Dataset: gym_membership.csv
Context: Health & Fitness Industry

This project focuses on **exploratory data analysis (EDA)** of a gym membership dataset to understand customer behavior and engagement trends.
I used **Pandas** to clean and structure the data, identify missing or inconsistent values, and perform statistical and behavioral analyses.
The notebook highlights how data-driven insights can support decisions related to customer retention, membership plans, and operational improvements.

Skills demonstrated: Data cleaning, exploratory analysis, descriptive statistics, behavioral analysis, Pandas

Part 1 - Opening and preparation of the analysis

In this first part, we will import the Pandas library from Python, read the CSV file, and store it in a dataframe. At the end of this first part, we will check whether the dataframe has been loaded correctly.

```python
# In the line below, the pandas library is being imported.
import pandas as pd

# The CSV file is read using the 'read_csv' function and stored in a dataframe called 'df_membros'.
# Correcting the file path to match the available file in the system.
df_membros= pd.read_csv("./gym_membership.csv")

# The first five lines are being displayed using the ".head()" function.
print(df_membros.head())
```

```
   id  gender    birthday  Age abonoment_type  visit_per_week  \
0   1  Female  1997-04-18   27        Premium               4
1   2  Female  1977-09-18   47       Standard               3
2   3    Male  1983-03-30   41        Premium               1
3   4    Male  1980-04-12   44        Premium               3
4   5    Male  1980-09-10   44       Standard               2

       days_per_week  attend_group_lesson            fav_group_lesson  \
0  Mon, Sat, Tue, Wed                 True  Kickboxen, BodyPump, Zumba
1       Mon, Sat, Wed                False                         NaN
2                 Sat                 True                       XCore
3       Sat, Tue, Wed                False                         NaN
4            Thu, Wed                 True        Running, Yoga, Zumba

  avg_time_check_in avg_time_check_out  avg_time_in_gym  drink_abo  \
0          19:31:00           21:27:00              116      False
1          19:31:00           20:19:00               48      False
2          08:29:00           10:32:00              123       True
3          09:54:00           11:33:00               99       True
4          08:29:00           09:19:00               50      False

           fav_drink  personal_training name_personal_trainer  uses_sauna
0                NaN              False                   NaN        True
1                NaN               True               Chantal       False
2  berry_boost, lemon              True                  Mike       False
3      passion_fruit              True                  Mike        True
4                NaN               True                  Mike       False
```

Part 2 - Data structures and types

Following analysis, it was found that the following columns had been incorrectly categorised:

- “birthday” was mistakenly classified as an object instead of a datetime.
- "avg_time_check_in" and "avg_time_check_out" were mistakenly classified as objects instead of datetime.time.

```python
# The '.shape' function was used in the code below to indicate the number of rows and columns in the dataframe.
print("\nDataFrame dimension (rows, columns):")
print(df_membros.shape)

# The ".info" function returned the dataframe class, the number of rows, the number of columns, the column names,
# whether there are null fields or not, the types of each column, and the RAM used for storage.
print("\nGeneral information about the DataFrame:")
print(df_membros.info())
```

```
DataFrame dimension (rows, columns):
(1000, 17)

General information about the DataFrame:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    1000 non-null   int64
 1   gender                1000 non-null   object
 2   birthday              1000 non-null   object
 3   Age                   1000 non-null   int64
 4   abonoment_type        1000 non-null   object
 5   visit_per_week        1000 non-null   int64
 6   days_per_week         1000 non-null   object
 7   attend_group_lesson   1000 non-null   bool
 8   fav_group_lesson      503 non-null    object
 9   avg_time_check_in     1000 non-null   object
 10  avg_time_check_out    1000 non-null   object
 11  avg_time_in_gym       1000 non-null   int64
 12  drink_abo             1000 non-null   bool
 13  fav_drink             496 non-null    object
 14  personal_training     1000 non-null   bool
 15  name_personal_trainer  518 non-null    object
 16  uses_sauna            1000 non-null   bool
dtypes: bool(4), int64(4), object(9)
memory usage: 105.6+ KB
None
```

```python
# In the code below, the ".head" function was used to display the first five records of the dataframe.
print("First five entries:")
display(df_membros.head())

# Below, the ".tail()" function was used to show the last five records in the dataframe.
print("\nLast five entries:")
display(df_membros.tail())

# A função ".info()" foi utilizada para mostrar dados gerais do dataframe.
print("\nGeneral information about the dataframe:")
print(df_membros.info())
```

```
First five entries:
```

	id int64	**gender** object	**birthday** object	**Age** int64	**abonement_type** o.	**visit_per_week** in...	**days_per_week** o...	**a**
0	1	Female	1997-04-18	27	Premium	4	Mon, Sat, Tue, Wed	T
1	2	Female	1977-09-18	47	Standard	3	Mon, Sat, Wed	F
2	3	Male	1983-03-30	41	Premium	1	Sat	T
3	4	Male	1980-04-12	44	Premium	3	Sat, Tue, Wed	F
4	5	Male	1980-09-10	44	Standard	2	Thu, Wed	T

5 rows, 17 cols 10 / page Page 1 of 1

```
Last five entries:
```

	id int64	**gender** object	**birthday** object	**Age** int64	**abonement_type** o.	**visit_per_week** in...	**days_per_week** o...	**a**
995	996	Female	1984-09-22	40	Standard	3	Thu, Tue, Wed	F
996	997	Female	2008-11-19	15	Standard	3	Fri, Mon, Sun	T
997	998	Male	1984-10-05	40	Standard	2	Fri, Tue	F
998	999	Male	2001-02-22	23	Standard	4	Mon, Sun, Thu, Tue	T
999	1000	Female	2006-05-07	18	Premium	2	Thu, Tue	F

5 rows, 17 cols 10 / page Page 1 of 1

```
General information about the dataframe:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   id                    1000 non-null   int64 
 1   gender                1000 non-null   object
 2   birthday              1000 non-null   object
 3   Age                   1000 non-null   int64 
 4   abonement_type        1000 non-null   object
 5   visit_per_week        1000 non-null   int64 
 6   days_per_week         1000 non-null   object
 7   attend_group_lesson   1000 non-null   bool  
 8   fav_group_lesson      503 non-null    object
 9   avg_time_check_in     1000 non-null   object
 10  avg_time_check_out    1000 non-null   object
 11  avg_time_in_gym       1000 non-null   int64 
 12  drink_abo             1000 non-null   bool  
 13  fav_drink             496 non-null    object
 14  personal_training     1000 non-null   bool  
 15  name_personal_trainer  518 non-null    object
 16  uses_sauna            1000 non-null   bool  
dtypes: bool(4), int64(4), object(9)
memory usage: 105.6+ KB
None
```

```python
# In the code below, the ".describe()" function is being called to describe some information related to the numeric
# columns. ".round(2)" was added to show only 2 decimal places (instead of 6, as is the default).
print("Analytical view - numeric columns:")
display(df_membros.describe().round(2))

# In the code below, the ".describe()" function is being called to describe some information related to the
# categorical columns.
print("Analytical view - categorical columns:")
display(df_membros.describe(include='object'))
```

```
Analytical view - numeric columns:
```

	id float64	Age float64	visit_per_week fl...	avg_time_in_gym f.
cou...	1000	1000	1000	1000
me...	500.5	30.6	2.68	105.26
std	288.82	10.82	1.24	43.56
min	1	12	1	30
25%	250.75	21	2	67
50%	500.5	30	3	104
75%	750.25	40	3	143
max	1000	49	5	180

8 rows, 4 cols 10 / page Page 1 of 1

```
Analytical view - categorical columns:
```

	gender object	birthday object	abonoment_type o.	days_per_week o...	fav_group_lesson o	avg_time_check_in	avg_time_check_...	fa
cou...	1000	1000	1000	1000	503	1000	1000	4
uni...	2	974	2	115	253	556	572	3
top	Female	1976-09-28	Standard	Sun	Yoga	17:57:00	14:08:00	c
freq	503	2	507	37	20	6	6	5

4 rows, 9 cols 10 / page Page 1 of 1

```python
# In the code below, a lambda structure was applied to the elements in the ‘days_per_week’ column, organizing the
# days of the week into a list and counting them using the len() function. Finally, the pandas “mean()” function
# was used to calculate the average number of days per week that customers attend the gym.

separated_days= df_membros['days_per_week'].apply(lambda x: len(x.split(",")))
mean_days_per_week= separated_days.mean()
print("Average number of days per week that customers attend the gym:", mean_days_per_week)

# Below, the “mean()” function was used in the ‘visit_per_week’ column, and the result was displayed using the
# “print()” function.

mean_visit_per_week= df_membros['visit_per_week'].mean()
print("Average visits per week:", mean_visit_per_week)

# To calculate the standard deviation of the members' ages, the “std()” function was used in the ‘Age’ column.
# “Round(2)” was used to aid visualization.

std_age= df_membros['Age'].std().round(2)
print("The standard deviation of the age of the members is:", std_age)

# The “nunique()” function is used to retrieve the number of unique favorite drinks mentioned by members.

unique_drinks= df_membros['fav_drink'].nunique()
print("There are", unique_drinks, "different types of drinks.")

# In this block of code, “Counter” was imported to help count the items and used together with a list comprehension
# of 2 “for” loops, first creating a list with each customer's group classes and then creating a list with all
# group classes for all customers. At this point, the “Counter” was called to count how many times each sport
# appears in this final list. The “most_common(1)” function was chosen here just to provide more details.

from collections import Counter
classes_split= df_membros['fav_group_lesson'].dropna().apply(lambda x: x.split(","))
counter_classes= Counter([classe.strip() for sublist in classes_split for classe in sublist])
most_frequent_group= counter_classes.most_common(1)
print("The most popular class group is:", most_frequent_group)

# In this code block, the “Counter” function and a list comprehension of 2 “for” loops on the ‘days_per_week’
# column were also used.

split_days= df_membros['days_per_week'].apply(lambda x: x.split(","))
days_counter= Counter([day.strip() for sublist in split_days for day in sublist])
busiest_day= days_counter.most_common(1)
print("The busiest day is", busiest_day)
```

```
Average number of days per week that customers attend the gym: 2.682
Average visits per week: 2.682
The standard deviation of the age of the members is: 10.82
There are 36 different types of drinks.
The most popular class group is: [('BodyPump', 112)]
The busiest day is [('Sun', 407)]
```

```python
# A new dataframe “df_resumo” was created from the original dataframe “df_membros”.
df_resumo= df_membros[[
    'Age',
    'abonoment_type',
    'personal_training',
    'days_per_week',
    'visit_per_week',
    'avg_time_in_gym'

]]
display(df_resumo)
```

	Age int64	abonoment_type o.	personal_training b	days_per_week o...	visit_per_week in...	avg_time_in_gym i..
	12 - 49	Standard 50.7% Premium 49.3%	True 51.8% False 48.2%	Sun 3.7% Fri 3.4% 113 others 92.9%	1 - 5	30 - 180
0	27	Premium	False	Mon, Sat, Tue, Wed	4	116
1	47	Standard	True	Mon, Sat, Wed	3	48
2	41	Premium	True	Sat	1	123
3	44	Premium	True	Sat, Tue, Wed	3	99
4	44	Standard	True	Thu, Wed	2	50
5	15	Standard	False	Mon	1	180
6	30	Premium	False	Sat, Thu, Wed	3	62
7	20	Standard	False	Mon, Wed	2	95
8	46	Premium	True	Sat, Sun, Thu	3	92
9	24	Premium	True	Mon	1	144

1,000 rows, 6 cols 10 / page Page 1 of 100