STRAVA FITNESS – DATA ANALYTICS

Task:

Analyze FitBit Fitness Tracker Data to gain insights into how consumers are using the FitBit app and discover trends and insights for Bellabeat marketing strategy.

Objectives:

1. What are the trends identified?
2. How could these trends apply to Bellabeat customers?
3. How could these trends help influence Bellabeat marketing strategy?

Deliverables:

1. A clear summary of the task
2. A description of all data sources used
3. Documentation of any cleaning or manipulation of data
4. A summary of analysis
5. Supporting visualizations and key findings
6. High-level content recommendations based on the analysis

Information on Data Source:

1. The dataset and stored in 18 csv files.
2. Data collected includes (1) physical activity recorded in minutes, (2) heart rate, (3) sleep monitoring, (4) daily activity and (5) steps.

PROCESS:

We are using SQL, Python, Power Bi

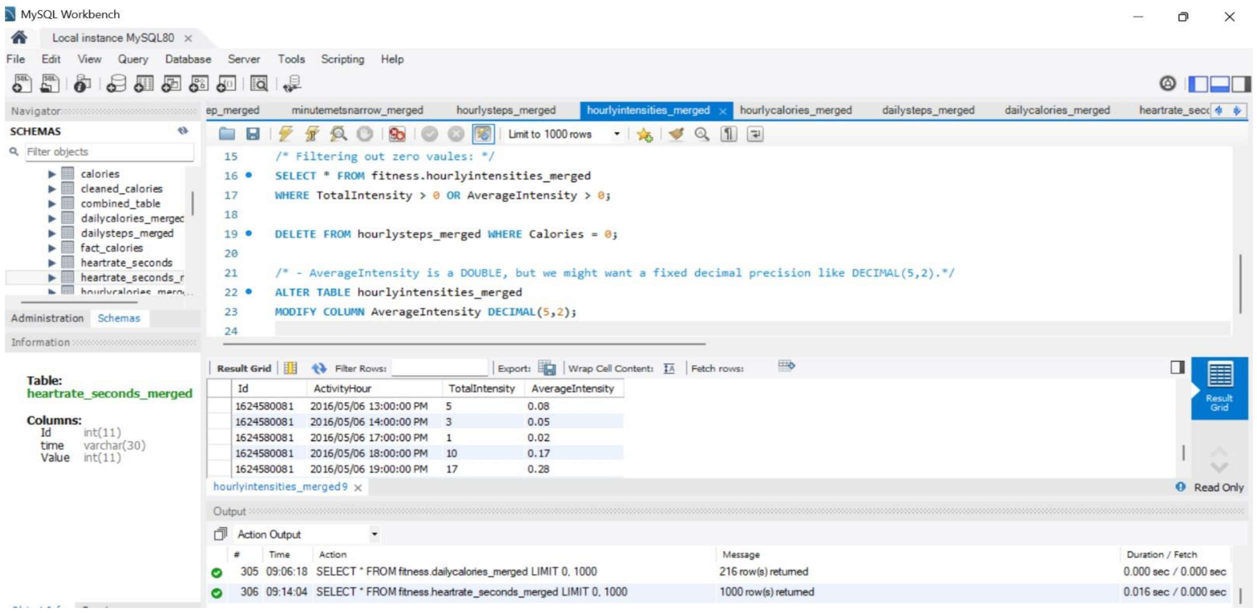
1. SQL for clean the data
2. Python for analysis the data
3. Power Bi for Dashboard

SQL Process:

Cleaning Summary:

1. Filtering out zero values
2. If it is a Double in row's values, but we might want a fixed decimal precision Like decimal(5,2).
3. Modify the query to filter out duplicates and select distinct values.
4. If want the highest values per Id,modify with Max():
5. Unique values while filtering data up to and use Distinct – ensures unique rows , eliminating duplicates.

Figure:



```
/* Filtering out zero vaules: */
SELECT * FROM fitness.hourlyintensities_merged
WHERE TotalIntensity > 0 OR AverageIntensity > 0;

DELETE FROM hourlysteps_merged WHERE Calories = 0;

/* - AverageIntensity is a DOUBLE, but we might want a fixed decimal precision like DECIMAL(5,2).*/
ALTER TABLE hourlyintensities_merged
MODIFY COLUMN AverageIntensity DECIMAL(5,2);
```

Id	ActivityHour	TotalIntensity	AverageIntensity
1624580081	2016/05/06 13:00:00 PM	5	0.08
1624580081	2016/05/06 14:00:00 PM	3	0.05
1624580081	2016/05/06 17:00:00 PM	1	0.02
1624580081	2016/05/06 18:00:00 PM	10	0.17
1624580081	2016/05/06 19:00:00 PM	17	0.28

Table: heartrate_seconds_merged

Columns:
Id int(11)
time varchar(30)
Value int(11)

```
#    Time      Action                                                 Message                Duration / Fetch
305  09:06:18  SELECT * FROM fitness.dailycalories_merged LIMIT 0, 1000       216 row(s) returned    0.000 sec / 0.000 sec
306  09:14:04  SELECT * FROM fitness.heartrate_seconds_merged LIMIT 0, 1000   1000 row(s) returned   0.016 sec / 0.000 sec
```

Python Process :

We are using Python to prepare and process the data.

1. Preparing the Environment :The numPy, pandas, matplotlib, datetime packages are installed and aliased for easy reading.

In [1]: # import packages

import numpy as np *# data arrays*

import pandas as pd *# data structure and data analysis*

import matplotlib as plt *# data visualization*

import datetime as dt *# date time*

2.Importing data set: Reading in the selected file.

In [2]: *# read_csv function to read the required CSV file*

daily_activity = pd.read_csv("../input/fitbit/Fitabase /dailyActivity_merged.csv")

3. Data cleaning and manipulation:

   1. Observe and familiarize with data

   2. Check for null or missing values

In [3]: # preview first 10 rows with all columns

```
daily_activity.head(30)
```

4.finding out whether there is any null or missing values in daily_activity.

In [4]: # obtain the # of missing data points per column

missing_values_count = daily_activity.isnull().sum()

# look at the # of missing points in all columns

missing_values_count[:]

In[5] : # show basic information of data

daily_activity.info()

6.Counting the unique ID and to confirm whether data set has 30 IDs.

In [6]: # count distinct value of "Id"

unique_id = len(pd.unique(daily_activity["Id"]))

print("# of unique Id: " + str(unique_id))

From the above observation, noted that

1. There is no typo, Null or missing values.
2. Data frame has 940 rows and 15 columns.
3. *ActivityDate* is wrongly classified as object dtype and has to be converted to datetime64 dtype.
4. There are 33 unique IDs, instead of 30 unique IDs as expected from 30 fitness tracker users.

8.Creating new list with rearranged column names and renaming daily_activity to a shorter name df_activity.

In [8]: #r create new list of rearranged columns

new_cols = ['Id', 'ActivityDate', 'DayOfTheWeek', 'TotalSteps', 'TotalDistance', 'TrackerDistance', 'LoggedActivitiesDistance', 'VeryActiveDistance', 'ModeratelyActiveDistance', 'LightActiveDistance', 'SedentaryActiveDistance', 'VeryActiveMinutes', 'FairlyActiveMinutes', 'LightlyActiveMinutes', 'SedentaryMinutes', 'TotalExerciseMinutes', 'TotalExerciseHours', 'Calories']

# reindex function to rearrange columns based on "new_cols"

df_activity = daily_activity.reindex(columns=new_cols)

# print 1st 30 rows to confirm

df_activity.head(30)

9. Creating new column by separating the date into day of the week for further analysis.

In [9]:

# create new column "day_of_the_week" to represent day of the week

df_activity["DayOfTheWeek"] = df_activity["ActivityDate"].dt.day_name()

# print 1st 30 rows to confirm

df_activity["DayOfTheWeek"].head(30)

Rearranging and renaming columns from XxxYyy to xxx_yyy.

In [10]: *# rename columns*

df_activity.rename(columns = {"Id":"id", "ActivityDate":"date", "DayOfTheWeek":"day_of_the_week", "TotalSteps":"total_steps", "TotalDistance":"total_dist", "TrackerDistance":"track_dist", "LoggedActivitiesDistance":"logged_dist", "VeryActiveDistance":"very_active_dist", "ModeratelyActiveDistance":"moderate_active_dist", "LightActiveDistance":"light_active_dist", "SedentaryActiveDistance":"sedentary_active_dist", "VeryActiveMinutes":"very_active_mins", "FairlyActiveMinutes":"fairly_active_mins", "LightlyActiveMinutes":"lightly_active_mins", "SedentaryMinutes":"sedentary_mins", "TotalExerciseMinutes":"total_mins","TotalExerciseHours":"total_hours","Calories":"calories"}, inplace = **True**)

# print column names to confirm

print(df_activity.columns.values)

df_activity.head(30)

11. Creating new column *total_mins* being the sum of total time logged.

In [11]:

# create new column "total_mins" containing sum of total minutes.

df_activity["total_mins"] = df_activity["very_active_mins"] + df_activity["fairly_active_mins"] + df_activity["lightly_active_mins"] + df_activity["sedentary_mins"]

df_activity["total_mins"].head(30)

12. Creating new column by converting *total_mins* to number of hours.

In [12]:

# create new column *total_hours* by converting to hour and round float to two decimal places

df_activity["total_hours"] = round(df_activity["total_mins"] **/** 60)

# print 1st 5 rows to confirm

df_activity["total_hours"].head(30)

ANALYZE

1. Perform calculations

Pulling the statistics of df_activity for analysis:

- count - no. of rows
- mean (average)
- std (standard deviation)
- min and max
- percentiles 25%, 50%, 75%

In [13]: # pull general statistics

```
df_activity.describe()
```

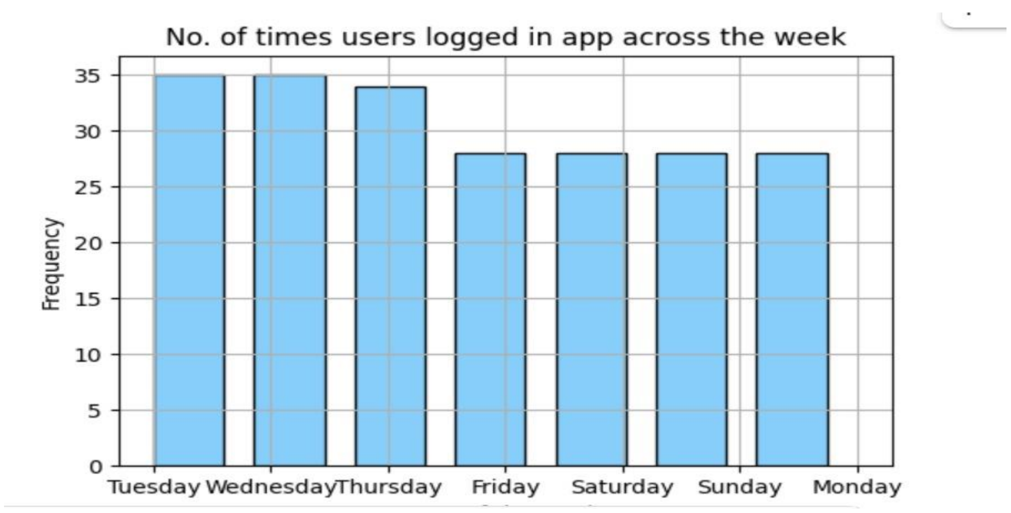
2.Data Visualisation and Findings

In [14]:

```
# import matplotlib package
import matplotlib.pyplot as plt

# plotting histogram
plt.style.use("default")
plt.figure(figsize=(6,4)) # specify size of the chart
plt.hist(df_activity.day_of_the_week, bins = 7,
        width = 0.6, color = "lightskyblue", edgecolor = "black")

# adding annotations and visuals
plt.xlabel("Day of the week")
plt.ylabel("Frequency")
plt.title("No. of times users logged in app across the week")
plt.grid(True)
plt.show()
```



Frequency of usage across the week

In this histogram, we are looking at the frequency of FitBit app usage in terms of days of the week.

1. We discovered that users prefer or remember (giving them the doubt of benefit that they forgotten) to track their activity on the app during midweek from Tuesday to Friday.

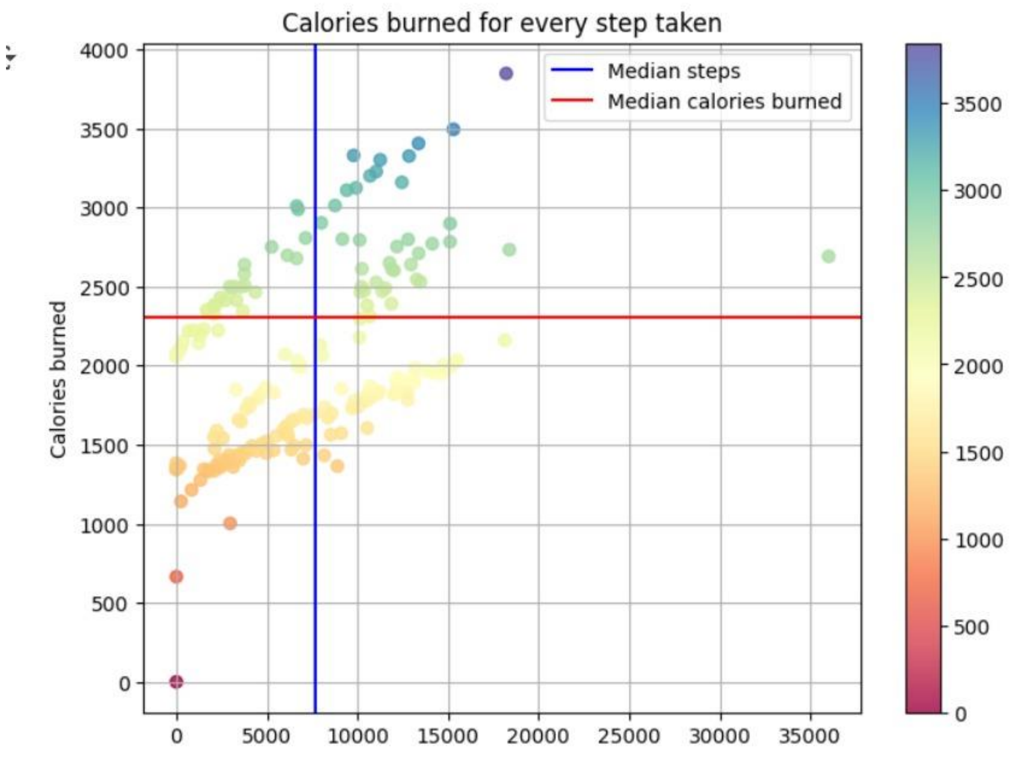
In [15]:

```
# import matplotlib package
import matplotlib.pyplot as plt

# plotting scatter plot
plt.style.use("default")
plt.figure(figsize=(8,6)) # specify size of the chart
plt.scatter(df_activity.total_steps, df_activity.calories,
        alpha = 0.8, c = df_activity.calories,
        cmap = "Spectral")

# add annotations and visuals
median_calories = 2303
median_steps = 7637

plt.colorbar(orientation = "vertical")
plt.axvline(median_steps, color = "Blue", label = "Median steps")
plt.axhline(median_calories, color = "Red", label = "Median calories burned")
plt.xlabel("Steps taken")
plt.ylabel("Calories burned")
plt.title("Calories burned for every step taken")
plt.grid(True)
plt.legend()
plt.show()
```



Calories burned for every step taken

From the scatter plot, we discovered that:

1. We observed that intensity of calories burned increase when users are at the range of > 0 to 15,000 steps with calories burn rate cooling down from 15,000 steps onwards.
2. Noted a few outliers:
    - Zero steps with zero to minimal calories burned.
    - 1 observation of > 35,000 steps with < 3,000 calories burned.

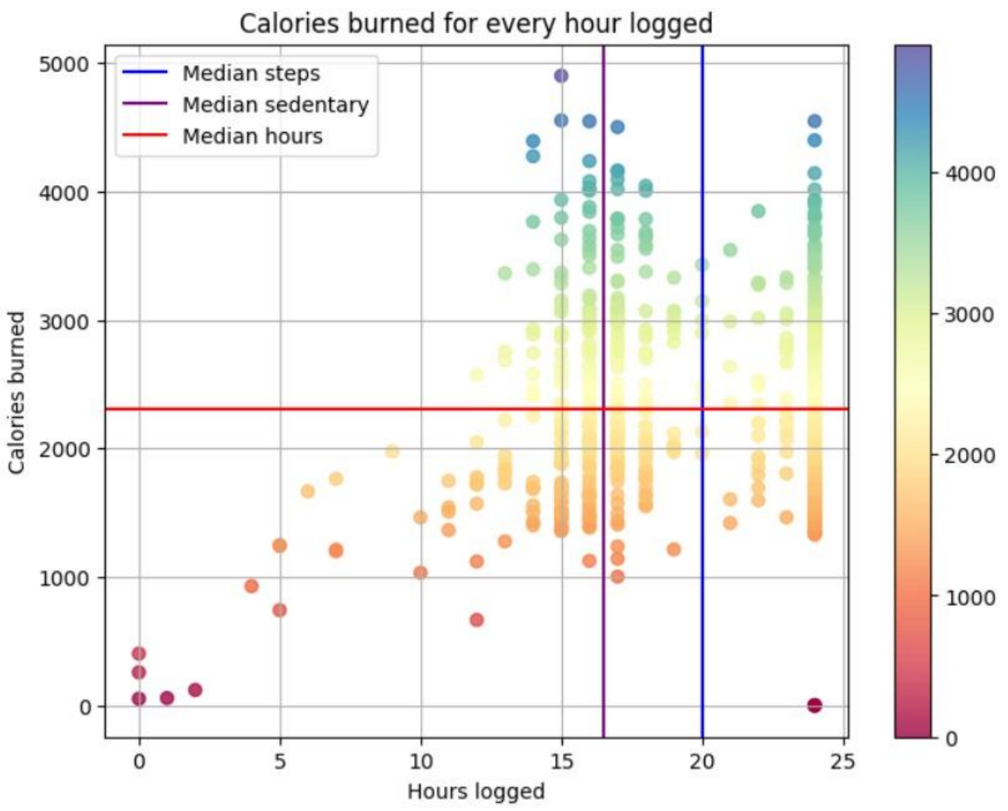
- Deduced that outliers could be due to natural variation of data, change in user's usage or errors in data collection

```
# import matplotlib package
import matplotlib.pyplot as plt

# plotting scatter plot
plt.style.use("default")
plt.figure(figsize=(8,6)) # Specify size of the chart
plt.scatter(df_activity.total_hours, df_activity.calories,
        alpha = 0.8, c = df_activity.calories,
            cmap = "Spectral")

# adding annotations and visuals
median_calories = 2303
median_hours = 20
median_sedentary = 991 / 60

plt.colorbar(orientation = "vertical")
plt.axvline(median_hours, color = "Blue", label = "Median steps")
plt.axvline(median_sedentary, color = "Purple", label = "Median sedentary")
plt.axhline(median_calories, color = "Red", label = "Median hours")
plt.xlabel("Hours logged")
plt.ylabel("Calories burned")
plt.title("Calories burned for every hour logged")
plt.legend()
plt.grid(True)
plt.show()
```



Calories burned for every hour logged

The scatter plot is showing:

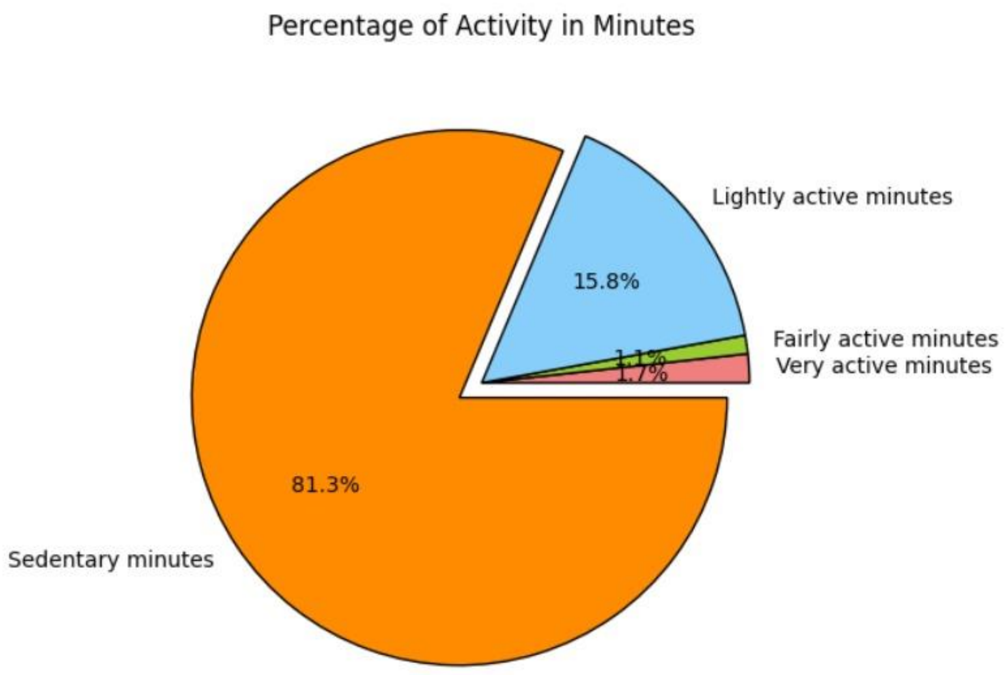
1. A weak positive correlation whereby the increase of hours logged does not translate to more calories being burned. That is largely due to the average sedentary hours (purple line) plotted at the 16 to 17 hours range.
2. Again, we can see a few outliers:
    - The same zero value outliers
    - An unusual red dot at the 24 hours with zero calorie burned which may be due to the same reasons as above.

```
# import packages
import matplotlib.pyplot as plt
import numpy as np

# calculating total of individual minutes column
very_active_mins = df_activity["very_active_mins"].sum()
fairly_active_mins = df_activity["fairly_active_mins"].sum()
lightly_active_mins = df_activity["lightly_active_mins"].sum()
sedentary_mins = df_activity["sedentary_mins"].sum()

# plotting pie chart
```

```
slices = [very_active_mins, fairly_active_mins, lightly_active_mins, sedentary_mins]
labels = ["Very active minutes", "Fairly active minutes", "Lightly active minutes", "Sedentary
minutes"]
colours = ["lightcoral", "yellowgreen", "lightskyblue", "darkorange"]
explode = [0, 0, 0, 0.1]
plt.style.use("default")
plt.pie(slices, labels = labels,
    colors = colours, wedgeprops = {"edgecolor": "black"},
    explode = explode, autopct = "%1.1f%%")
plt.title("Percentage of Activity in Minutes")
plt.tight_layout()
plt.show()
```



Percentage of Activity in Minutes
As seen from the pie chart,

1. Sedentary minutes takes the biggest slice at 81.3%.
2. This indicates that users are using the FitBit app to log daily activities such as daily commute, inactive movements (moving from one spot to another) or running errands.
3. App is rarely being used to track fitness (ie. running) as per the minor percentage of fairly active activity (1.1%) and very active activity (1.7%). This is highly discouraging as FitBit app was developed to encourage fitness.

In the final step, we will be delivering our insights and providing recommendations based on our analysis.

POWER BI Process:

Summary:

Transforms raw fitness data into actionable insights, tracking movement, sleep, calorie burn, and weight trends across multiple users.

Fig 1:

From analysis showing in Dashboard on highlighted patterns like peak performers, consistent activity periods, and standout effort by certain user Ids

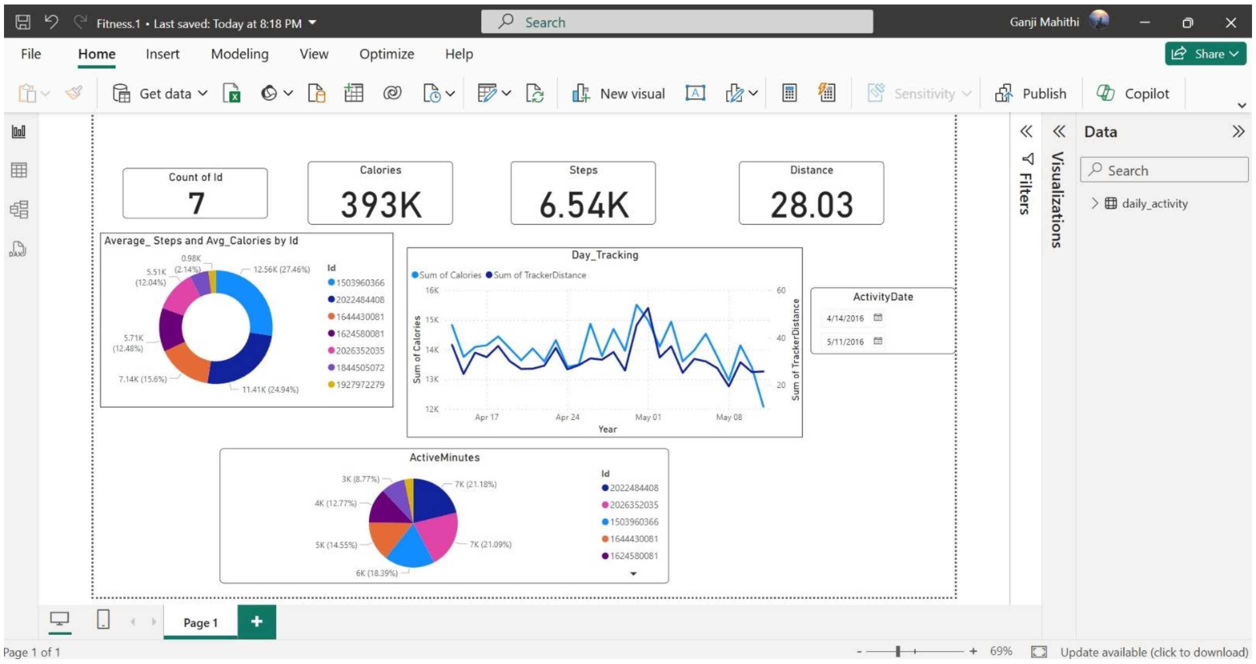
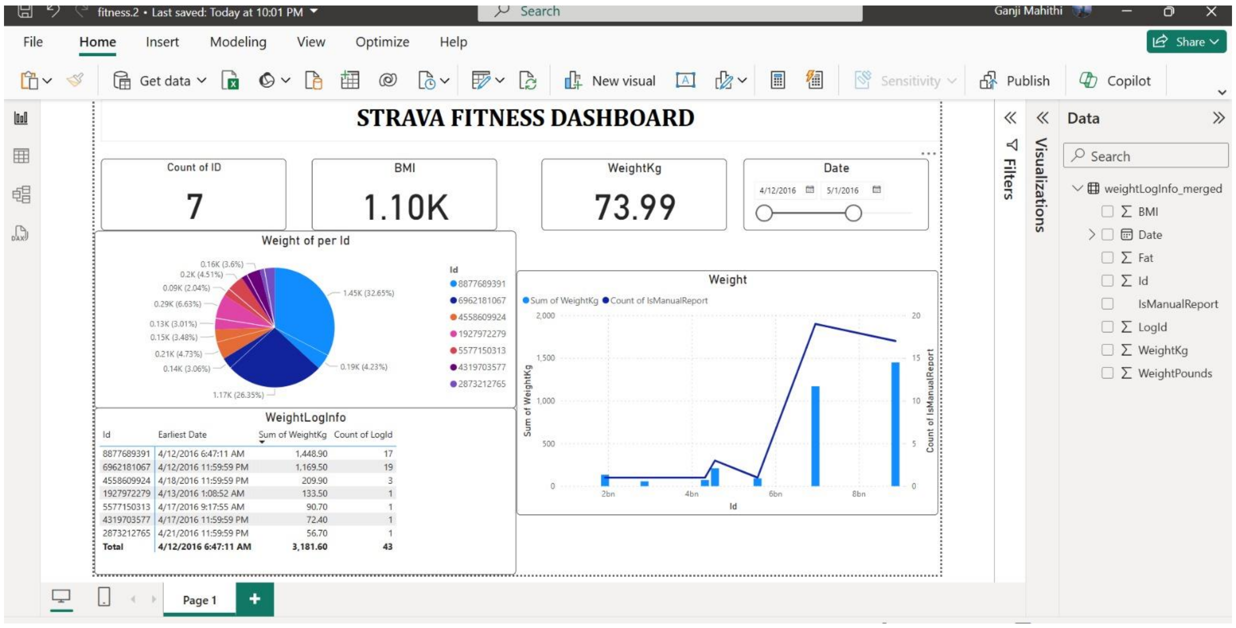


Fig 2:

About the users log their weight over time, spotlighting active loggers and trends.



WeightLogInfo

Id	Earliest Date	Sum of WeightKg	Count of LogId
8877689391	4/12/2016 6:47:11 AM	1,448.90	17
6962181067	4/12/2016 11:59:59 PM	1,169.50	19
4558609924	4/18/2016 11:59:59 PM	209.90	3
1927972279	4/13/2016 1:08:52 AM	133.50	1
5577150313	4/17/2016 9:17:55 AM	90.70	1
4319703577	4/17/2016 11:59:59 PM	72.40	1
2873212765	4/21/2016 11:59:59 PM	56.70	1
Total	**4/12/2016 6:47:11 AM**	**3,181.60**	**43**

Conclusion: As per Bellabeat stakeholder recommendation,Fitbit user open data has been analyzed to identify patterns of fitness device usage. Although the data set lacked some information , as per analysis .