well. The ads should target highly dense areas (such as downtown Chicago) instead of more rural areas. People are more likely to use the bike when downtown as opposed to rural areas of Chicago so it would make more sense to put advertisements in physical places downtown instead of solely just advertising through digital media. Also they should buy the most ads during the summer because of the higher amounts of Casual riders between the months of May and September. They should scale back ads during the fall and winter (since more Members than Casuals ride at that time).

● What is your final conclusion based on your analysis?

Advertise speed and accessibility of the bikes (especially the electric bikes) as opposed to other forms of mass transportation such as buses to convince Casuals to become members.

● How could your team and business apply your insights?

They could pay for more advertisements in physical locations downtown to boost performance of memberships. Also, prioritize location based ads for digital media

● What next steps would you or your stakeholders take based on your findings?

Buy more electric bikes for casuals to use and become members and focus more heavily on downtown areas. Look into more physical forms of advertisement or location-based advertisements since there is a large difference between heavily dense areas and more lightly populated areas for member and Causal riders.

● Is there additional data you could use to expand on your findings?

Getting the routes themselves that bikers took would be helpful. This could be used to determine where physical advertisements would be most effective to turn casuals into members.

Citation(s)

(MLA)

Cyclistic. “Divvy-Tripdata.” *Bucket Loading...*, https://divvy-tripdata.s3.amazonaws.com/index.html.