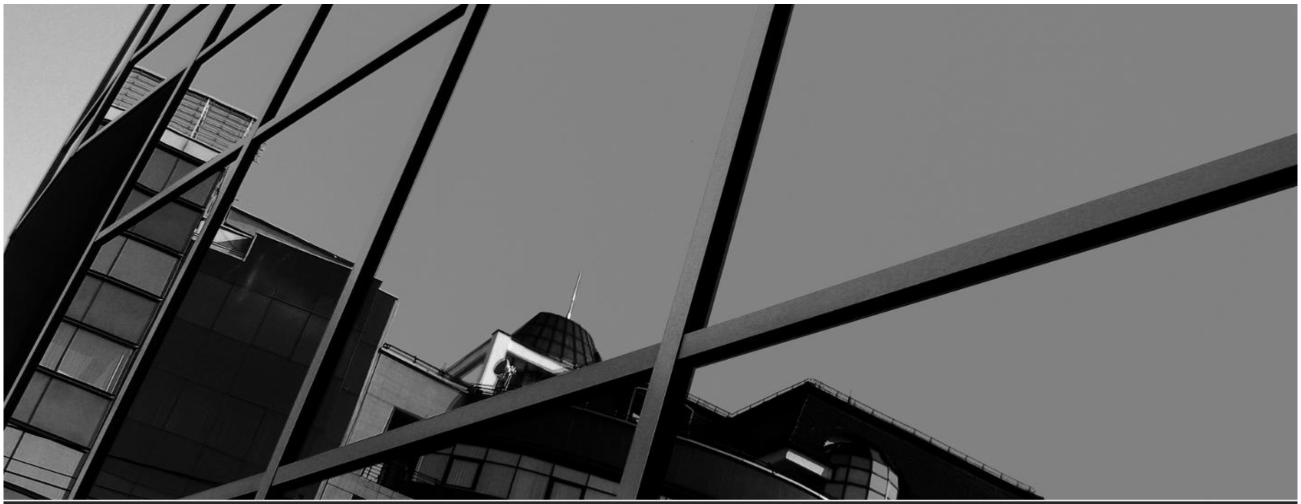


CASE STUDY

BELLABEAT

PREPARED BY DEEPINDER RANDHAWA

2024

Table of Contents

- Summary Statistics
    - Calculating Average Steps per Day
    - Calculating Average Sleep Hours
    - Calculating Max Steps per Day
    - Calculating Max Sleep Hour
- Feature Engineering
    - Adding Day of Week
    - User Categorization on Activity Level
    - User Categorization on Sleep Duration

10. OBSERVATIONS AND VISUALIZATION

- Steps Taken Over Time
- Calories Burnt Over Time
- Steps Taken by Various User Groups
- Calories Burnt by Various User Groups
- Time Spent in Bed vs Time Asleep
- Sleep Pattern Across Week
- Calories Burnt vs Total Minutes Sleep
- Calories Burnt vs Total Steps
- Calories Burnt vs Very Active Minutes
- Calories Burnt vs Fairly Active Minutes
- Calories Burnt vs Lightly Active Minutes
- Calories Burnt vs Sedentary Minutes

11. Insights and Findings

- User Activity Levels
- Sedentary Behavior
- Calories Burned
- Sleep Patterns
- Weight and BMI
- Analysis of Correlations

12. SHARE Phase

13. ACT Phase

- Strategic Marketing Initiatives

Company Overview

BellaBeat, founded in 2013, is a tech-driven women's wellness brand that offers smart wearables, apps, and personalized wellness programs to support women's health. Under the visionary leadership of Urška Sršen, co-founder and Chief Creative Officer, BellaBeat leverages smart device fitness data to create personalized health insights and drive growth. Their mission is to empower women with tools and knowledge to take control of their health.

Case Study Objectives

Data Preparation: Clean, transform, and prepare the data using SQL.
Data Analysis: Analyze the cleaned data to extract meaningful insights.
Data Visualization: Present the analysis results through interactive Power BI dashboards.
Share: Share the finding that help business to make intelligent decisions

Tools and Technologies

SQL: For data cleaning, transformation, and analysis.
Power BI: For creating interactive visualizations and dashboards.

Scenario Assumption

As a data analyst on the marketing analyst team at Bellabeat, your task is to analyze smart device data to gain insights into consumer behavior. These insights will inform Bellabeat's marketing strategy, aiming to expand its global presence. Findings will be presented to the Bellabeat executive team with strategic recommendations.

Key Steps in the Data Analysis:

1.Ask
2.Prepare
3.Process
4.Analyze
5.Share
6. Act

1. ASK

Recognize the Current Problem: Understand the challenges Bellabeat faces in the competitive smart device market.
Define the Business Task: Analyze smart device data to uncover consumer insights and identify trends for the Bellabeat App marketing strategy.
Consider Key Stakeholders: Urška Sršen, Sando Mur, and the Bellabeat Marketing Analytics Team.

Identify the Impact: Enable stakeholders to understand consumer behavior, identify trends, and make informed decisions to enhance Bellabeat's marketing strategy and product development.
Guiding Questions:

- What are some trends in smart device usage?
- How can these trends be applied to Bellabeat customers?
- How can these trends influence Bellabeat's marketing strategy?

2. PREPARE

Selection of Technology: Utilizing SQL (MySQL) for data storage and querying, combined with Power BI for visualization and reporting, offers a robust and effective approach to analyzing smart device data.

Why SQL (MySQL)
Data Integrity: MySQL maintains robust data integrity, ensuring accuracy and reliability.
Extensive SQL Support: Allows execution of complex queries and efficient data manipulation.
Scalability: Effectively handles large datasets, suitable for analyzing extensive smart device data.
Performance: Optimized for high performance, ensuring fast query execution for timely data analysis.

Why Power BI
Powerful Visualization and Reporting: Creates insightful and interactive visualizations, making complex data accessible and understandable.
Real-Time Data Insights: Supports real-time data updates, enabling timely decision-making and strategic adjustments.
Integration Capabilities: Seamlessly integrates with MySQL and other data sources for smooth data extraction, transformation, and loading (ETL).
Customizable Dashboards: Allows creation of customizable dashboards tailored to specific business needs, providing a comprehensive view of key performance indicators (KPIs) and metrics.

Selection Of Target Data: We will utilize public data that examines the daily habits of smart device users. This Kaggle dataset contains personal tracker data from Fitbit users who have consented to submit their data. Public dataset providing minute-level activity for physical activity, sleep monitoring and weight data from Fitbit users.
Note: Bellabeat's co-founder has suggested that the dataset has some limitations as it is not representative of the entire population. The dataset may have sampling biases.

We are going to explore the Fitbit Fitness Tracker data across ten CSV files. The data is organized in rows and columns. They are broadly classified based on time tracked into:

- Daily data
- Hourly data

Analyzing the data on daily and hourly level can give us insights into customer trends that can be applied to Bellabeat customers.
The following CSV files were used for analysis:

dailyActivity_merged	2024-06-19 5:12 PM	Microsoft Excel Co...	109 KB
dailyCalories_merged	2024-06-19 5:12 PM	Microsoft Excel Co...	25 KB
dailyIntensities_merged	2024-06-19 5:12 PM	Microsoft Excel Co...	69 KB
dailySteps_merged	2024-06-19 5:13 PM	Microsoft Excel Co...	25 KB
hourlyCalories_merged	2024-06-19 5:13 PM	Microsoft Excel Co...	783 KB
hourlyIntensities_merged	2024-06-19 5:13 PM	Microsoft Excel Co...	878 KB
hourlySteps_merged	2024-06-19 5:13 PM	Microsoft Excel Co...	778 KB
sleepDay_merged	2024-06-19 5:14 PM	Microsoft Excel Co...	18 KB
weightLogInfo_merged	2024-06-19 5:14 PM	Microsoft Excel Co...	7 KB

Import Data

We need to import the above CSV files into tables in MySQL for data analysis. Firstly, I created a database 'Project_Bella_Beat' and import all the tables. After importing, I check if all tables were imported successfully.

Reason: To confirm that all necessary tables are available in the database for further processing.

Impact: Ensures that we are working with the correct dataset and all required tables are present.

```
USE project_bella_beat;
-- Checking if all the tables imported successfully
SHOW tables;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Tables_in_project_bella_beat
dailyactivity_merged
dailyactivity_merged_new
hourlycalories_merged_new
hourlyintensities_merged
hourlyintensities_merged_new
hourlysteps_merged
hourlysteps_merged_new
master_table
sleepday_merged
sleepday_merged_new
weightloginfo_merged
weightloginfo_merged_new

STEP 3: PROCESS

The third phase of data analysis is to process the data and clean it for analysis. Data processing involves finding inaccuracies, errors, inconsistency in data and getting rid of them to have a neat data to work on that avoids any discrepancy or result skew due to "bad data". This will ensure that the results are accurate and don't affect the credibility of the analysis.

Data Cleaning: Now that we have imported the data, we need to ensure that the data within each table is clean and ready to analyze. The following steps are performed to make sure the data is consistent and clean before drawing any meaningful insights.

1. Identify and Remove Duplicates in Each Table
2. Change Column Names for standardization
3. Check and Impute Null Values
4. Standardizing Data Types
5. Checking Consistency in Date Ranges

1. Finding and Removing Duplicates

Duplicate records can distort statistical analyses and lead to incorrect conclusions. By systematically identifying and eliminating duplicates, we ensured that each data point accurately represented unique user interactions, enhancing the reliability of our findings. The same query is run on all tables and there are no duplicates found in any tables except sleepday-merged.

Purpose: Identify and remove duplicate records.
Queries Executed: Used ROW_NUMBER () to identify duplicates.
Actions Taken: Deleted duplicates to maintain data integrity.
Impact: Eliminates inaccuracies caused by duplicate data.
Usefulness: Provides a cleaner dataset for more accurate analysis.

Detect duplicates in table dailyactivity_merged

```sql
-- Duplicates in dailyactivity_merged
WITH c1 AS
(SELECT *, ROW_NUMBER () OVER(PARTITION BY id, activitydate )AS row_no FROM dailyactivity_merged)
SELECT * FROM c1
WHERE row_no>1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance	LightActiveDistance	SedentaryActiveDistance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActive

Detect duplicates in table sleepday_merged

```sql
-- Duplicates in sleepday_merged
WITH c8 AS
(SELECT *, ROW_NUMBER () OVER(PARTITION BY id, sleepday )AS row_no FROM sleepday_merged)
SELECT * FROM c8
WHERE row_no>1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	SleepDay	TotalSleepRecords	TotalMinutesAsleep	TotalTimeInBed	row_no
4388161847	5/5/2016 12:00:00 AM	1	471	495	2
4702921684	5/7/2016 12:00:00 AM	1	520	543	2
8378563200	4/25/2016 12:00:00 AM	1	388	402	2

Removing duplicates: To address the duplicates in the sleepday_merged table, we will create a new table and delete the duplicate entries. This approach ensures that our original data remains intact for reference while providing a clean and accurate dataset for analysis. By working with a new table, we can safeguard data integrity and ensure that our analysis is based on unique and reliable data entries, leading to more precise insights into users' sleep patterns.

Create copy of table sleepday_merged

```
-- Removing duplicates: Create copy of table and remove duplicates
CREATE TABLE Sleepday_merged_new (
    Id bigint default null,
    SleepDay Text,
  TotalSleepRecords int default null,
TotalMinutesAsleep  int default null,
TotalTimeInBed  int default null,
row_num int
)ENGINE= InnoDB DEFAULT CHARSET= utf8mb4 COLLATE =utf8mb4_0900_ai_ci;
SELECT * FROM Sleepday_merged_new;

INSERT INTO sleepday_merged_new
SELECT * ,ROW_NUMBER() OVER(PARTITION BY id, sleepday )AS row_no
FROM sleepday_merged;
SELECT * FROM sleepday_merged_new;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	SleepDay	TotalSleepRecords	TotalMinutesAsleep	TotalTimeInBed	row_num
1503960366	4/28/2016 12:00:00 AM	1	366	393	1
1503960366	4/29/2016 12:00:00 AM	1	341	354	1
1503960366	4/30/2016 12:00:00 AM	1	404	425	1

Removing duplicates from table sleepday_merged

```
DELETE FROM sleepday_merged_new
WHERE row_num> 1;
COMMIT;
SELECT * FROM sleepday_merged_new
WHERE row_num>1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	SleepDay	TotalSleepRecords	TotalMinutesAsleep	TotalTimeInBed	row_num

2. Change Column Names for standardization

Ambiguous or cryptic column names can impede data interpretation and analysis. By standardizing column names to be descriptive and intuitive, we facilitated smoother data operations and minimized the risk of misinterpretation during analysis processes. All the other *column names in all other tables are quite clear except one in weightloginfo_merged that has column name **Date**.*

Purpose: Ensure clear and consistent column names.
Queries Executed: Renamed ambiguous column names.
Actions Taken: Change name of columns to increase readability.
Impact: Enhances data manipulation and analysis.
Usefulness: Reduces errors due to ambiguous column names.

Renaming column name in weightloginfo_merged

```
-- Changing column_name "DATE" to "Weight_date"
ALTER TABLE  weightloginfo_merged
CHANGE Date WeightDate text;
COMMIT;
SELECT * from weightloginfo_merged;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	WeightDate	WeightKg	WeightPounds	Fat	BMI	IsManualReport	LogId
1503960366	5/2/2016 11:59:59 PM	52.5999984741211	115.963146545323	22	22.6499996185303	True	1462233599000
1503960366	5/3/2016 11:59:59 PM	52.5999984741211	115.963146545323		22.6499996185303	True	1462319999000
1927972279	4/13/2016 1:08:52 AM	133.5	294.317120016975		47.5400009155273	False	1460509732000
2873212765	4/21/2016 11:59:59 PM	56.7000007629395	125.002104340889		21.4500007629395	True	1461283199000

3. Checking and Imputing Null Values

Missing data points can skew statistical analyses and compromise the validity of insights drawn. By systematically identifying and addressing null values through appropriate imputation techniques, we preserved data completeness and enhanced the reliability of our analytical outcomes. The same query is run on all tables and there are no null values found in any tables.

Purpose: Identify and handle missing values.
Queries Executed: Checked records with null values for Id or ActivityDate.
Actions Taken: Imputed missing data where appropriate.
Impact: Reduces data loss and maintains dataset completeness.
Usefulness: Ensures more robust and comprehensive analysis

Check null values in table dailyactivity_merged

```
-- Checking null values in dailyactivity_merged
select * from dailyactivity_merged
where Id is null or ActivityDate is null;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance

4. Standardizing Data Types

*Inconsistent or incorrect data types can lead to computational errors or hinder data integration processes. By standardizing data types and creating dedicated tables, we optimized data handling processes and facilitated seamless data interactions across various analytical tools and platforms. Across all the tables, the data type for 'DATE' column is **TEXT** so it was changed to **DATE.** The same query is executed for all the tables to change data type*

Purpose: Standardize data types for accurate analysis. The main issue in all the tables is data type of date field that is in text, so creating new tables with data type date.
Queries Executed: Created new tables with correct data types.
Actions Taken: Ensured accurate calculations and analysis.
Impact: Prevents data type-related errors.
Usefulness: Improves data processing efficiency.

Create new table dailyactivity_merged_new

```sql
-- Ensuring appropriate Datatypes for all columns of tables
DESCRIBE dailyactivity_merged;
CREATE TABLE dailyactivity_merged_new (
  Id bigint default null,  ActivityDate Date,  TotalSteps int default null,  TotalDistance double default null,
  TrackerDistance int default null,  LoggedActivitiesDistance int default null,  VeryActiveDistance int default null,
  ModeratelyActiveDistance int default null,  LightActiveDistance double default null,
  SedantaryActiveDistance int default null,  VeryActiveMinutes int default null,  FairlyActiveMinutes int default null,
  LightlyActiveMinutes int default null,  SedantaryMinutes int default null,
  Calories int default null)ENGINE= InnoDB DEFAULT CHARSET= utf8mb4 COLLATE =utf8mb4_0900_ai_ci;

INSERT INTO dailyactivity_merged_new
SELECT *
FROM dailyactivity_merged;
DESCRIBE dailyactivity_merged_new;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Id	bigint	YES		NULL	
ActivityDate	date	YES		NULL	
TotalSteps	int	YES		NULL	
TotalDistance	double	YES		NULL	
TrackerDistance	int	YES		NULL	
LoggedActivitiesDistance	int	YES		NULL	
VeryActiveDistance	int	YES		NULL	
ModeratelyActiveDistance	int	YES		NULL	
LightActiveDistance	double	YES		NULL	
SedantaryActiveDistance	int	YES		NULL	
VeryActiveMinutes	int	YES		NULL	
FairlyActiveMinutes	int	YES		NULL	
LightlyActiveMinutes	int	YES		NULL	
SedantaryMinutes	int	YES		NULL	
Calories	int	YES		NULL	

5. Checking for Consistency in Date Ranges

Discrepancies in date ranges can compromise the validity of time-sensitive analyses. By verifying and aligning date ranges across datasets, we facilitated precise temporal comparisons and trend analyses, enabling deeper insights into consumer behavior patterns and market dynamics. All the tables have consistent date range as shown in the screenshot.

Purpose: Ensure consistent date ranges across datasets.
Queries Executed: Verified date ranges with MAX and MIN functions.
Actions Taken: Aligned date ranges for accurate analysis.
Impact: Facilitates accurate time-series analysis.
Usefulness: Ensures temporal consistency in data analysis.

```sql
-- Check for consistency of dates among all tables
SELECT MIN(ActivityDate) AS MinDate,   MAX(ActivityDate) AS MaxDate FROM dailyactivity_merged_new
UNION ALL
SELECT MIN(ActivityDay_new) AS MinDate, MAX(ActivityDay_new) AS MaxDate FROM dailycalories_merged_new
UNION ALL
SELECT MIN(ActivityDay_new) AS MinDate, MAX(ActivityDay_new) AS MaxDate FROM dailyintensities_merged_new
UNION ALL
SELECT MIN(ActivityDay_new) AS MinDate, MAX(ActivityDay_new) AS MaxDate FROM dailyintensities_merged_new
UNION ALL
SELECT MIN(ActivityDay_new) AS MinDate, MAX(ActivityDay_new) AS MaxDate FROM dailysteps_merged_new
UNION ALL
SELECT MIN(ActivityHour_new) AS MinDate, MAX(ActivityHour_new) AS MaxDate FROM hourlycalories_merged_new
UNION ALL
SELECT MIN(ActivityHour_new) AS MinDate, MAX(ActivityHour_new) AS MaxDate FROM hourlyintensities_merged_new
UNION ALL
SELECT MIN(ActivityHour_new) AS MinDate, MAX(ActivityHour_new) AS MaxDate FROM hourlysteps_merged_new
UNION ALL
SELECT MIN(SleepDayNew) AS MinDate, MAX(SleepDayNew) AS MaxDate FROM sleepday_merged_new
UNION ALL
SELECT MIN(WeightDate_new) AS MinDate, MAX(WeightDate_new) AS MaxDate FROM weightloginfo_merged_new;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

MinDate	MaxDate
2016-04-12	2016-05-12
2016-04-12	2016-05-12
2016-04-12	2016-05-12
2016-04-12	2016-05-12
2016-04-12	2016-05-12
2016-04-12	2016-05-12
2016-04-12	2016-05-12
2016-04-12	2016-05-12
2016-04-12	2016-05-12
2016-04-12	2016-05-12

STEP 4: ANALYZE

During the analyze phase, the goal is to derive insights from the cleaned and processed data to answer specific questions or discover patterns that can inform decision-making. The phase includes Exploratory Data Analysis (EDA), calculating summary statistics, engineering features, and preparing for visualization and reporting. It involves examining and summarizing key characteristics of the data, identifying patterns, relationships, and potential issues.

Exploratory Data Analysis: It is a crucial step in understanding the dataset by examining its underlying patterns, relationships and anomalies.

1. Identify Unique Users in Each table

This step involved cross-verifying user counts across multiple datasets to identify any discrepancies that could affect the accuracy of subsequent analyses. By ensuring consistency in user counts, we mitigated the risk of biased insights or incomplete consumer profiles. The same query is run on all tables and all the other tables have 33 users except sleepday_merged and weightloginfo_merged.

Purpose: Verify unique users across datasets.

Queries Executed: Counted distinct user IDs in each dataset.

Impact: Prevents incorrect analysis results due to missing or extra users.

Usefulness: Ensures data integrity and consistency across all datasets.

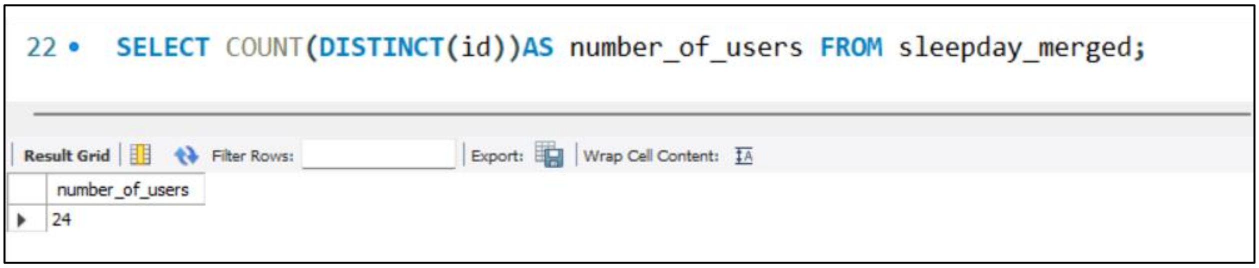
Check unique numbers of users in dailyactivity_merged

```sql
-- Exploring Data to check unique numbers of users in each table
SELECT COUNT(DISTINCT(id))AS number_of_users FROM dailyactivity_merged;
```

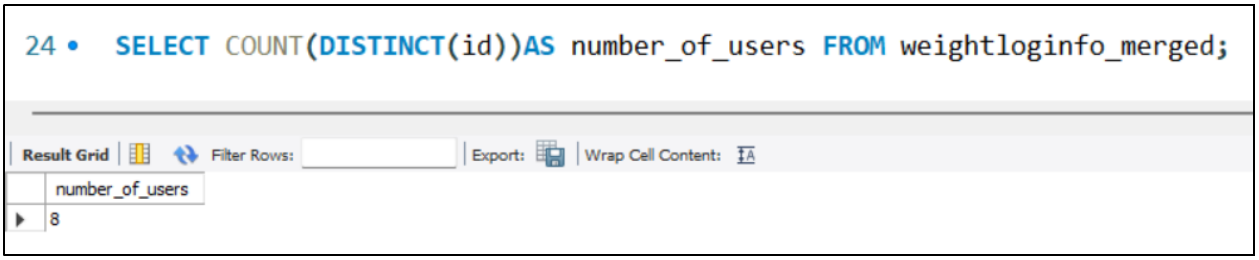
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

number_of_users
33

Check unique numbers of users in sleepday_merged



Check unique numbers of users in weightloginfo_merged



2. Identifying Common Users Between Tables

This step ensures that analyses involving user data from multiple sources are valid, enhancing the reliability of insights drawn. All the tables have 33 users in common except sleepday_merged_new that has 24 and weightloginfo_merged_new that has 8 users in common. All the tables are compared with dailyactivity_merged_new

Purpose: To find common data points between tables, which helps in understanding the overlap and integration of datasets. It focuses on verifying common users between tables before dropping any redundant tables.

Queries Executed: used a WITH clause to join distinct user IDs from both tables, flagged common users, and counted them.

Impact: Helps in understanding user overlap, ensuring data integration is accurate and comprehensive.

Usefulness: Enables accurate user-centric analysis by ensuring data from different tables can be combined effectively.

Comparing dailyactivity_merged_new and dailycalories_merged_new

```sql
WITH base AS(
SELECT *,
CASE WHEN id1=id2 THEN "Common Users"
ELSE "Non Common Users"
END AS flag FROM
( SELECT DISTINCT (id) as id1 from dailyactivity_merged_new ) as a
LEFT JOIN (SELECT DISTINCT (id) as id2 from sleepday_merged_new )as b
ON a.id1= b.id2)
Select flag, count(*)as number_of_users
from base
GROUP BY flag
ORDER BY flag
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

flag	number_of_users
Common Users	24
Non Common Users	9

Comparing dailyactivity_merged and sleepday_merged

```sql
WITH base AS(
SELECT *,
CASE WHEN id1=id2 THEN "Common Users"
ELSE "Non Common Users"
END AS flag FROM
( SELECT DISTINCT (id) as id1 from dailyactivity_merged_new ) as a
LEFT JOIN (SELECT DISTINCT (id) as id2 from sleepday_merged_new )as b
ON a.id1= b.id2)
Select flag, count(*)as number_of_users
from base
GROUP BY flag
ORDER BY flag
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

flag	number_of_users
Common Users	24
Non Common Users	9

Comparing dailyactivity_merged_new and weightloginfo_merged_new

```sql
WITH base AS(
SELECT *,
CASE WHEN id1=id2 THEN "Common Users"
ELSE "Non Common Users"
END AS flag FROM
( SELECT DISTINCT (id) as id1 from dailyactivity_merged_new ) as a
LEFT JOIN (SELECT DISTINCT (id) as id2 from weightloginfo_merged_new )as b
ON a.id1= b.id2)
Select flag, count(*)as number_of_users
from base
GROUP BY flag
ORDER BY flag
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

flag	number_of_users
Common Users	8
Non Common Users	25

3.**Comparing tables and dropping the tables that has redundant data**

This step ensures that we streamline our dataset by removing unnecessary duplication, thereby optimizing storage and enhancing the clarity of subsequent analyses. All the tables have duplicate data with dailyactivity_merged_new except hourlyintensities_merged_new, sleepday_merged_new and weightloginfo_merged_new.

Purpose: To identify and eliminate tables containing redundant data to ensure a streamlined, efficient dataset for analysis.

Queries Executed: Used CASE statement to compare datasets and DROP TABLE to remove redundant tables.
Impact: Optimizes storage, improves query performance, and enhances data clarity.
Usefulness: Facilitates efficient analysis and ensures accurate, reliable insights.

Comparing dailyactivity_merged_new and dailycalories_merged_new

```sql
-- Steps to Verify and Drop the tables consisting redundant data
SELECT
    da.Id,
    da.ActivityDate,
    da.Calories AS ActivityCalories,
    dc.Calories AS DailyCalories,
    CASE
        WHEN da.Calories = dc.Calories THEN 'Match'
        ELSE 'Mismatch'
    END AS MatchStatus
FROM
    dailyactivity_merged_new da
LEFT JOIN
    dailycalories_merged_new dc ON da.Id = dc.Id AND da.ActivityDate = dc.ActivityDay_new
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	ActivityDate	ActivityCalories	DailyCalories	MatchStatus
1503960366	2016-04-12	1985	1985	Match
1503960366	2016-04-13	1797	1797	Match
1503960366	2016-04-14	1776	1776	Match
1503960366	2016-04-15	1745	1745	Match
1503960366	2016-04-16	1863	1863	Match
1503960366	2016-04-17	1728	1728	Match
1503960366	2016-04-18	1921	1921	Match
1503960366	2016-04-19	2035	2035	Match
1503960366	2016-04-20	1786	1786	Match
1503960366	2016-04-21	1775	1775	Match
1503960366	2016-04-22	1827	1827	Match

Comparing dailyactivity_merged_new and dailycalories_merged_new

```sql
SELECT
    da.Id,
    da.ActivityDate,
    da.TotalSteps,
    ds.StepTotal,
    CASE
        WHEN da.TotalSteps = ds.StepTotal THEN 'Match'
        ELSE 'Mismatch'
    END AS StepMatchStatus
FROM
    dailyactivity_merged_new da
LEFT JOIN
    dailysteps_merged_new ds ON da.Id = ds.Id AND da.ActivityDate = ds.ActivityDay_new;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	ActivityDate	TotalSteps	StepTotal	StepMatchStatus
1503960366	2016-04-12	13162	13162	Match
1503960366	2016-04-13	10735	10735	Match
1503960366	2016-04-14	10460	10460	Match
1503960366	2016-04-15	9762	9762	Match
1503960366	2016-04-16	12669	12669	Match
1503960366	2016-04-17	9705	9705	Match
1503960366	2016-04-18	13019	13019	Match
1503960366	2016-04-19	15506	15506	Match
1503960366	2016-04-20	10544	10544	Match
1503960366	2016-04-21	9819	9819	Match
1503960366	2016-04-22	12764	12764	Match
1503960366	2016-04-23	14371	14371	Match

Dropping all the tables that has redundant data

```sql
drop table dailycalories_merged;
drop table dailycalories_merged_new;
drop table dailyintensities_merged;
drop table dailyintensities_merged_new;
drop table dailysteps_merged;
drop table dailysteps_merged_new;
drop table hourlycalories_merged;
drop table hourlycalories_merged_new;
drop table hourlysteps_merged;
drop table hourlysteps_merged_new;
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
38	14:38:18	drop table dailycalories_merged	0 row(s) affected	0.047 sec
39	14:38:19	drop table dailycalories_merged_new	0 row(s) affected	0.000 sec
40	14:38:19	drop table dailyintensities_merged	0 row(s) affected	0.015 sec
41	14:38:19	drop table dailyintensities_merged_new	0 row(s) affected	0.000 sec
42	14:38:19	drop table dailysteps_merged	0 row(s) affected	0.016 sec
43	14:38:19	drop table dailysteps_merged_new	0 row(s) affected	0.000 sec
44	14:38:19	drop table hourlycalories_merged	0 row(s) affected	0.016 sec

4. Creating Master Table

Consolidating cleaned datasets into a master table streamlines data accessibility and enhances analytical efficiency. By integrating diverse data sources into a cohesive dataset, we facilitated seamless data exploration and in-depth analyses, empowering stakeholders with actionable insights for strategic business decisions.

Purpose: Integrate cleaned datasets into a comprehensive master table.

Queries Executed: Joined datasets on user ID and date field.

Actions Taken: Created a unified dataset ready for analysis.

Impact: Provides a centralized dataset for analysis.

Usefulness: Simplifies data access and manipulation for effective analysis.

Create master _table

```sql
-- Create Master table
CREATE TABLE master_table AS
SELECT
    da.id,da.ActivityDate,da.TotalSteps,da.TrackerDistance,da.LoggedActivitiesDistance,da.VeryActiveDistance,da.ModeratelyActiveDistance,da.LightActiveDistance,
    da.SedantaryActiveDistance,da.VeryActiveMinutes,da.FairlyActiveMinutes,da.LightlyActiveMinutes,da.SedantaryMinutes,da.Calories,
    sd.TotalMinutesAsleep,sd.TotalTimeInBed,
    wl.WeightPounds,wl.Fat,wl.BMI
FROM
    dailyactivity_merged_new da
LEFT JOIN
    sleepday_merged_new sd ON da.id = sd.id AND da.ActivityDate= sd.SleepDayNew
LEFT JOIN
    weightloginfo_merged_new wl ON da.id = wl.id AND da.ActivityDate=wl.WeightDate_new
    group by da.id,da.ActivityDate
    order by da.id,da.ActivityDate;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ggedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance	LightActiveDistance	SedantaryActiveDistance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesA

5. Imputing Null Values in Master Table

Imputing null values involved identifying and filling missing entries in the master table to prevent skewed results. This step was crucial for maintaining the robustness of the dataset, ensuring all analyses are based on comprehensive and reliable data.

Purpose: To ensure dataset completeness and maintain the integrity of the analysis by addressing any missing data points.

Queries Executed: Used COALESCE function

Actions Taken: Used appropriate imputation technique and filling null values with the zero

Impact: Reduces potential biases and inaccuracies in the analysis by ensuring all records are complete and usable.

Usefulness: Enhances the reliability of analytical outcomes, leading to more robust insights and informed decision-making.

```sql
-- Imputing null values in master table
UPDATE master_table
SET TotalSteps = COALESCE(TotalSteps, 0);
UPDATE master_table
SET TrackerDistance = COALESCE(TrackerDistance, 0);
UPDATE master_table
SET LoggedActivitiesDistance = COALESCE(LoggedActivitiesDistance, 0);
UPDATE master_table
SET VeryActiveDistance = COALESCE(VeryActiveDistance, 0);
UPDATE master_table
SET ModeratelyActiveDistance = COALESCE(ModeratelyActiveDistance, 0);
UPDATE master_table
SET LightActiveDistance = COALESCE(LightActiveDistance, 0);
UPDATE master_table
SET SedantaryActiveDistance = COALESCE(SedantaryActiveDistance, 0);
UPDATE master_table
SET VeryActiveMinutes = COALESCE(VeryActiveMinutes, 0);
UPDATE master_table
SET FairlyActiveMinutes = COALESCE(FairlyActiveMinutes, 0);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

edantaryActiveDistance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesAsleep	TotalTimeInBed	WeightPounds	BMI	Totalintensity	DayOfWeek	User_leveL
	25	13	328	728	1985	327	346	0	0	429	Tuesday	Advanced
	21	19	217	776	1797	384	407	0	0	318	Wednesday	Advanced
	30	11	181	1218	1776	0	0	0	0	293	Thursday	Advanced
	29	34	209	726	1745	412	442	0	0	364	Friday	Intermediate
	36	10	221	773	1863	340	367	0	0	349	Saturday	Advanced
	38	20	164	539	1728	700	712	0	0	318	Sunday	Intermediate
	42	16	233	1149	1921	0	0	0	0	391	Monday	Advanced
	50	31	264	775	2035	304	320	0	0	476	Tuesday	Advanced
	28	12	205	818	1786	360	377	0	0	313	Wednesday	Advanced

Summary Statistics: It provides a concise overview of the dataset's key metrics, such as mean, median, minimum, maximum, and standard deviation.

1.Calculating Average steps per day: This analysis computes the mean number of steps taken by each user daily, providing insights into overall activity levels.

```sql
-- calculate average steps per user
ALTER TABLE master_table
ADD AverageSteps FLOAT;

UPDATE master_table
SET AverageSteps = (
    SELECT AVG(TotalSteps)
    FROM dailyactivity_merged_new AS d
    WHERE d.ID = master_table.id
    GROUP BY master_table.id
);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

edantaryActiveDistance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesAsleep	TotalTimeInBed	WeightPounds	BMI	Totalintensity	DayOfWeek	AverageSteps
	44	8	203	574	1740	594	611	0	0	351	Sunday	12116.7
	46	11	206	835	1819	338	342	0	0	366	Monday	12116.7
	46	31	214	746	1859	383	403	0	0	414	Tuesday	12116.7
	36	23	251	669	1783	285	306	0	0	393	Wednesday	12116.7
	0	0	0	1440	0	0	0	0	0	429	Thursday	12116.7
	0	0	146	1294	1432	0	0	0	0	146	Tuesday	5743.9
	0	0	148	1292	1411	0	0	0	0	148	Wednesday	5743.9
	0	0	236	1204	1572	0	0	0	0	236	Thursday	5743.9
	0	0	96	1344	1344	0	0	0	0	96	Friday	5743.9
	0	0	176	1264	1463	0	0	0	0	176	Saturday	5743.9
	15	22	127	1276	1554	0	0	0	0	216	Sunday	5743.9

2.Calculating Average Sleep hours: Identifying the Average sleep hours recorded by each user daily.

```sql
select *,AVG(TotalMinutesAsleep) FROM master_table
group by id,ActivityDate;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ctiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesAsleep	TotalTimeInBed	Totalintensity	DayOfWeek	AverageSteps	User_leveL	AVG(TotalMinutesAsleep)
	13	328	728	1985	327	346	429	Tuesday	12116.7	Advanced	327.0000
	19	217	776	1797	384	407	318	Wednesday	12116.7	Advanced	384.0000
	11	181	1218	1776	0	0	293	Thursday	12116.7	Advanced	0.0000

3.Calculating Max steps per day: Identifying the maximum steps recorded by each user daily reveals peak activity levels and engagement with fitness goals.

```sql
select *,MAX(TotalSteps) FROM master_table
group by id,ActivityDate;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesAsleep	TotalTimeInBed	Totalintensity	DayOfWeek	AverageSteps	User_leveL	max(TotalSteps)
25	13	328	728	1985	327	346	429	Tuesday	12116.7	Advanced	13162
21	19	217	776	1797	384	407	318	Wednesday	12116.7	Advanced	10735
30	11	181	1218	1776	0	0	293	Thursday	12116.7	Advanced	10460
29	34	209	726	1745	412	442	364	Friday	12116.7	Advanced	9762
36	10	221	773	1863	340	367	349	Saturday	12116.7	Advanced	12669

4.Calculating Max sleep hour: This analysis captures the highest number of sleep hours logged by users in a day, highlighting optimal rest periods.

```sql
select *,MAX(TotalMinutesAsleep) FROM master_table
group by id,ActivityDate;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ctiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesAsleep	TotalTimeInBed	Totalintensity	DayOfWeek	AverageSteps	User_leveL	MAX(TotalMinutesAsleep)
	13	328	728	1985	327	346	429	Tuesday	12116.7	Advanced	327
	19	217	776	1797	384	407	318	Wednesday	12116.7	Advanced	384
	11	181	1218	1776	0	0	293	Thursday	12116.7	Advanced	0
	34	209	726	1745	412	442	364	Friday	12116.7	Advanced	412
	10	221	773	1863	340	367	349	Saturday	12116.7	Advanced	340

Feature Engineering: Feature engineering is crucial for enhancing the dataset's analytical capabilities by creating meaningful variables that can improve model performance.

1. **Adding day of week:** By categorizing activity data by the day of the week, we enhance analysis granularity and identify weekly trends in user behavior.

```sql
-- Add column for day of week in master_table
ALTER table master_table
ADD COLUMN DayOfWeek text;
UPDATE master_table
SET DayOfWeek= CASE
WHEN weekday(ActivityDate)='0' THEN 'Monday'
WHEN weekday(ActivityDate)='1' THEN 'Tuesday'
WHEN weekday(ActivityDate)='2' THEN 'Wednesday'
WHEN weekday(ActivityDate)='3' THEN 'Thursday'
WHEN weekday(ActivityDate)='4' THEN 'Friday'
WHEN weekday(ActivityDate)='5' THEN 'Saturday'
WHEN weekday(ActivityDate)='6' THEN 'Sunday'
END
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nce	SedantaryActiveDistance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesAsleep	TotalTimeInBed	WeightPounds	Fat	BMI	Totalintensity	DayOfWeek
	0	25	13	328	728	1985	327	346	NULL	NULL	NULL	429	Tuesday
	0	21	19	217	776	1797	384	407	NULL	NULL	NULL	318	Wednesday
	0	30	11	181	1218	1776	NULL	NULL	NULL	NULL	NULL	293	Thursday
	0	29	34	209	726	1745	412	442	NULL	NULL	NULL	364	Friday

2. **User Categorization on Activity Level**: Classifying users based on average steps.

```sql
-- Add column for categorizing users on basis of level of activity
ALTER table master_table
ADD COLUMN User_leveL text;

UPDATE master_table
SET User_level=case
WHEN averagesteps < 1000 THEN 'Inactive'
WHEN averagesteps < 5000 THEN 'Beginner'
WHEN averagesteps BETWEEN 5000 AND 9999 THEN 'Intermediate'
Else 'Advanced'
END ;
SELECT * from master_table;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Distance	VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesAsleep	TotalTimeInBed	WeightPounds	BMI	Totalintensity	DayOfWeek	AverageSteps	User_leveL
	25	13	328	728	1985	327	346	0	0	429	Tuesday	12116.7	Advanced
	21	19	217	776	1797	384	407	0	0	318	Wednesday	12116.7	Advanced
	30	11	181	1218	1776	0	0	0	0	293	Thursday	12116.7	Advanced
	29	34	209	726	1745	412	442	0	0	364	Friday	12116.7	Advanced

3. **User Categorization on sleep Duration**: Classifying users based on average sleeping hours.

```sql
ALTER TABLE master_table
ADD COLUMN SleeperCategory VARCHAR(20);

UPDATE master_table
SET SleeperCategory = CASE
    WHEN TotalMinutesAsleep >= 540 THEN 'Prolonged Sleeper'
    WHEN TotalMinutesAsleep >= 420 THEN 'Optimal Sleeper'
    ELSE 'Minimal Sleeper'
END;
```

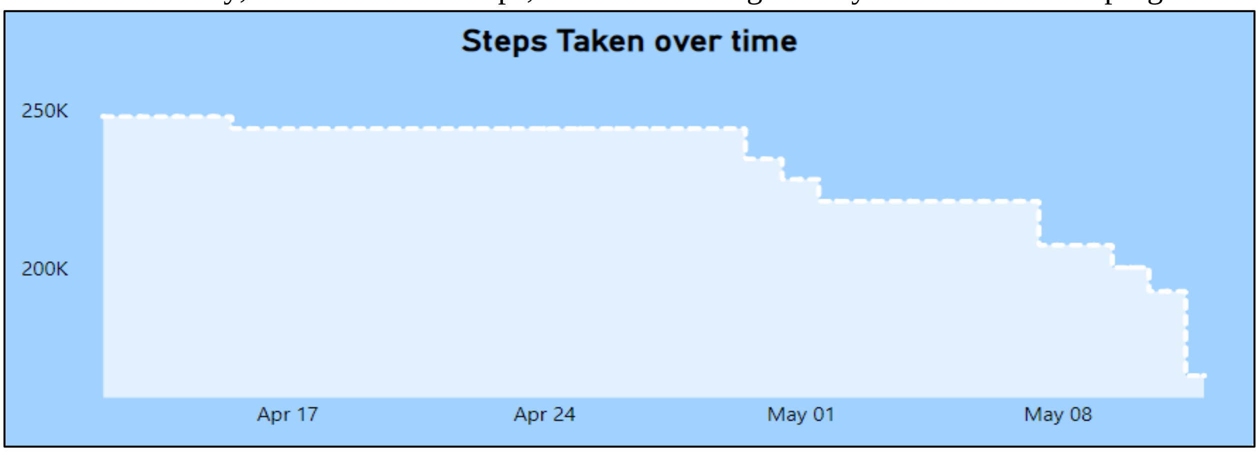
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

VeryActiveMinutes	FairlyActiveMinutes	LightlyActiveMinutes	SedantaryMinutes	Calories	TotalMinutesAsleep	TotalTimeInBed	Totalintensity	DayOfWeek	AverageSteps	User_leveL	SleeperCategory
25	13	328	728	1985	327	346	429	Tuesday	12116.7	Advanced	Minimal Sleeper
21	19	217	776	1797	384	407	318	Wednesday	12116.7	Advanced	Minimal Sleeper
30	11	181	1218	1776	0	0	293	Thursday	12116.7	Advanced	Minimal Sleeper
29	34	209	726	1745	412	442	364	Friday	12116.7	Advanced	Minimal Sleeper
36	10	221	773	1863	340	367	349	Saturday	12116.7	Advanced	Minimal Sleeper
38	20	164	539	1728	700	712	318	Sunday	12116.7	Advanced	Prolonged Sleeper
42	16	233	1149	1921	0	0	391	Monday	12116.7	Advanced	Minimal Sleeper

OBSERVATIONS AND VISUALIZATION:

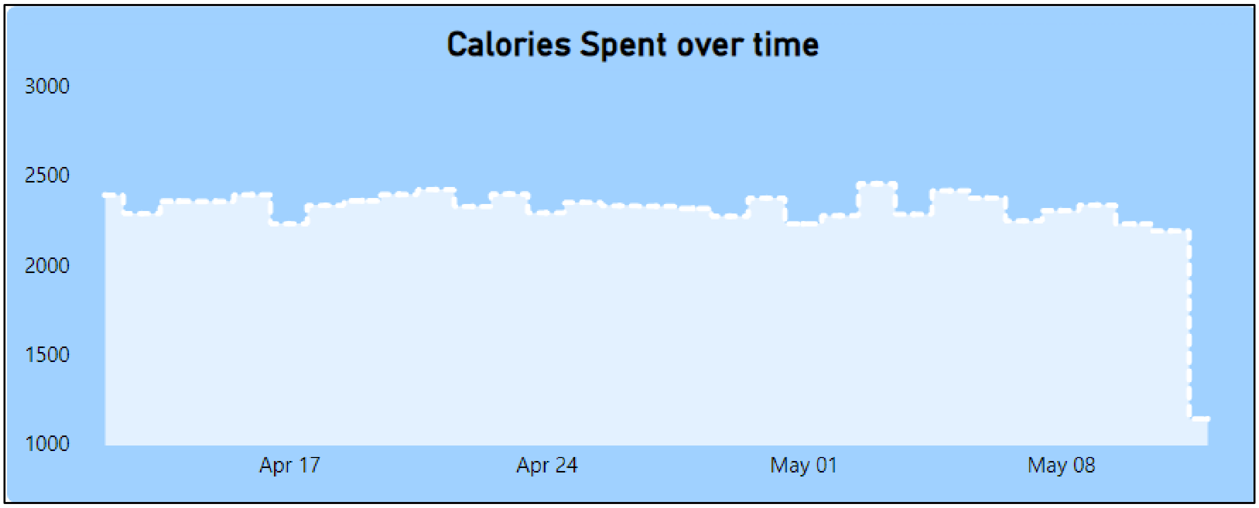
Steps taken over time:

As illustrated in the graph, the average number of steps taken by users shows a declining trend over time. Initially, users took more steps, but this number gradually decreased as time progressed.



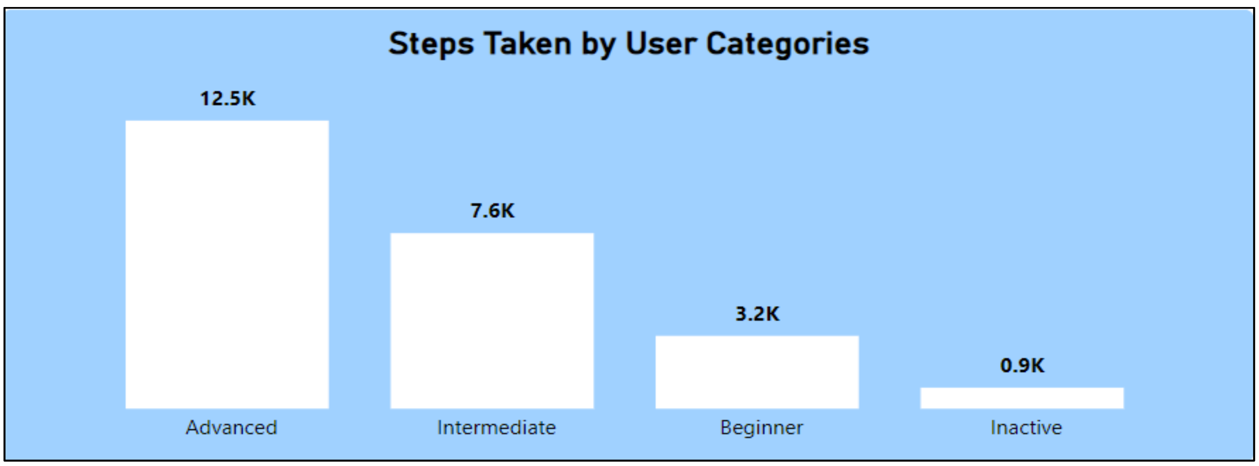
Calories Burnt over time:

The graph illustrating calories burnt reveals a relatively static pattern. There are fluctuations, with some days showing higher calories burnt and other days showing lower values, but no significant overall trend is observed.



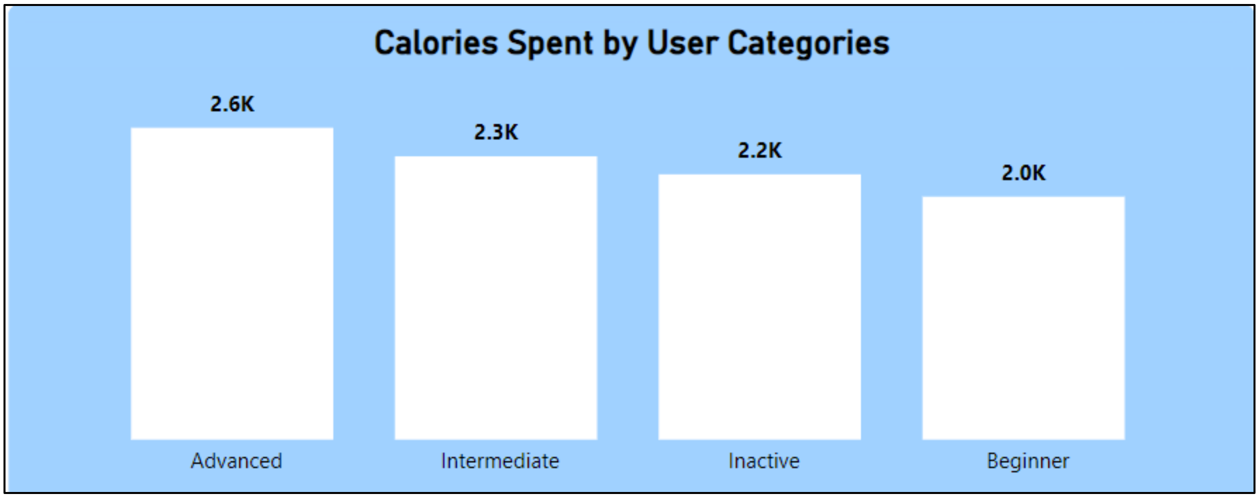
Steps Taken by Various User Groups

Advanced users average 12.5 steps per day, while Intermediate users take an average of 7.6 steps. Beginners average 3.2 steps, and Inactive users average just 0.9 steps. This trend highlights a clear correlation between user activity level and the number of steps taken: as user engagement decreases, so does the average step count. This suggests that higher engagement is associated with significantly more physical activity.



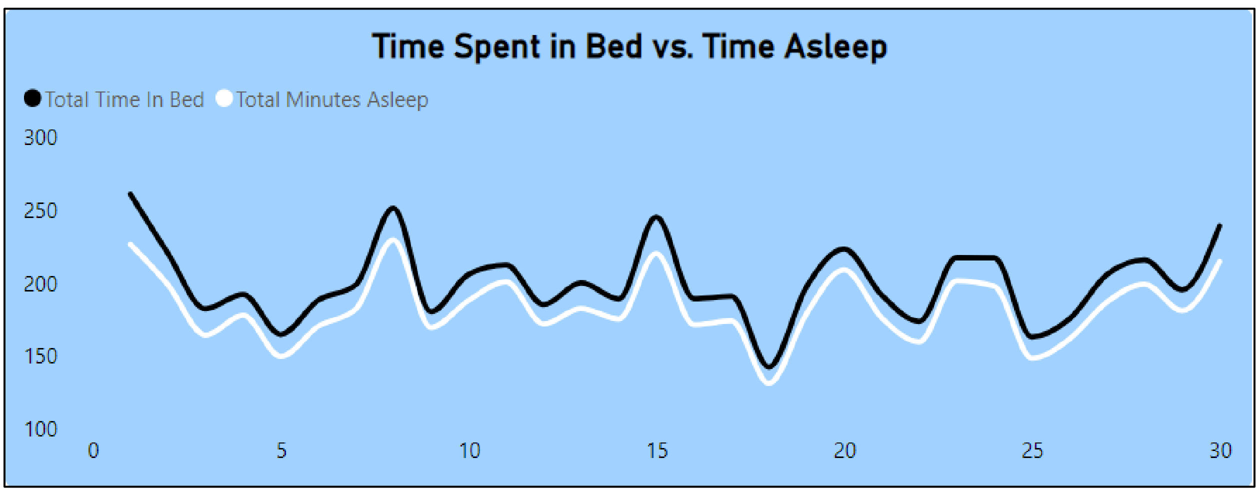
Calories Burnt by Various User Groups

Advanced users burn an average of 2.6 calories per day, whereas intermediate users burn 2.3 calories. Beginners average 2.0 calories, and inactive users average 2.2 calories. While there is some variation in calorie expenditure among different user categories, advanced users generally burn slightly more calories than the others. The differences across groups are relatively minor, suggesting that although higher engagement is associated with a slight increase in calories burnt, this effect is less pronounced compared to the variation seen in step counts.



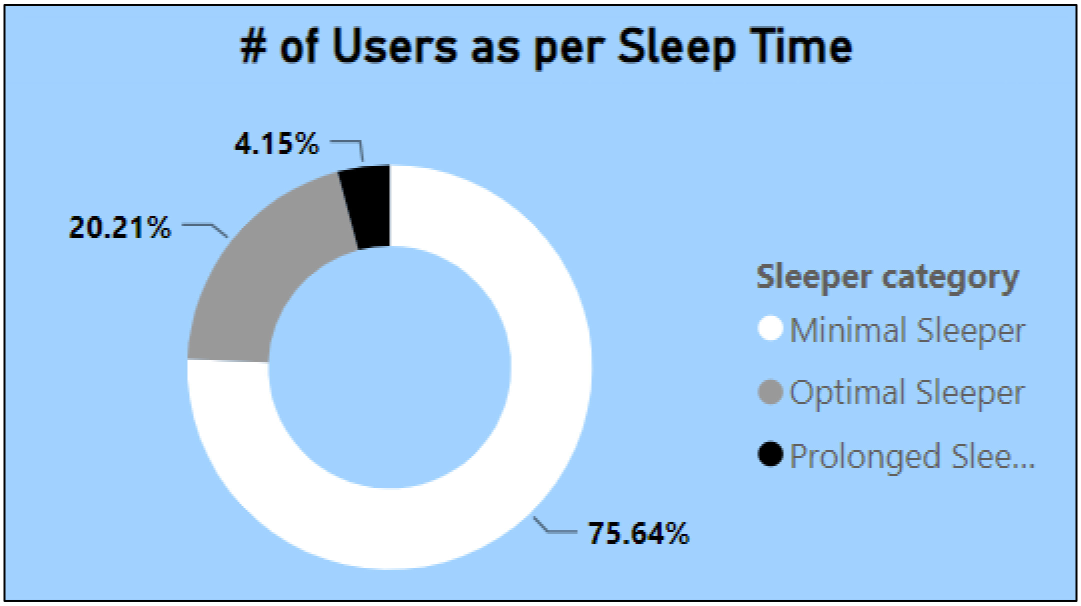
Time Spent in Bed vs Time Asleep

The analysis of time spent in bed versus time spent sleeping reveals a generally strong correlation, with individuals who spend more time in bed typically sleeping longer. However, discrepancies between these two metrics suggest potential issues like difficulty falling asleep or interrupted sleep. Understanding this relationship helps identify opportunities to improve sleep quality and efficiency.



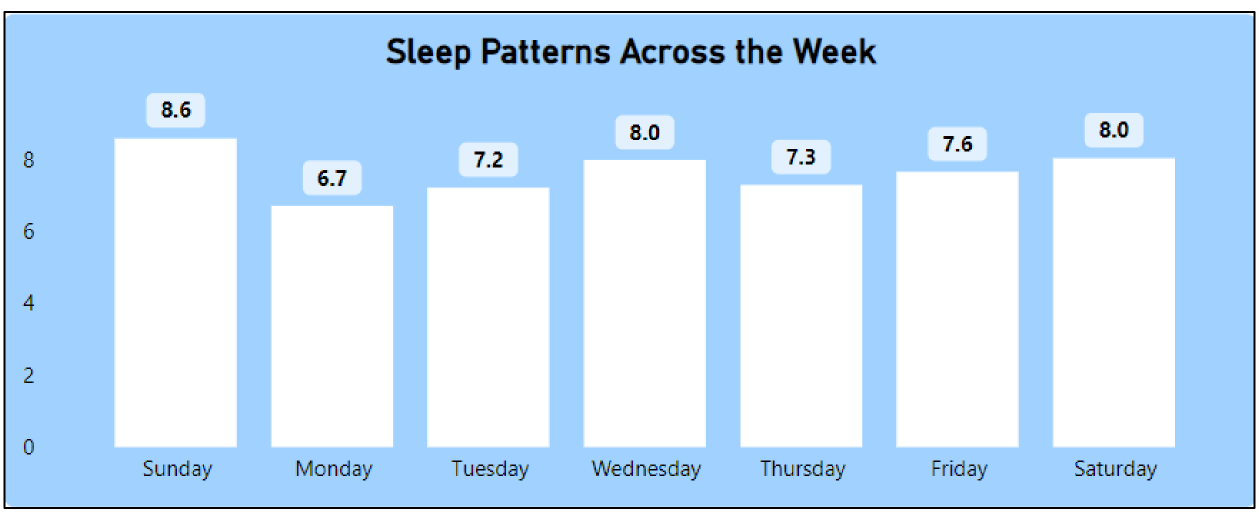
Time Spent in Bed vs Time Asleep

Users were categorized into three sleep duration groups: Minimal, Optimal, and Prolonged. The distribution is as follows: 75.64% are Minimal sleepers, 20.21% are Optimal sleepers, and 4.15% are Prolonged sleepers. The data indicates a predominant trend towards Minimal sleep, with a smaller proportion achieving Optimal sleep, and only a few experiencing Prolonged sleep. This suggests that most users may be getting insufficient sleep, while only a minority reach or exceed recommended sleep durations.



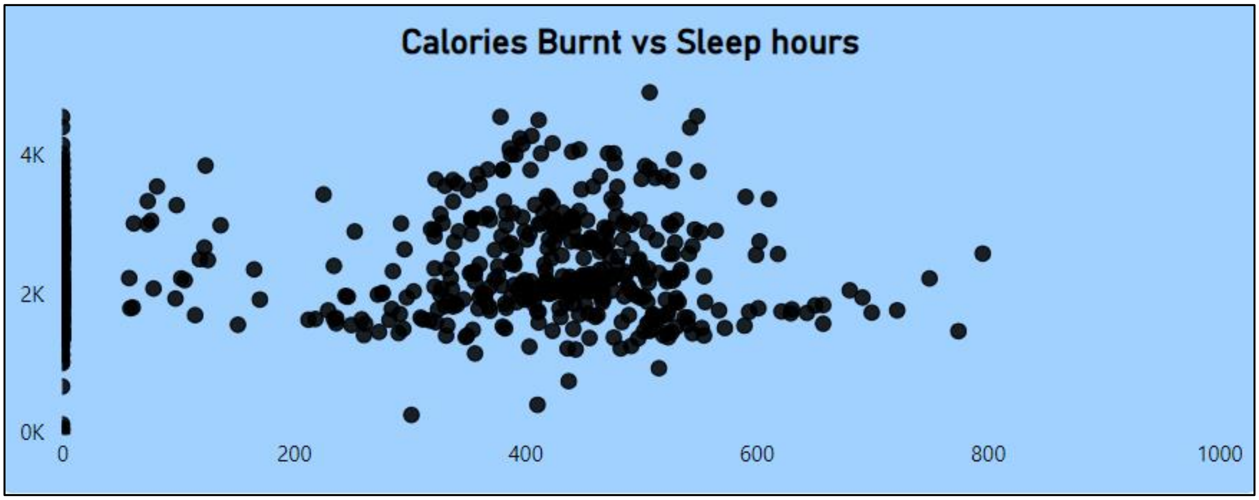
Sleep pattern across week

The average sleep duration varies significantly throughout the week. Users get the most sleep on Sundays, averaging 8.6 hours, the highest of the week. This amount decreases during the workweek, with Monday showing the least at 6.7 hours. This reduction may be attributed to the shift from weekend relaxation to weekday obligations. From Tuesday to Thursday, sleep duration stabilizes between 7.2 and 7.6 hours. Mid-week, there is a slight increase to 8.0 hours on Wednesday and again on Saturday. This pattern suggests that users generally achieve more rest on weekends than on weekdays, indicating that weekday schedules might influence overall sleep duration.



Calories burnt vs Total Minutes Sleep

An R squared value of 0.01 between total minutes asleep and calories burnt shows a very weak correlation, meaning only 1% of the variation in calories burnt can be explained by sleep duration. This indicates that sleep has minimal predictive power for calories burnt, suggesting that other factors like physical activity or metabolism play a more significant role in determining calorie expenditure.

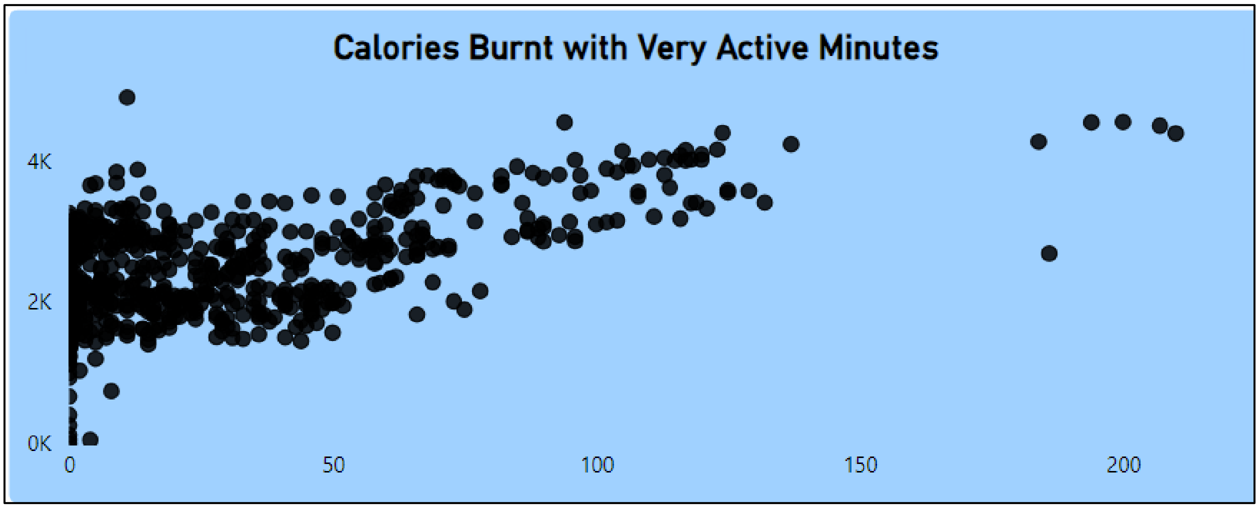


Calories burnt vs Total Steps

An R Squared value of 0.35 for the relationship between calories burnt and total steps indicates a moderate correlation. This means that approximately 35% of the variability in calories burnt can be explained by the number of steps taken. While not a perfect predictor, this moderate correlation suggests that as the number of steps increases, there is a noticeable increase in calories burnt. However, other factors also contribute to calorie expenditure, so the relationship is not entirely dependent on the number of steps alone.

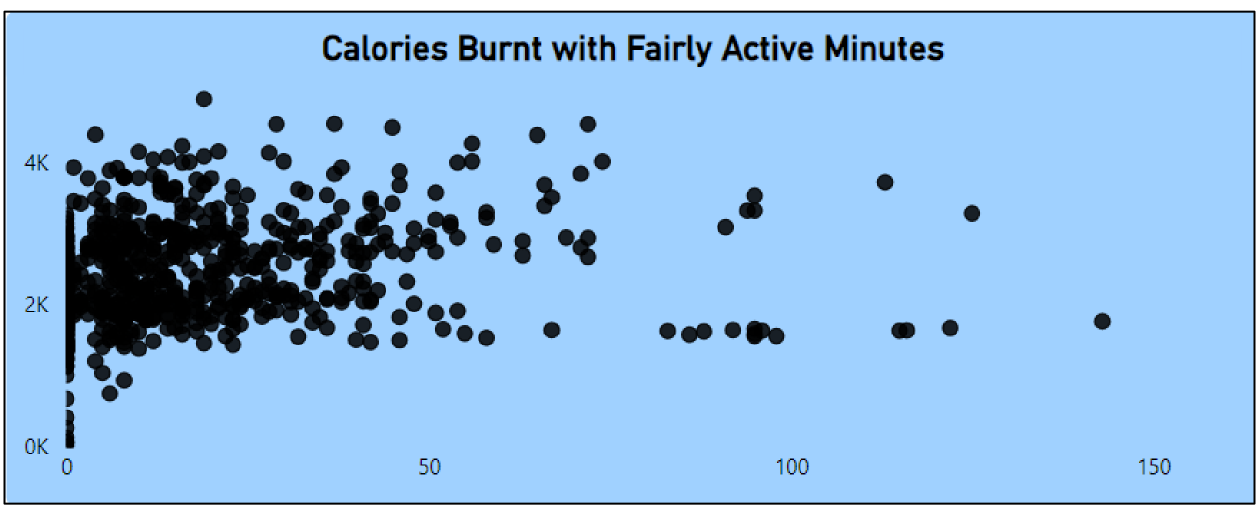
Calories burnt vs Very Active Minutes

An R Squared value of 0.38 for the relationship between calories burnt and very active minutes indicates a moderate correlation. This means that approximately 38% of the variation in calories burnt can be explained by the number of very active minutes. The moderate correlation suggests that while very active minutes do have a noticeable impact on calorie expenditure, other factors also contribute significantly to the overall calories burnt.



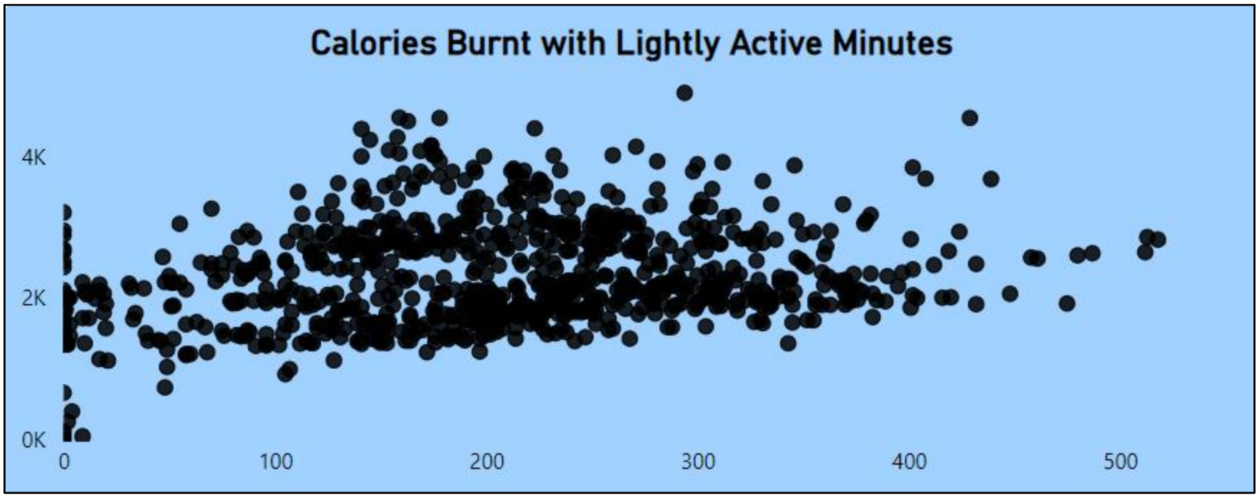
Calories burnt vs Fairly Active Minutes

An R Squared value of 0.09 for the relationship between calories burnt and fairly active minutes indicates a weak correlation. This suggests that only 9% of the variability in calories burnt can be explained by the time spent in fairly active minutes. The remaining 91% of the variability is likely influenced by other factors not captured in this analysis. Therefore, while there is some connection between fairly active minutes and calories burnt, it is relatively minor, and additional variables may be more significant in determining calorie expenditure.



Calories burnt vs Lightly Active Minutes

An R Square value of 0.08 for the relationship between calories burnt and lightly active minutes indicates a weak correlation. This means that only 8% of the variation in calories burnt can be explained by the amount of lightly active minutes. Therefore, while there is some association between lightly active minutes and calories burnt, other factors likely have a more significant impact on calorie expenditure.



Calories burnt vs Sedentary Minutes

An R Square value of 0.01 for the relationship between calories burnt and sedentary minutes indicates a very weak correlation. This means that only 1% of the variability in calories burnt can be explained by the amount of sedentary time. Essentially, sedentary minutes have minimal influence on calories burnt, suggesting that other factors likely play a more significant role in determining calorie expenditure.

Insights and Findings

User Activity Levels: Users, on average, walk 7,638 steps daily, covering 5.49 km. However, this is below the CDC's recommended 10,000 steps per day.

The average time spent in highly active minutes is 21.16 minutes per day, which is less than the recommended 30 minutes of exercise.

Sedentary Behavior: Users spend an average of 991.2 minutes (16.52 hours) sedentary within a 24-hour period. This extended sedentary time poses health risks and is above the advised limit of 8 hours or less.

Calories Burned: Users burn an average of 2,304 kCal daily. While this varies based on individual factors, it provides a baseline for understanding user activity levels.

Sleep Patterns: Users sleep an average of 419 minutes (around 7 hours) per day and spend 458 minutes (7 hours 30 minutes) in bed, indicating an average of 30 minutes awake in bed. This suggests users are generally getting sufficient sleep.

Weight and BMI: The average BMI of users is 25.19, categorizing them as slightly overweight. The average weight is 72 kg (158.8 pounds). However, data on weight is limited, with only 8 respondents sharing their weight information.

Analysis of Correlations

Steps and Calories:

A moderate correlation ($R^2 = 0.35$) exists between total steps taken and calories burned, indicating that increased steps generally lead to higher calorie expenditure.

Active Minutes and Calories:

Very active minutes have a moderate correlation ($R^2 = 0.38$) with calories burned, while fairly active minutes ($R^2 = 0.09$) and lightly active minutes ($R^2 = 0.08$) show weaker correlations.

Sedentary minutes show a very weak correlation ($R^2 = 0.01$) with calories burned, highlighting that other factors influence calorie expenditure more significantly.

Sleep and Calories:

A very weak correlation ($R^2 = 0.01$) between total minutes asleep and calories burned suggests that sleep duration has minimal impact on calorie expenditure.

STEP 5: SHARE

The fifth phase involves sharing the findings from the data analysis. This is achieved by creating compelling visualizations, such as graphs and charts, which effectively communicate the results. By converting numerical data into visual formats, we simplify the understanding of patterns and insights for the audience. The report is shared on PowerBI Services.

STEP 6: ACT

The sixth phase is the Implement phase, where the insights gained from the analysis are transformed into actionable strategies. This phase is crucial for helping stakeholders make informed, data-driven decisions to tackle business challenges.

Strategic Marketing Initiatives:

Enhancing User Engagement: Promote increased physical activity through targeted campaigns that emphasize the health benefits of walking and regular exercise. Highlight the importance of reaching 10,000 steps and 30 minutes of active exercise daily. Develop initiatives to reduce sedentary behavior by encouraging short, frequent breaks for physical activity throughout the day.

Improving Sleep Quality: Address sleep quality issues by providing users with tips and resources to improve sleep hygiene. Highlight the importance of consistent sleep patterns and adequate rest.

Customized Marketing Campaigns: Utilize the data to create personalized marketing strategies that resonate with users based on their activity levels and sleep patterns. Emphasize the benefits of Bellabeat products in helping users achieve their health goals.

Building a Supportive Community: Foster a sense of community among users by creating online platforms and social media groups where users can share their progress, challenges, and successes. Encourage participation in group activities and challenges to boost motivation.

Continuous Data Monitoring and Feedback: Implement continuous data monitoring to track user progress and provide personalized feedback. Use this data to refine and improve Bellabeat's product offerings and marketing strategies.

Conclusion

The analysis of smart device data reveals significant insights into user behavior, highlighting areas for improvement in physical activity, sedentary behavior, and sleep patterns. By leveraging these insights, Bellabeat can enhance its marketing strategies, foster user engagement, and empower women to take control of their health. The strategic recommendations aim to address the identified challenges and promote a healthier lifestyle among Bellabeat users, ultimately contributing to the company's mission of empowering women with tools and knowledge for better health.