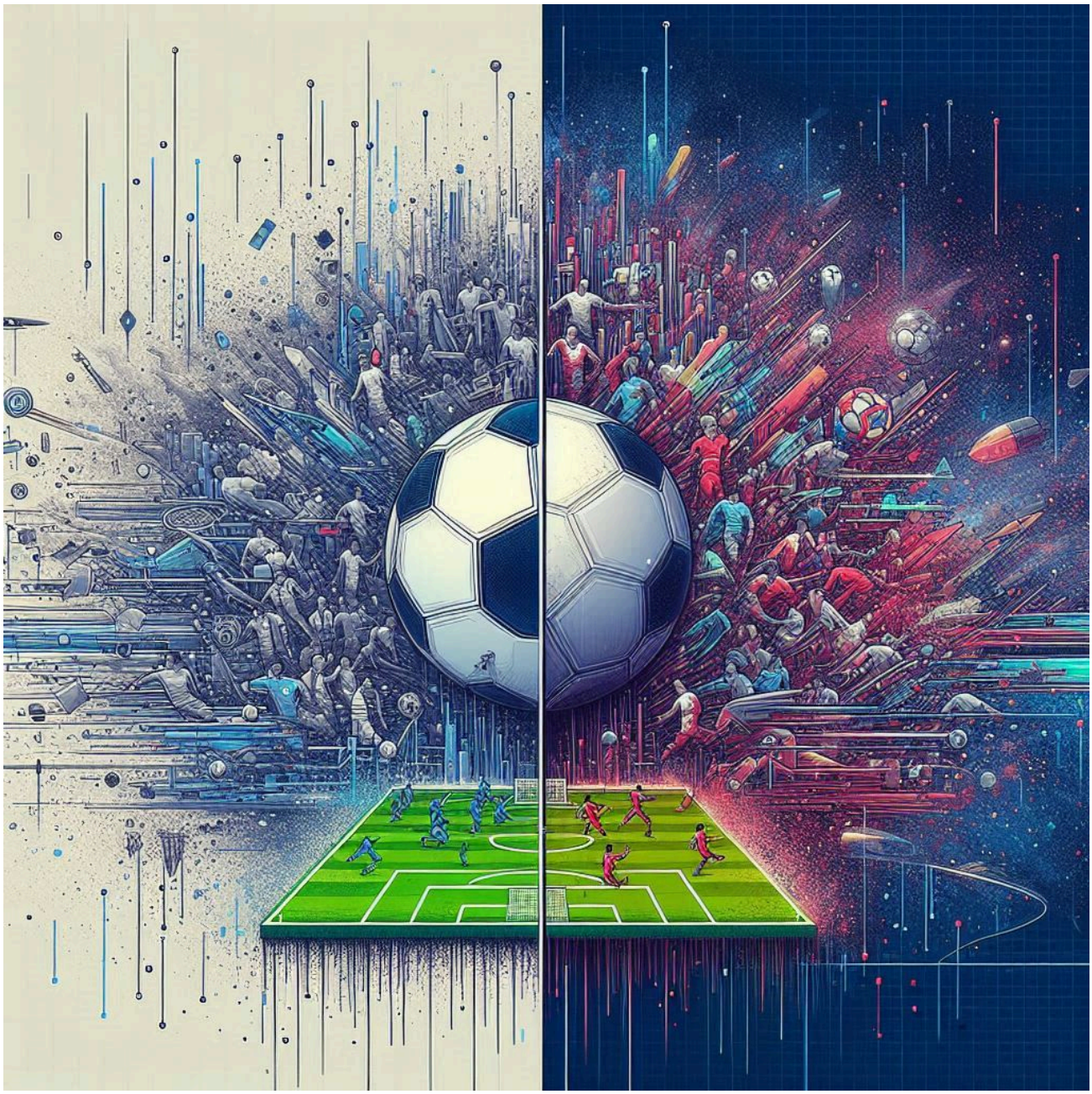
Comprehensive Data Cleaning of FIFA21 Player Attributes



1.0 About Project

- Project: Data Cleaning of FIFA21 Dataset
- Author: Muhammad Waqas
- Author's Contact Info:
  - **Email:** waqasliaqat630@gmailcom
  - Linkedin
  - Github
  - kaggle

2.0 About Data

`Data` : FIFA 21 messy, raw dataset for cleaning/ exploring

Dataset link

3.0 Objective

- The objective of this project is to clean and preprocess the FIFA21 dataset, which was scraped from sofifa.com. The dataset contains messy and raw data, and the goal is to transform it into a clean, consistent, and analysis-ready format. This involves handling missing values, converting data types, cleaning text data, and ensuring overall data quality.

4.0 Import Libraries

- Let's Start the task by importing necessary libraries

In [1]:
```
# For Data Manipulation
import pandas as pd
import numpy as np
# For Data Visualization
import seaborn as sns
# For Ignore Warnings
import warnings
warnings.filterwarnings('ignore')
```

5.0 Increase Readability of DataFrame

- As data have much features, we will increase readability of data for better understanding.

In [2]:
```
# To display rows and columns at maximum
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
```

6.0 Load Dataset

In [3]:
```
# Load Dataset
df_original=pd.read_csv("../data/raw/fifa21_raw_data.csv")

# Copy Dataset
df=df_original.copy()
```

7.0 Overview Of Dataset

- Having some understanding of data is necessary to know what should be done.

In [4]:
```
# First 5 Rows
df.head()
```

Out[4]:

	photoUrl	LongName	playerUrl	Na
0	https://cdn.sofifa.com/players/158/023/21_60.png	Lionel Messi	http://sofifa.com/player/158023/lionel-messi/2...	A
1	https://cdn.sofifa.com/players/020/801/21_60.png	C. Ronaldo dos Santos Aveiro	http://sofifa.com/player/20801/c-ronaldo-dos-s...	
2	https://cdn.sofifa.com/players/200/389/21_60.png	Jan Oblak	http://sofifa.com/player/200389/jan-oblak/210005/	
3	https://cdn.sofifa.com/players/192/985/21_60.png	Kevin De Bruyne	http://sofifa.com/player/192985/kevin-de-bruyn...	
4	https://cdn.sofifa.com/players/190/871/21_60.png	Neymar da Silva Santos Jr.	http://sofifa.com/player/190871/neymar-da-silv...	

In [5]:

```
# Information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18979 entries, 0 to 18978
Data columns (total 77 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   photoUrl          18979 non-null  object
 1   LongName          18979 non-null  object
 2   playerUrl         18979 non-null  object
 3   Nationality       18979 non-null  object
 4   Positions         18979 non-null  object
 5   Name              18979 non-null  object
 6   Age               18979 non-null  int64
 7   ↓OVA              18979 non-null  int64
 8   POT               18979 non-null  int64
 9   Team & Contract   18979 non-null  object
 10  ID                18979 non-null  int64
 11  Height            18979 non-null  object
 12  Weight            18979 non-null  object
 13  foot              18979 non-null  object
 14  BOV               18979 non-null  int64
 15  BP                18979 non-null  object
 16  Growth            18979 non-null  int64
 17  Joined            18979 non-null  object
 18  Loan Date End     1013 non-null   object
 19  Value             18979 non-null  object
 20  Wage              18979 non-null  object
 21  Release Clause    18979 non-null  object
 22  Attacking         18979 non-null  int64
 23  Crossing          18979 non-null  int64
 24  Finishing         18979 non-null  int64
 25  Heading Accuracy  18979 non-null  int64
 26  Short Passing     18979 non-null  int64
 27  Volleys           18979 non-null  int64
 28  Skill             18979 non-null  int64
 29  Dribbling         18979 non-null  int64
 30  Curve             18979 non-null  int64
 31  FK Accuracy       18979 non-null  int64
 32  Long Passing      18979 non-null  int64
 33  Ball Control      18979 non-null  int64
 34  Movement          18979 non-null  int64
 35  Acceleration      18979 non-null  int64
 36  Sprint Speed      18979 non-null  int64
 37  Agility           18979 non-null  int64
 38  Reactions         18979 non-null  int64
 39  Balance           18979 non-null  int64
 40  Power             18979 non-null  int64
 41  Shot Power        18979 non-null  int64
 42  Jumping           18979 non-null  int64
 43  Stamina           18979 non-null  int64
 44  Strength          18979 non-null  int64
 45  Long Shots        18979 non-null  int64
 46  Mentality         18979 non-null  int64
 47  Aggression        18979 non-null  int64
 48  Interceptions     18979 non-null  int64
 49  Positioning       18979 non-null  int64
 50  Vision            18979 non-null  int64
 51  Penalties         18979 non-null  int64
 52  Composure         18979 non-null  int64
 53  Defending         18979 non-null  int64
 54  Marking           18979 non-null  int64
 55  Standing Tackle   18979 non-null  int64
 56  Sliding Tackle    18979 non-null  int64
 57  Goalkeeping       18979 non-null  int64
 58  GK Diving         18979 non-null  int64
 59  GK Handling       18979 non-null  int64
 60  GK Kicking        18979 non-null  int64
```

```
 61  GK Positioning     18979 non-null  int64
 62  GK Reflexes        18979 non-null  int64
 63  Total Stats        18979 non-null  int64
 64  Base Stats         18979 non-null  int64
 65  W/F                18979 non-null  object
 66  SM                 18979 non-null  object
 67  A/W                18979 non-null  object
 68  D/W                18979 non-null  object
 69  IR                 18979 non-null  object
 70  PAC                18979 non-null  int64
 71  SHO                18979 non-null  int64
 72  PAS                18979 non-null  int64
 73  DRI                18979 non-null  int64
 74  DEF                18979 non-null  int64
 75  PHY                18979 non-null  int64
 76  Hits               18979 non-null  object
dtypes: int64(55), object(22)
memory usage: 11.1+ MB
```

In [6]:
```
# Summary Statistics
df.describe()
```

Out[6]:

	Age	↓OVA	POT	ID	BOV	Growth	Attac
count	18979.000000	18979.000000	18979.000000	18979.000000	18979.000000	18979.000000	18979.00
mean	25.194583	65.718636	71.136098	226404.790242	66.751620	5.417461	248.93
std	4.710753	6.968999	6.114176	27141.673349	6.747017	5.663954	74.29
min	16.000000	47.000000	47.000000	41.000000	48.000000	0.000000	42.00
25%	21.000000	61.000000	67.000000	210135.000000	62.000000	0.000000	222.00
50%	25.000000	66.000000	71.000000	232424.000000	67.000000	4.000000	263.00
75%	29.000000	70.000000	75.000000	246925.500000	71.000000	9.000000	297.00
max	53.000000	93.000000	95.000000	259216.000000	93.000000	26.000000	437.00

Observation:

- Height and Weight should be converted to integer.
- Joined date should be in Proper DateTime format.
- Wage,Release Clause, Value, Hits should be cleaned and be in integer format.
- SM,IR and W/F should get rid of star and be in proper best suggested DataType.
- Term and Contract should be separated and in proper format.

8.0 Data Cleaning

8.1 Missing Values

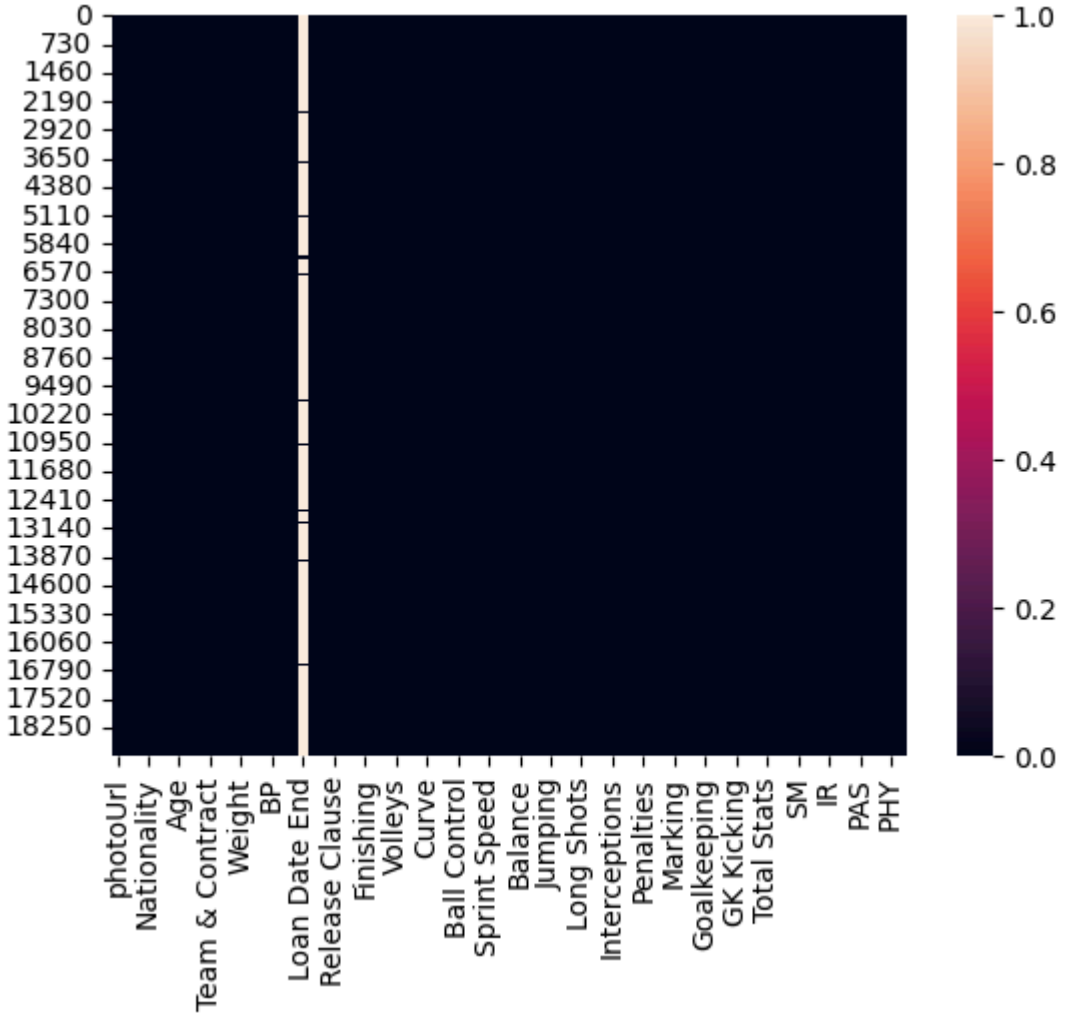
- Missing values can lead to biased or inaccurate analysis, making it essential to treat them to ensure data integrity and reliable results.

In [7]:
```
# Find Percentage of Null Values
(df.isnull().sum()/len(df)*100).sort_values(ascending=False).head()
```

Out[7]:
```
Loan Date End      94.662522
photoUrl            0.000000
GK Diving           0.000000
Sliding Tackle      0.000000
Standing Tackle     0.000000
dtype: float64
```

In [8]:
```
# Heatmap For Missing Values
sns.heatmap(df.isnull())
```

Out[8]:
```
<Axes: >
```



- As NULL Loan Date End values means player was not on Loan, So we will fill it with "Not Present".

In [9]:
```
df["Loan Date End"].fillna("Not Present",inplace=True)
```

8.2 Duplicates Treatment

- It is essential to treat duplicates to ensure data integrity and reliability and to avoid bias.

In [10]:
```
# Checking Number of Duplicates
df.duplicated().sum()
```

Out[10]:
```
1
```

In [11]:
```
df[df.duplicated()]
```

Out[11]:

	photoUrl	LongName	playerUrl	Na
944	https://cdn.sofifa.com/players/251/698/21_60.png	Kevin Berlaso	http://sofifa.com/player/251698/kevin-berlaso/...	

In [12]:
```
df[df['LongName']=='Kevin Berlaso']
```

Out[12]:

	photoUrl	LongName	playerUrl	Na
899	https://cdn.sofifa.com/players/251/698/21_60.png	Kevin Berlaso	http://sofifa.com/player/251698/kevin-berlaso/...	
944	https://cdn.sofifa.com/players/251/698/21_60.png	Kevin Berlaso	http://sofifa.com/player/251698/kevin-berlaso/...	

In [13]:
```
# Drop Duplicates
df.drop_duplicates(inplace=True)
```

In [14]:
```
# Checking Number of Duplicates
df.duplicated().sum()
```

Out[14]: 0

- All Duplicates have been removed.

8.3 Inconsistencies Treatment

- Addressing data inconsistencies is essential for accurately understanding and analyzing all features within the dataset.

8.3.1 Giving Height and Weight appropriate Data Types

In [15]:
```
# Head of Height and Weight
df[["Height","Weight"]].head()
```

Out[15]:

	Height	Weight
0	5'7"	159lbs
1	6'2"	183lbs
2	6'2"	192lbs
3	5'11"	154lbs
4	5'9"	150lbs

- **Weight**

In [16]:
```
# Removing lbs from Weight
df["Weight"]=df["Weight"].str.replace("lbs","")
```

```
# renaming weight
df.rename(columns={"Weight":"Weight(lbs)"},inplace=True)
# Changing datatpe of Weight to INT
df["Weight(lbs)"]=df["Weight(lbs)"].astype(int)
```

- **Height**

In [17]:
```
# Removing " from height string
df["Height"]=df["Height"].str.replace("\"","")
# Splitting Height into feet and inches
df[["Height in Feets","Height in Inches"]]=df["Height"].str.split("'",expand=True)
# Making height in integers
df["Height in Feets"]=df["Height in Feets"].astype(int)
df["Height in Inches"]=df["Height in Inches"].astype(int)
# Calculating total height in inches
df["Height in Inches"]=((df["Height in Feets"]*12)+df["Height in Inches"]).astype(int)
# Dropping rest of the two features
df.drop(columns=["Height","Height in Feets"],inplace=True)
```

In [18]:
```
# Information of Height and Weight
df[["Height in Inches","Weight(lbs)"]].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 18978 entries, 0 to 18978
Data columns (total 2 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Height in Inches  18978 non-null  int32
 1   Weight(lbs)       18978 non-null  int32
dtypes: int32(2)
memory usage: 296.5 KB
```

8.3.2 Converting joined date into datetime

- Having Datetime data in Proper Format is essential for analysis.

In [19]:
```
df['Joined'].head()
```

Out[19]:
```
0     Jul 1, 2004
1    Jul 10, 2018
2    Jul 16, 2014
3    Aug 30, 2015
4     Aug 3, 2017
Name: Joined, dtype: object
```

In [20]:
```
# Converting Joined o Datetime
df["Joined"]=pd.to_datetime(df["Joined"],format="%b %d, %Y")
```

In [21]:
```
df['Joined'].head()
```

Out[21]:
```
0   2004-07-01
1   2018-07-10
2   2014-07-16
3   2015-08-30
4   2017-08-03
Name: Joined, dtype: datetime64[ns]
```

In [22]:
```
# Information of Joined
df["Joined"].info()
```

```
<class 'pandas.core.series.Series'>
Index: 18978 entries, 0 to 18978
Series name: Joined
Non-Null Count  Dtype
--------------  -----
18978 non-null  datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 296.5 KB
```

8.3.3 Converting Wage Feature to integer

In [23]:
```
df["Wage"].head()
```

Out[23]:
```
0    €560K
1    €220K
2    €125K
3    €370K
4    €270K
Name: Wage, dtype: object
```

In [24]:
```
df["Wage"].isnull().sum()
```

Out[24]:
```
0
```

In [25]:
```
# Removing €
df["Wage"]=df["Wage"].str.replace("€","")
```

In [26]:
```
# Total length of dataset
print(f"Total length of dataset is: {len(df)}")
having_k=df[df["Wage"].str.contains("K")].shape[0]
print(f"Number of rows having K is: {having_k}")
except_k=len(df)-having_k
```

```
Total length of dataset is: 18978
Number of rows having K is: 14824
Number of rows having K is: 14824
```

In [27]:
```
# No Salary in Million Spotted
having_k=df[df["Wage"].str.contains("M")].shape[0]
having_k
```

Out[27]:
```
0
```

In [28]:
```
# Gettig data not having k
without_k=df[~df["Wage"].str.contains("K")]
# checking if it is convertable to integer
without_k["Wage"]=without_k['Wage'].astype(int)
```

- As wages without K are completely convertable to integer, It means they are less than thousand.
- We will convert it to format of thousand.

In [29]:
```
# Removing data not having wages in k
df=df[df["Wage"].str.contains("K")]
df["Wage"]=df["Wage"].str.replace("K","")
df["Wage"]=df["Wage"].astype(float)
df["Wage"]=df["Wage"]*1000
```

In [30]:
```
# Combining Both
df=pd.concat([df,without_k],axis=0)
df["Wage"].sample(5)
```

Out[30]:
```
17638     3000.0
7266       500.0
16169     4000.0
3236      3000.0
2197     11000.0
Name: Wage, dtype: float64
```

In [31]:
```
# Renaming Wages
df.rename(columns={"Wage":"Wage(€)"},inplace=True)
# Making wage to integer
df["Wage(€)"]=df["Wage(€)"].astype(float)
```

In [32]:
```
df["Wage(€)"].isnull().sum()
```

Out[32]:
```
0
```

In [33]:
```
df["Wage(€)"].info()
```

```
<class 'pandas.core.series.Series'>
Index: 18978 entries, 0 to 18978
Series name: Wage(€)
Non-Null Count  Dtype
--------------  -----
18978 non-null  float64
dtypes: float64(1)
memory usage: 296.5 KB
```

8.3.4 Converting Release Clause to integer

In [34]:
```
df["Release Clause"].head()
```

Out[34]:
```
0    €138.4M
1     €75.9M
2    €159.4M
3      €161M
4    €166.5M
Name: Release Clause, dtype: object
```

In [35]:
```
# Removing €
df['Release Clause']=df["Release Clause"].str.replace('€','')
df['Release Clause'].head()
```

Out[35]:
```
0    138.4M
1     75.9M
2    159.4M
3      161M
4    166.5M
Name: Release Clause, dtype: object
```

In [36]:
```
# Cecking if dataset have Release Clause in K
df[df["Release Clause"].str.contains("K")].head()
```

Out[36]:

	photoUrl	LongName	playerU
2055	https://cdn.sofifa.com/players/156/433/21_60.png	Alfredo Talavera	http://sofifa.com/player/156433/alfred talave
2293	https://cdn.sofifa.com/players/165/769/21_60.png	Cássio Albuquerque Anjos	http://sofifa.com/player/165769/cassi albuque
2585	https://cdn.sofifa.com/players/107/298/21_60.png	Yohann Pelé	http://sofifa.com/player/107298/yohan pele/21
2651	https://cdn.sofifa.com/players/049/472/21_60.png	Ludovic Butelle	http://sofifa.com/player/49472/ludovi butelle
2768	https://cdn.sofifa.com/players/216/692/21_60.png	Sebastián Torrico	http://sofifa.com/player/216692/sebastia tor

In [37]:

```
print(f"Total Length of dataset is: {len(df)}")
havingk=df["Release Clause"].str.contains("K").sum()
havingm=df["Release Clause"].str.contains("M").sum()
k_m=havingk+havingm
print(f"Number of rows having K and M is: {len(df)-k_m}")
```

```
Total Length of dataset is: 18978
Number of rows having K and M is: 1261
```

In [38]:

```
without_k_m=df[(~df["Release Clause"].str.contains("K")) & (~df["Release Clause"].str.contain
without_k_m.head(3)
```

Out[38]:

	photoUrl	LongName	playerUrl
205	https://cdn.sofifa.com/players/173/731/21_60.png	Gareth Bale	http://sofifa.com/player/173731/gareth-bale/21...
250	https://cdn.sofifa.com/players/200/888/21_60.png	Danilo Luís Hélio Pereira	http://sofifa.com/player/200888/danilo-luis-he...
257	https://cdn.sofifa.com/players/193/105/21_60.png	Alphonse Areola	http://sofifa.com/player/193105/alphonse-areol...

In [39]:

```
len(without_k_m[without_k_m["Release Clause"]=='0'])
```

Out[39]: 1261

In [40]:

```
len(without_k_m)
```

Out[40]: 1261

- It means Release Clause is in millions, thousands and zero.

- Currency Covertor function

In [41]:
```
#let's define a function
def currency_convertor_M(datat):
    if 'K' in datat:
        return float(datat.replace('K',''))*1000
    elif 'M' in datat:
        return float(datat.replace('M',''))*1000000
    else:
        return float(datat)
```

In [42]:
```
having_k=df["Release Clause"].str.contains("K").sum()
having_m=df["Release Clause"].str.contains("M").sum()
print(having_k+having_m)
nothavingmk=len(df["Release Clause"])-(having_k+having_m)
```

```
17717
```

In [43]:
```
df[(~df["Release Clause"].str.contains("K"))& (~df["Release Clause"].str.contains("M"))].head
```

Out[43]:

	photoUrl	LongName	playerUrl
205	https://cdn.sofifa.com/players/173/731/21_60.png	Gareth Bale	http://sofifa.com/player/173731/gareth-bale/21...
250	https://cdn.sofifa.com/players/200/888/21_60.png	Danilo Luís Hélio Pereira	http://sofifa.com/player/200888/danilo-luis-he...
257	https://cdn.sofifa.com/players/193/105/21_60.png	Alphonse Areola	http://sofifa.com/player/193105/alphonse-areol...

In [44]:
```
having_0=df["Release Clause"].str.contains("€0").sum()
print(having_0)
print(nothavingmk)
```

```
0
1261
```

- It means Release Clause is in millions, thousands and zero.

In [45]:
```
df['Release Clause']=df['Release Clause'].apply(currency_convertor_M)
# Renaming Release Clause
df.rename(columns={"Release Clause":"Release Clause(€)"},inplace=True)
```

In [46]:
```
df["Release Clause(€)"].head()
```

Out[46]:
```
0    138400000.0
1     75900000.0
2    159400000.0
3    161000000.0
4    166500000.0
Name: Release Clause(€), dtype: float64
```

In [47]:
```
df["Release Clause(€)"].info()
```

```
<class 'pandas.core.series.Series'>
Index: 18978 entries, 0 to 18978
Series name: Release Clause(€)
Non-Null Count  Dtype
--------------  -----
18978 non-null  float64
dtypes: float64(1)
memory usage: 296.5 KB
```

8.3.5 Converting Hits to Integer

In [48]:
```
df["Hits"].isnull().sum()
```

Out[48]:
```
0
```

In [49]:
```
df["Hits"].sample(5)
```

Out[49]:
```
11273     \n39
7389      \n13
4984     \n168
6955       \n4
8773       \n2
Name: Hits, dtype: object
```

In [50]:
```
df["Hits"] = df["Hits"].astype(str).str.replace("\n", "")
```

In [51]:
```
df['Hits'].sample(5)
```

Out[51]:
```
11144    2
7943     6
12379    4
17629    1
13652    2
Name: Hits, dtype: object
```

In [52]:
```
print(df["Hits"].str.contains("K").sum())
```

```
15
```

In [53]:
```
df.loc[df["Hits"].str.contains("K", na=False), "Hits"] = (
    df["Hits"].str.replace("K", "", regex=True).astype(float) * 1000
)
```

In [54]:
```
df["Hits"].info()
```

```
<class 'pandas.core.series.Series'>
Index: 18978 entries, 0 to 18978
Series name: Hits
Non-Null Count  Dtype
--------------  -----
18978 non-null  object
dtypes: object(1)
memory usage: 812.6+ KB
```

8.3.6 Value

In [55]:
```
df["Value"].isnull().sum()
```

Out[55]:
```
0
```

In [56]:
```
df["Value"].head()
```

Out[56]:
```
0    €67.5M
1      €46M
2      €75M
3      €87M
4      €90M
Name: Value, dtype: object
```

In [57]:
```
df["Value"]=df["Value"].str.replace("€","")
df["Value"]=df["Value"].apply(currency_convertor_M)
df.rename(columns={"Value":"Value(€)"},inplace=True)
```

8.3.7 Team & Contract Feature

In [58]:
```
# sneak peak
df["Team & Contract"].head()
```

Out[58]:
```
0           \n\n\n\nFC Barcelona\n2004 ~ 2021\n\n
1               \n\n\n\nJuventus\n2018 ~ 2022\n\n
2        \n\n\n\nAtlético Madrid\n2014 ~ 2023\n\n
3        \n\n\n\nManchester City\n2015 ~ 2023\n\n
4    \n\n\n\nParis Saint-Germain\n2017 ~ 2022\n\n
Name: Team & Contract, dtype: object
```

In [59]:
```
df["Team & Contract"]=df["Team & Contract"].str.replace("\n\n\n\n","").str.replace("\n\n","")
df["Team & Contract"].head()
```

Out[59]:
```
0           FC Barcelona\n2004 ~ 2021
1               Juventus\n2018 ~ 2022
2        Atlético Madrid\n2014 ~ 2023
3        Manchester City\n2015 ~ 2023
4    Paris Saint-Germain\n2017 ~ 2022
Name: Team & Contract, dtype: object
```

In [60]:
```
with_multiple_n=len(df[df["Team & Contract"].str.count("\n")>1])
print(f"Number of entries with Multiple n\ {with_multiple_n}")
with_multiple_n=len(df[df["Team & Contract"].str.count("\n")==2])
print(f"Number of entries with two n\{with_multiple_n}")
```

```
Number of entries with Multiple n\ 237
Number of entries with two n\237
```

In [61]:
```
df[df["Team & Contract"].str.count("\n")>1]["Team & Contract"].sample(3)
```

Out[61]:
```
1307    \n Paraguay\nFree
4993     \n Iceland\nFree
1579     \n Ecuador\nFree
Name: Team & Contract, dtype: object
```

- After Watching a lot of Samples wih multiple \n, we can assume that extra \n is present at start.
- So we will remove first \n of these.
- If it is not True, it will generate error while typecasting.

In [62]:
```
# Removing \n of only those points having multiple \n
df.loc[df["Team & Contract"].str.count("\n")>1, "Team & Contract"] = df["Team & Contract"].st
# Splitting Team and Contract
df[["Team","Contract"]] = df["Team & Contract"].str.split("\n", expand=True)
```

- As anticipated, our analysis aligns with our expectations😉.

In [63]:
```
# Dropping Team and Contract
df.drop(columns=["Team & Contract"],inplace=True)
```

8.3.8 SM and W/F and IR

- Just by removing ★, we can have pure integer format data.

In [64]:
```
# Removing ★ from SM and W/F and IR
df["SM"]=df["SM"].str.replace("★","")
df["W/F"]=df["W/F"].str.replace(" ★","")
df["IR"]=df["IR"].str.replace("★","")
```

In [65]:
```
# Converting SM and W/F and IR to Integer
df["SM"]=df["SM"].astype(int)
df["W/F"]=df["W/F"].astype(int)
df["IR"]=df["IR"].astype(int)
```

In [66]:
```
df[["SM", "W/F", "IR"]].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 18978 entries, 0 to 18978
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   SM      18978 non-null  int32
 1   W/F     18978 non-null  int32
 2   IR      18978 non-null  int32
dtypes: int32(3)
memory usage: 886.7 KB
```

- 🎉Fantastic! Data cleaning is now finished. Time to validate and verify our work.

09 Validation and Verification

9.1 Checking Data Types of Updated Features

In [67]:
```
# Checking Datatypes
df.dtypes
```

Out[67]:

photoUrl	object
LongName	object
playerUrl	object
Nationality	object
Positions	object
Name	object
Age	int64
↓OVA	int64
POT	int64
ID	int64
Weight(lbs)	int32
foot	object
BOV	int64
BP	object
Growth	int64
Joined	datetime64[ns]
Loan Date End	object
Value(€)	float64
Wage(€)	float64
Release Clause(€)	float64
Attacking	int64
Crossing	int64
Finishing	int64
Heading Accuracy	int64
Short Passing	int64
Volleys	int64
Skill	int64
Dribbling	int64
Curve	int64
FK Accuracy	int64
Long Passing	int64
Ball Control	int64
Movement	int64
Acceleration	int64
Sprint Speed	int64
Agility	int64
Reactions	int64
Balance	int64
Power	int64
Shot Power	int64
Jumping	int64
Stamina	int64
Strength	int64
Long Shots	int64
Mentality	int64
Aggression	int64
Interceptions	int64
Positioning	int64
Vision	int64
Penalties	int64
Composure	int64
Defending	int64
Marking	int64
Standing Tackle	int64
Sliding Tackle	int64
Goalkeeping	int64
GK Diving	int64
GK Handling	int64
GK Kicking	int64
GK Positioning	int64
GK Reflexes	int64
Total Stats	int64
Base Stats	int64
W/F	int32
SM	int32

```
A/W                         object
D/W                         object
IR                           int32
PAC                          int64
SHO                          int64
PAS                          int64
DRI                          int64
DEF                          int64
PHY                          int64
Hits                        object
Height in Inches             int32
Team                        object
Contract                    object
dtype: object
```

9.2 Checking for Missing Values of Updated Features

In [68]:
```
# Find Percentage of Null Values
df.isnull().sum().sort_values(ascending=False).head()
```

Out[68]:
```
photoUrl          0
Penalties         0
GK Diving         0
Goalkeeping       0
Sliding Tackle    0
dtype: int64
```

9.3 Information about dataset

In [69]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 18978 entries, 0 to 18978
Data columns (total 78 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   photoUrl           18978 non-null  object
 1   LongName           18978 non-null  object
 2   playerUrl          18978 non-null  object
 3   Nationality        18978 non-null  object
 4   Positions          18978 non-null  object
 5   Name               18978 non-null  object
 6   Age                18978 non-null  int64
 7   ↓OVA               18978 non-null  int64
 8   POT                18978 non-null  int64
 9   ID                 18978 non-null  int64
 10  Weight(lbs)        18978 non-null  int32
 11  foot               18978 non-null  object
 12  BOV                18978 non-null  int64
 13  BP                 18978 non-null  object
 14  Growth             18978 non-null  int64
 15  Joined             18978 non-null  datetime64[ns]
 16  Loan Date End      18978 non-null  object
 17  Value(€)           18978 non-null  float64
 18  Wage(€)            18978 non-null  float64
 19  Release Clause(€)  18978 non-null  float64
 20  Attacking          18978 non-null  int64
 21  Crossing           18978 non-null  int64
 22  Finishing          18978 non-null  int64
 23  Heading Accuracy   18978 non-null  int64
 24  Short Passing      18978 non-null  int64
 25  Volleys            18978 non-null  int64
 26  Skill              18978 non-null  int64
 27  Dribbling          18978 non-null  int64
 28  Curve              18978 non-null  int64
 29  FK Accuracy        18978 non-null  int64
 30  Long Passing       18978 non-null  int64
 31  Ball Control       18978 non-null  int64
 32  Movement           18978 non-null  int64
 33  Acceleration       18978 non-null  int64
 34  Sprint Speed       18978 non-null  int64
 35  Agility            18978 non-null  int64
 36  Reactions          18978 non-null  int64
 37  Balance            18978 non-null  int64
 38  Power              18978 non-null  int64
 39  Shot Power         18978 non-null  int64
 40  Jumping            18978 non-null  int64
 41  Stamina            18978 non-null  int64
 42  Strength           18978 non-null  int64
 43  Long Shots         18978 non-null  int64
 44  Mentality          18978 non-null  int64
 45  Aggression         18978 non-null  int64
 46  Interceptions      18978 non-null  int64
 47  Positioning        18978 non-null  int64
 48  Vision             18978 non-null  int64
 49  Penalties          18978 non-null  int64
 50  Composure          18978 non-null  int64
 51  Defending          18978 non-null  int64
 52  Marking            18978 non-null  int64
 53  Standing Tackle    18978 non-null  int64
 54  Sliding Tackle     18978 non-null  int64
 55  Goalkeeping        18978 non-null  int64
 56  GK Diving          18978 non-null  int64
 57  GK Handling        18978 non-null  int64
 58  GK Kicking         18978 non-null  int64
 59  GK Positioning     18978 non-null  int64
 60  GK Reflexes        18978 non-null  int64
```

```
 61  Total Stats         18978 non-null  int64
 62  Base Stats          18978 non-null  int64
 63  W/F                 18978 non-null  int32
 64  SM                  18978 non-null  int32
 65  A/W                 18978 non-null  object
 66  D/W                 18978 non-null  object
 67  IR                  18978 non-null  int32
 68  PAC                 18978 non-null  int64
 69  SHO                 18978 non-null  int64
 70  PAS                 18978 non-null  int64
 71  DRI                 18978 non-null  int64
 72  DEF                 18978 non-null  int64
 73  PHY                 18978 non-null  int64
 74  Hits                18978 non-null  object
 75  Height in Inches    18978 non-null  int32
 76  Team                18978 non-null  object
 77  Contract            18978 non-null  object
dtypes: datetime64[ns](1), float64(3), int32(5), int64(55), object(14)
memory usage: 11.6+ MB
```

9.4 Sneaking Peek of Cleaed Data

In [70]:
```
df.head()
```

Out[70]:

	photoUrl	LongName	playerUrl	Na
0	https://cdn.sofifa.com/players/158/023/21_60.png	Lionel Messi	http://sofifa.com/player/158023/lionel-messi/2...	A
1	https://cdn.sofifa.com/players/020/801/21_60.png	C. Ronaldo dos Santos Aveiro	http://sofifa.com/player/20801/c-ronaldo-dos-s...	
2	https://cdn.sofifa.com/players/200/389/21_60.png	Jan Oblak	http://sofifa.com/player/200389/jan-oblak/210005/	
3	https://cdn.sofifa.com/players/192/985/21_60.png	Kevin De Bruyne	http://sofifa.com/player/192985/kevin-de-bruyn...	
4	https://cdn.sofifa.com/players/190/871/21_60.png	Neymar da Silva Santos Jr.	http://sofifa.com/player/190871/neymar-da-silv...	

9.5 Saving Cleaned Data

In [71]:
```
#df.to_csv("../data/processed/fifa21_clean_data.csv",index=False)
```

10 Conclusion / Summary

In this data cleaning project, I performed several key transformations to ensure that the FIFA21 dataset is clean, consistent, and ready for analysis:

Height and Weight Conversion:

The Height column was converted to inches after removing extra commas and performing necessary calculations. The Weight column was converted to an integer by removing the "lbs" suffix from the string values. Monetary Values Transformation:

The Wage, Release Clause, Hits, and Value columns were converted to integers by removing the pound sign (£) and appropriately multiplying the values by 1,000 or 1,000,000 to account for 'k' and 'm' suffixes. Date Conversion:

The Joined column was typecasted to a datetime format to standardize and facilitate date-related analyses. Feature Splitting:

The Team and Contract column was split into two separate features: Team and Contract, providing clearer insights into the player's affiliations. Star Ratings Conversion:

The columns SM (Skill Moves), W/F (Weak Foot), and IR (Injury Rating) were converted to integers by removing star symbols. These transformations were validated through various checks to ensure data integrity and consistency, resulting in a dataset that is well-prepared for subsequent analysis or modeling.