**Data Cleaning: FIFA-21 Dataset**

****

* Introduction:

A data cleaning task was introduced using the hashtag #datacleaningchallenge to allow beginners, intermediate and pro data analyst in the tech space to exercise their skills using any tool of their choice.

My preferred tool for this data cleaning is Power Query on Power BI.

* About the Dataset:



The dataset used for this challenge is the FIFA-U21 data. The dataset was sourced from [Kaggle.com](https://www.kaggle.com/datasets/yagunnersya/fifa-21-messy-raw-dataset-for-cleaning-exploring). The dataset was obtained in its raw state after web scrapping from sofifa.com. It contains the details of football players alongside the performance.

The dataset contains 18979 rows and 77 columns. The datafile also include a data dictionary to further give the data analyst more insight on the player information.

* Data Cleaning Process:

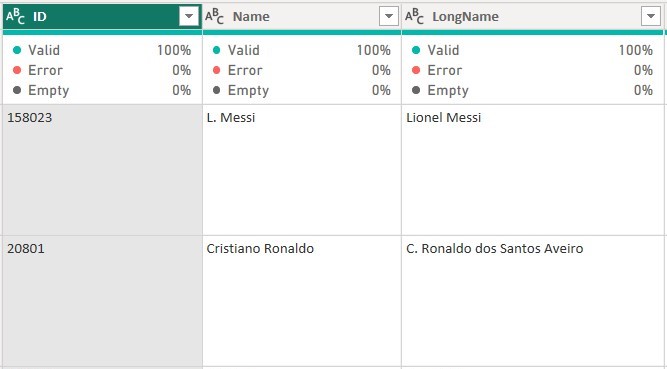


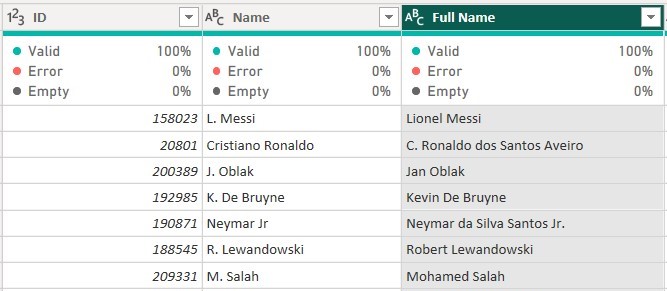
The dataset was imported into the power query editor and loaded for transformation.

Walk with me as I take you through the step-by-step process of cleaning the dataset from its messy state to a clean state ready to use for analysis.

* ID, Name & Long Name:

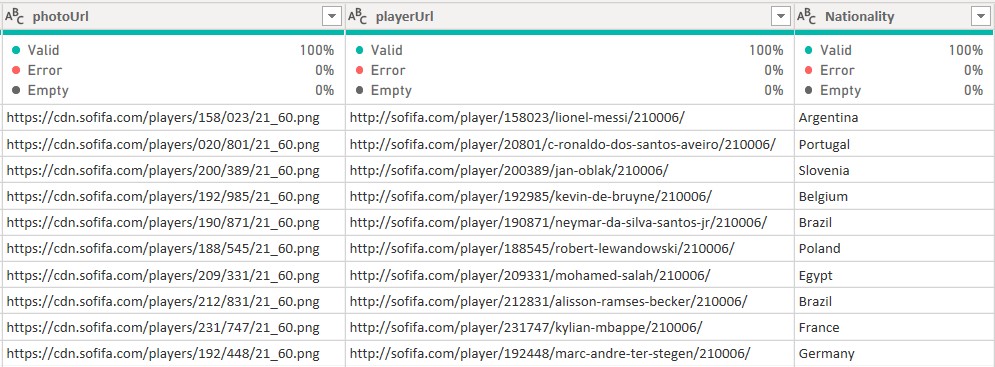
The ID column contains the unique number assigned to every player. The data type was converted to numbers. The ‘Name’ column contains the players short name while the ‘Long Name’ column contains the players full name. The column name was changed to ‘Full Name’.





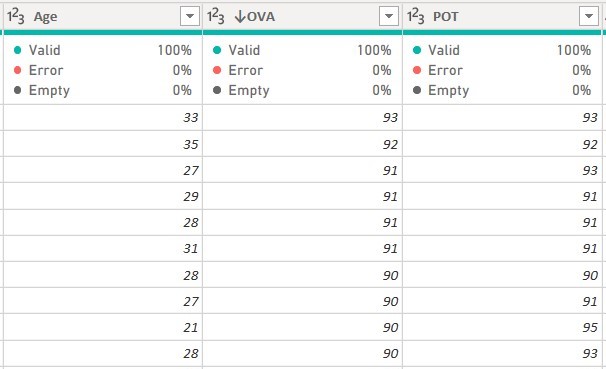
* PhotoURL, PlayerURL & Nationality:

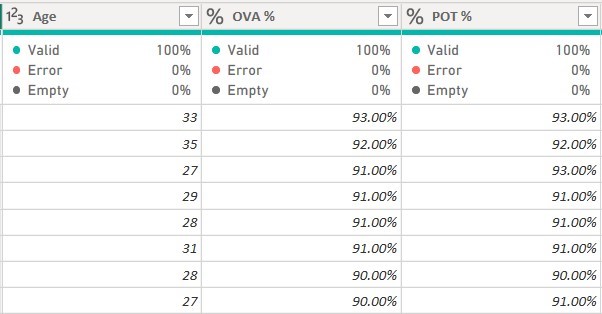
The ‘PhotoUrl’ and ‘PlayerUrl’ columns contains the players’ photo and full names embedded in its URL, while the ‘Nationality’ column contain the players country of origin. The decision to retain the ‘PhotoUrl & PlayerUrl’ column was based on the purpose of this challenge which was to clean the data.



* Age, OVA & POT:

The ‘Age’ column contains the age of individual players. The column was converted to number data type. The Overall Players’ Analysis (OVA) and Players Potential (POT) column were converted to percentage as written in the data dictionary. A custom column was used to multiply the value by 0.01 (100) to give the answer in percentage.





* Club:

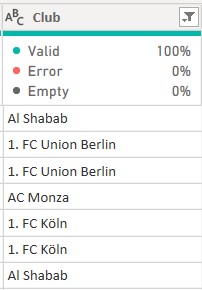
The column consists of inconsistent data entry. Some of the club’s name had numerical prefix attached to their names, like 1.FC Koln. The prefix was removed from the data.

After Cleaning

Table

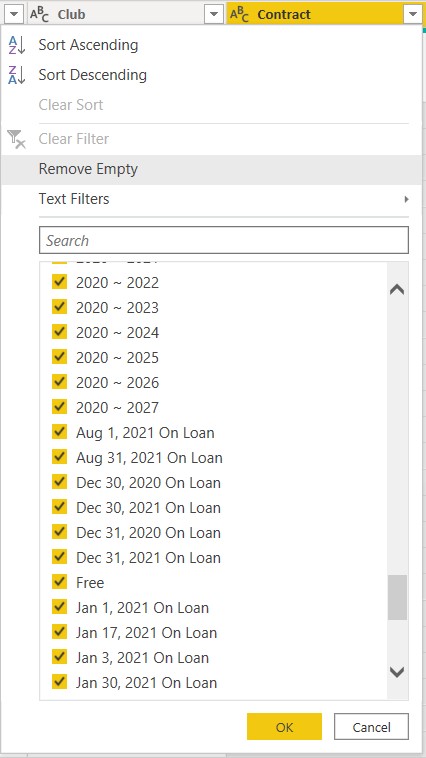
Description automatically generated with medium confidence

Before Cleaning



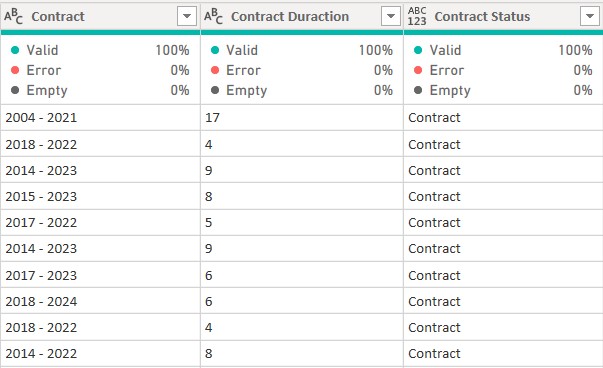
* Contract:

Filtering the ‘Contract’ column, I realized it contained inconsistent values. It contains values in 3 format, 2004~2021, free and 23rd July 2020 on loan. Thus, conditional formatting was used to achieve clean data.



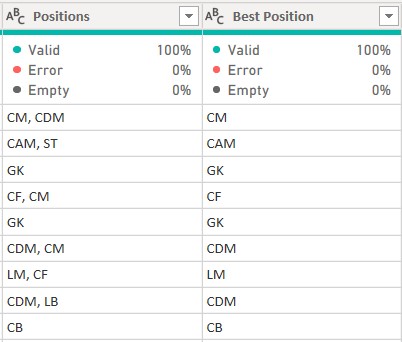
The delimiter was changed from ‘ ~ ’ to ‘ - ‘. A conditional formatting was used to separate the players who have active contract (2004-2021) from players on ‘loan’ and players who are ‘free’. I further created a custom column to calculate the contract duration of each player. The formula below was used.

=If text.contains([contract],”-“), then Number.From(Text.AfterDelimiter([contract],”-“)) – Number.From(Text.BeforeDelimiter([contract],”-“)), else “No Contract”)



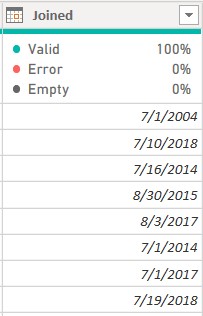
* Position:

The ‘Posotion’ column consist one or more positions the player has ever played in. The ‘position’ column was removed as there’s another column ‘Best Position’ which describe the best position for the player. This is best fit for analysis.



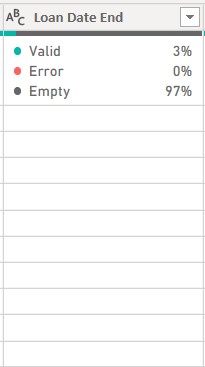
* Joined:

The joined column was filtered to see if there are any inconsistencies in the data. The data has consistent data and the correct data type (date).

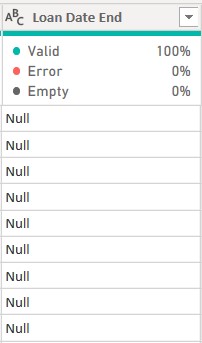


* Loan Date End:

The ‘loan date end’ column has 97% empty rows. This is because the players on loan account for only 3% of the entire dataset.



To solve the problem, I replaced the empty cells with ‘Null’.



* Height & Weight:

The column contains inconsistent data. The value of height was entered in two different format- 120cm & 5’10” (foot&inches), while the ‘weight’ column has values entered in two format- ‘kg’ and ‘ibs’.



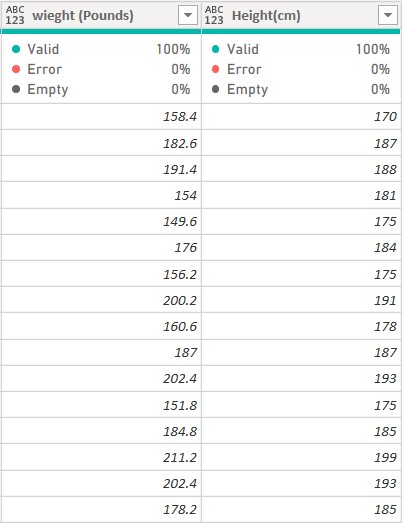
To achieve consistent data in this column, I converted the foot & inches values to cm. This was done using a formula to multiply the feet by 30.48 and the inches by 2.54 which is the standard conversion.

Let cm = if Text.Contains([Height],"cm") then Number.From( Text.BeforeDelimiter([Height],"cm")) else null, ft = Number.From ( Text.BeforeDelimiter([Height], "'")), inch = Number.From ( Text.BetweenDelimiters( [Height], "'", """")),

Result = if cm is null then (ft\*30.48) + (inch\*2.54) else cm in Result

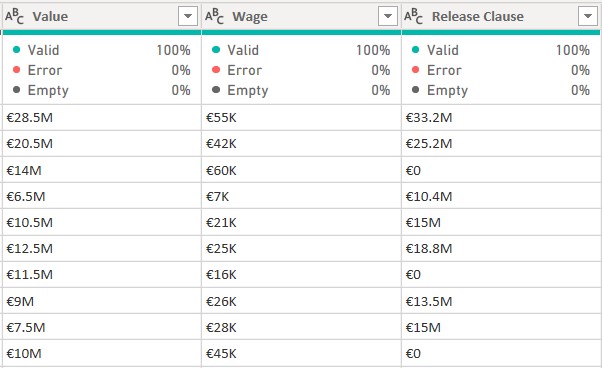
While for the weight column, I converted all the values in kg to lbs using the standard conversion rate of 2.20

= If text.contains([weight],”kg“), then Number.From(Text.BeforeDelimiter([weight],”kg“)) \*2-20 else Text.BeforeDelimiter([weight],”lbs“)),



* Value, Wage & Release Clause:

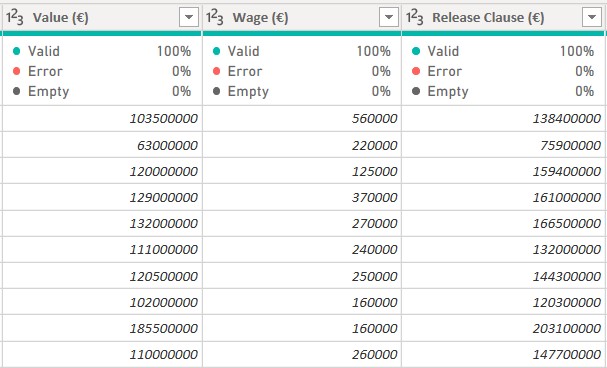
The above columns had data type issues and the values with suffixes ‘M & k’ in front. Where M is for millions, ‘K’ for thousands and ‘€’ for euro.



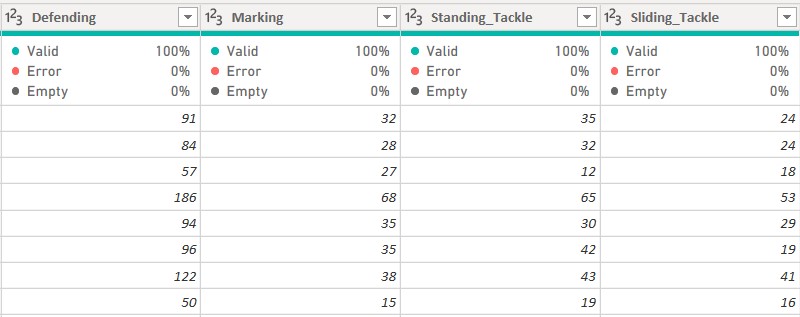
To normalize the values, the ‘€’, ‘K’ & ‘M’ was removed. A custom column was created to to multiply the values with ‘K’ by 1000 and values with ‘M’ by 1000000 using the formula below.

=IF Text.Contains([value],”M”) then Number.From(Text.beforeDelimiter([value],”M”))\*1000000 else Number.From(Text.BeforeDelimiter([value],”K”))\*1000)

The data type was converted to whole number and the column remained.

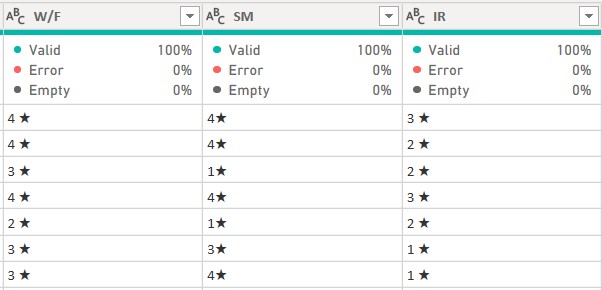


* The following columns ‘Attacking’, ‘Crossing’, ‘Finishing’, ‘Heading Accuracy’, ‘Short Passing’, ‘Volley’, ‘Skill’, ‘Dribbling’, ‘Curve’, ‘FK\_Accuracy’, ‘Long\_Passing’, ‘Ball\_Control’, ‘Movement’, ‘Acceleration’, ‘Sprint\_Speed’, ‘Agility’, ‘Reaction’, ‘Balance’, ‘Power’, ‘Short\_Power’, ‘Jumping’, ‘Stamina’, ‘Strength’, ‘Long\_Shot’, ‘Mentality’, ‘Aggression’, ‘Interceptions’, ‘Positioning’, ‘Vision’, ‘Penalties’, ‘Composure’, ‘Defending’, Marking’, ‘Standing\_Tackle’, ‘Sliding\_Tackle’, ‘GoalKeeping’, ‘GK\_Diving’, ‘GK\_Handling’, ‘GK\_Kicking’, ‘Gk\_Positioning’, ‘GK\_Reflexex’, ‘Total\_Stats’, ‘Base\_Stats’, ‘Preferred\_Foot’, ‘BOV’, ‘Attacking\_Workrate’, ‘Defending\_Workrate’, ‘PAC’, ‘SHO’, ‘DRI’, ‘DEF’, ‘PHY’, were all crosschecked and changes were made for the wrong data type. Also, some column names were corrected.

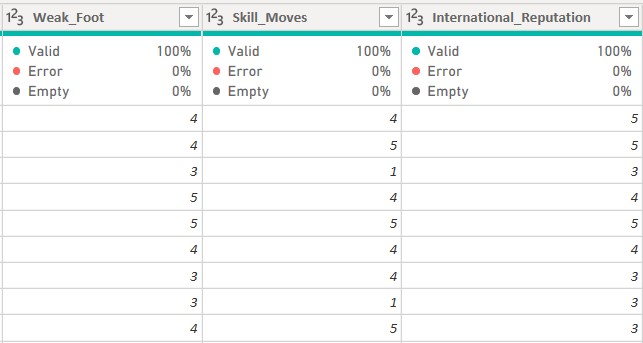


* W/F, SM & IR:

The following columns contains rating on a scale of 1-5 with a special character ‘★’ in front of each value. The columns had a wrong data type.

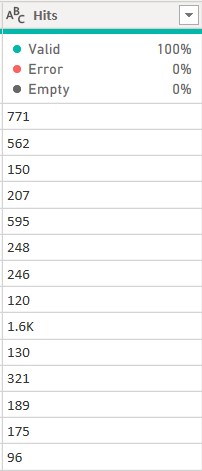


The special character was replaced with space and the column names was written in full with the data type changed to whole number.

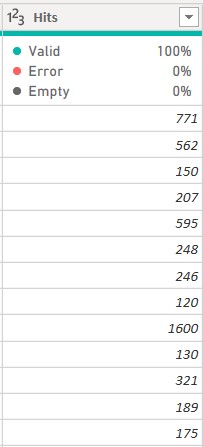


* Hits:

The column had a text data type and contained numbers in tens, hundreds, and thousands but the thousands were written in short form where 1600 is written as 1.6k.



A custom column was created to multiply the values with ‘K’ by 1000 using the formula =IF Text.Contains(Hits),”K”) then Number.From(Text.BeforeDelimiter([Hits],”K”))\*1000 else Text.BeforeDelimiter([Hits],”K”))



* Conclusion:

After a thorough checking and cross-checking for errors, missing values, empty cells, outliers, and wrong data type. I can say the messy dataset is now clean and consistent which can now be used for analysis.

I’m glad I participated in this challenge to horn and sharpen skill. Quite a lot of learning I did while taking up this task. What initially looked difficult and impossible has been done. Good to say I’m proud of myself.

Thank you for reading through. I hope you find this worth reading and helpful. You need my support on working on this dataset, you can reach me on twitter @[pinnacle\_paul](https://twitter.com/pinnacle_paul) and LinkedIn @[PaulAderounmu](https://www.linkedin.com/in/pauladerounmu)