PYTHON CODE FOR GENERATING DATA

Let's first import all the libraries we required for generating these 2 dataset.

```
import pandas as pd

import numpy as np

import random

import datetime

from scipy.stats import pearsonr

from scipy.optimize import minimize

from random import randint
```

Now we have to set column name for 1st Dataset(GPS Dataset) and make it pandas Dataframe.

```
columns = ['Date', 'Player_Name', 'Session_Duration', 'Player_Load', 'Number_of_Accelerations',
'Total_Distance', 'Rate_of_Perceived_Exertion']

df = pd.DataFrame(columns=columns)
```

Then we Have to fill Manually our 20 Players in a list.

```
Players = ['Alisson Becker', 'Caoimhin Kelleher', 'Virgil van Dijk', 'Ibrahima Konaté', 'Joe Gomez', 'Kostas
Tsimikas', 'Andrew Robertson', 'Joël Matip', 'Trent Alexander-Arnold', 'Fabinho', 'Thiago', 'Naby Keïta',
'Jordan Henderson', 'Harvey Elliott', 'Alex Oxlade-Chamberlain', 'Roberto Firmino', 'Sadio Mané',
'Mohamed Salah', 'Diogo Jota', 'Divock Origi']
```

We took into consideration for only working days of 1st 10 week of each year starting from 2014. so to make list of desired date we have to write a function.

```
def workdays (start, end, exclude=(6,7)):
    days = []
    while start.date() <= end.date():
        if start.isoweekday() not in exclude:
            days.append(start)
        start += datetime.timedelta(days=1)
    return days
Dates = workdays(datetime.datetime(2014, 1, 1), datetime.datetime(2014, 3, 7))
```

Then we started filling columns using forloop and random libraries.

```
session = [45, 60, 75, 90]

load = list(range(300, 601))

n = 0

for r in range(48):

   Date = Dates[r]

   Session_Duration = random.choice(session)

   for i in range(20):

      player_Name = Players[i]

      Session_Duration = random.choice(session)

      Number_of_Accelerations = random.randint(50, 200)

      Rate_of_Perceived_Exertion = random.randint(1, 10)

      Player_Load = random.randint(300,600)

      df.loc[n] = [Date, player_Name, Session_Duration, Player_Load, Number_of_Accelerations, "NA",
Rate_of_Perceived_Exertion]

      n += 1
```

Colmun Player_Load and Total_Distance are corelated. so we use here personr function to generate 2nd column using corelation value 0.5.

```
def fun(x):

   return abs(0.5 - pearsonr(df['Player_Load'], x)[0])

df['Total_Distance'] = minimize(fun, [ randint(4000, 10000) for _ in range(960)], method = 'SLSQP',
bounds = [(4000, 10000) for _ in range(960)]).x.astype(int)
```

Similar way we can generate our 2nd dataset(WELLNESS Dataset) and save both as csv file.

```
players = ['Alisson Becker_GK', 'Caoimhin Kelleher_GK', 'Virgil van Dijk_CB', 'Ibrahima Konaté_CB', 'Joe
Gomez_CB', 'Kostas Tsimikas_WB', 'Andrew Robertson_WB', 'Joël Matip_CB', 'Trent Alexander-
Arnold_WB', 'Fabinho_CM', 'Thiago_CM', 'Naby Keïta_CM', 'Jordan Henderson_CM', 'Harvey Elliott_CM',
'Alex Oxlade-Chamberlain_CM', 'Roberto Firmino_CF', 'Sadio Mané_CF', 'Mohamed Salah_CF', 'Diogo
Jota_CF', 'Divock Origi_CF']

columns = ['Date', 'Player_Name', 'Sleep_Hours', 'Motivation', 'Soreness']

df1 = pd.DataFrame(columns=columns)

n = 0
```

```
for r in range(48):
    Date = Dates[r]
    for i in range(20):
        player_Name = players[i]
        Sleep_Hours = random.randint(5, 8)
        Motivation = random.randint(1, 10)
        Soreness = random.randint(1, 10)
        df1.loc[n] = [Date, player_Name, Sleep_Hours, Motivation, Soreness]
        n += 1
df.to_csv('GPS2014.csv')
df1.to_csv('WELLNESS2014.csv')
```

As we can see, we generated our two desired dataset for only 2014 year here. Similarly we can generate for 2015, 2016, 2017 and 2018 year.

SQL QUERY

SOURNESS

```
SELECT t.Day, AVG(Soreness)
FROM GPSANDWELLNESS.GPS AS t
JOIN GPSANDWELLNESS.WELLNESS AS ti
ON t.Number = ti.Number
GROUP BY t.Day;
```

TOTAL DISTANCE

```
SELECT t.Player_Name,COUNT(t.Total_Distance), (COUNT(t.Total_Distance) >((75 *(SELECT
COUNT(Player_Name)
                    FROM GPSANDWELLNESS.GPS
                    WHERE Player_Name = "Thiago"))/100) )
FROM GPSANDWELLNESS.GPS AS t
INNER JOIN GPSANDWELLNESS.WELLNESS AS ti
ON t.Number = ti.Number
WHERE t.Total_Distance > 5000 AND ti.Position = 'CF'
GROUP BY t.Player_Name
```

AVERAGE LOAD

```
SELECT ti.Position, AVG(Player_Load)
FROM GPSANDWELLNESS.GPS AS t
JOIN GPSANDWELLNESS.WELLNESS AS ti
ON t.Number = ti.Number
GROUP BY ti.position;
```