Munetaka Murakami's Dominance: Exploring the NPB Career of Japan's Young Slugger

During the NPB 2022 season, a young japanese infielder broke a lot of records at just 22 years old. His name is Munetaka Murakami, the Tokyo Yakult Swallows' third base.

In 2022, he scored 56 homeruns, a record for a Japanese baseball player. Even Sadaharu Oh, the best NPB player of all time, had never scored so many homeruns. During this season, his team broke many offensive records and reached to in Japan Series (a losing final against Orix Buffaloes).

During the 2023 World Baseball Championship, Murakami scored the final point against Mexico to gave his country victory in the semi-finals. In final against USA, his scored a wonderful Homerun and won the World Cup with the "Samurai Japan".

Thanks to this performance, many MLB recruters were impressed by his level and potential and a bright future in the big American league seems promised. But since this 2023 World Baseball Championship, the Murakami's performance has been deceiving.

In this analyse, we will try to evaluate de Murakami level and his future prospects in MLB. We will start to analyse its first 5 seasons NPB season and compare them with those of former Japanese stars who joined the MLB. We are going to look at the player's strengths and weaknesses. We will also look at his rise and fall since the start of 2023. Finally, let's find out what kind of level Murakami could aspire to in MLB.

Source

We will use three differents websites to extract NPB or MLB data. when the data is not sure or convergent between several sources, I will report it.

- NPB data from 2000 to 2022 : http://npbstats.com/eng/
- NPB data from 2020 to 2024 : https://1point02.jp
- NPB and MLB data : https://www.baseball-reference.com
- MLB data : https://www.fangraphs.com
- MLB data : https://baseballsavant.mlb.com

```
#Imports libraries for data analysis and visualization in Python

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import percentileofscore
from soccerplots.radar_chart import Radar
```

Chapter 1 : Murakami's 2022 season and his evolution since his NPB debut

Firsly, we import the Murakami's data and analyze it Warning : In 2020, fewer matches were played (around twenty) because of the covid epidemic. Last update : May 10, 2024

```
# Read the Excel file with Murakami data
murakami = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\murakami.xlsx")
# Filter Murakami's data to 2020 and 2024 NPB seasons
murakami20 = murakami.loc[(murakami["Year"] == 2020)]
murakami21 = murakami.loc[(murakami["Year"] == 2021)]
murakami22 = murakami.loc[(murakami["Year"] == 2022)]
murakami23 = murakami.loc[(murakami["Year"] == 2023)]
murakami24 = murakami.loc[(murakami["Year"] == 2024)]
# Display the DataFrame
murakami
```

```
   Year      Name  Age      Tm Pos    G   PA    H  HR  RBI  SB    BB%
K%  \
0  2018  Murakami   18  Yakult  3B    6   14    1   1    2   0  0.143
0.357
1  2019  Murakami   19  Yakult  1B  143  593  118  36   96   5  0.125
0.310
2  2020  Murakami   20  Yakult  1B  120  515  130  28   86  11  0.169
0.223
3  2021  Murakami   21  Yakult  3B  143  615  139  39  112  12  0.172
0.216
4  2022  Murakami   22  Yakult  3B  141  612  155  56  134  12  0.193
0.209
5  2023  Murakami   23  Yakult  3B  140  597  127  31   84   5  0.151
0.281
6  2024  Murakami   24  Yakult  3B   32  144   32   8   16   2  0.208
0.271

     AVG    OBP    SLG    OPS  BABIP   WAR
0  0.083  0.214  0.333  0.548  0.000  -0.1
1  0.231  0.332  0.481  0.814  0.279   1.0
2  0.307  0.427  0.585  1.012  0.362   4.7
3  0.278  0.408  0.566  0.974  0.302   6.3
4  0.318  0.458  0.710  1.168  0.327  10.2
5  0.256  0.375  0.500  0.875  0.319   3.7
6  0.283  0.438  0.513  0.951  0.364   1.1
```

In this firts analysis, we use a scatterplot to highlight Murakami's performance in OBP and OPS, two important advanced baseball statistics for offense. In the "Hue" parameter, I've decided to highlight the WAR too.

Glossary :

"OBP" stands for On-base Percentage in baseball . It is a statistic that measures the frequency at which a batter reaches base safely in relation to their number of plate appearances . On-base percentage provides a comprehensive measure of a batter's ability to avoid making outs and get on base through hits, walks, and hit by pitches . It reflects how effectively a batter contributes to their team's offensive performance by extending innings and creating scoring opportunities.

"OPS" stands for On-base Plus Slugging in baseball. It is a comprehensive statistic that combines a player's on-base percentage (OBP) and slugging percentage (SLG) into a single metric. OPS provides a more complete picture of a player's offensive production by incorporating both their ability to get on base (OBP) and their ability to hit for power (SLG). It evaluates a player's performance in two critical aspects of hitting. OPS correlates strongly with a team's ability to score runs and win games. Players with higher OPS tend to contribute more to their team's offensive success by generating scoring opportunities and driving in runs.

"WAR" stands for Wins Above Replacement in baseball. It is a comprehensive statistic that quantifies a player's total contribution to their team's success compared to a replacement-level player. WAR combines a player's offensive, defensive, and baserunning contributions into a single value. WAR allows for easy comparison of players across different positions and skill sets. WAR correlates strongly with a team's ability to win games and contend for championships.

Rating :

- On-base Percentage in baseball in Baseball

Awful : Below .290 Poor : .300 Below Average : .310 Average : .320 Above Average : 0.340 Great : .370 Excellent : .390

Source : https://library.fangraphs.com/offense/ops/

- On-base Plus Slugging in baseball

Awful : Below .570 Poor : .600 Below Average : .670 Average : .710 Above Average : 0.800 Great : .900 Excellent : 1.000

Source : https://library.fangraphs.com/offense/ops/

- Wins Above Replacement in baseball

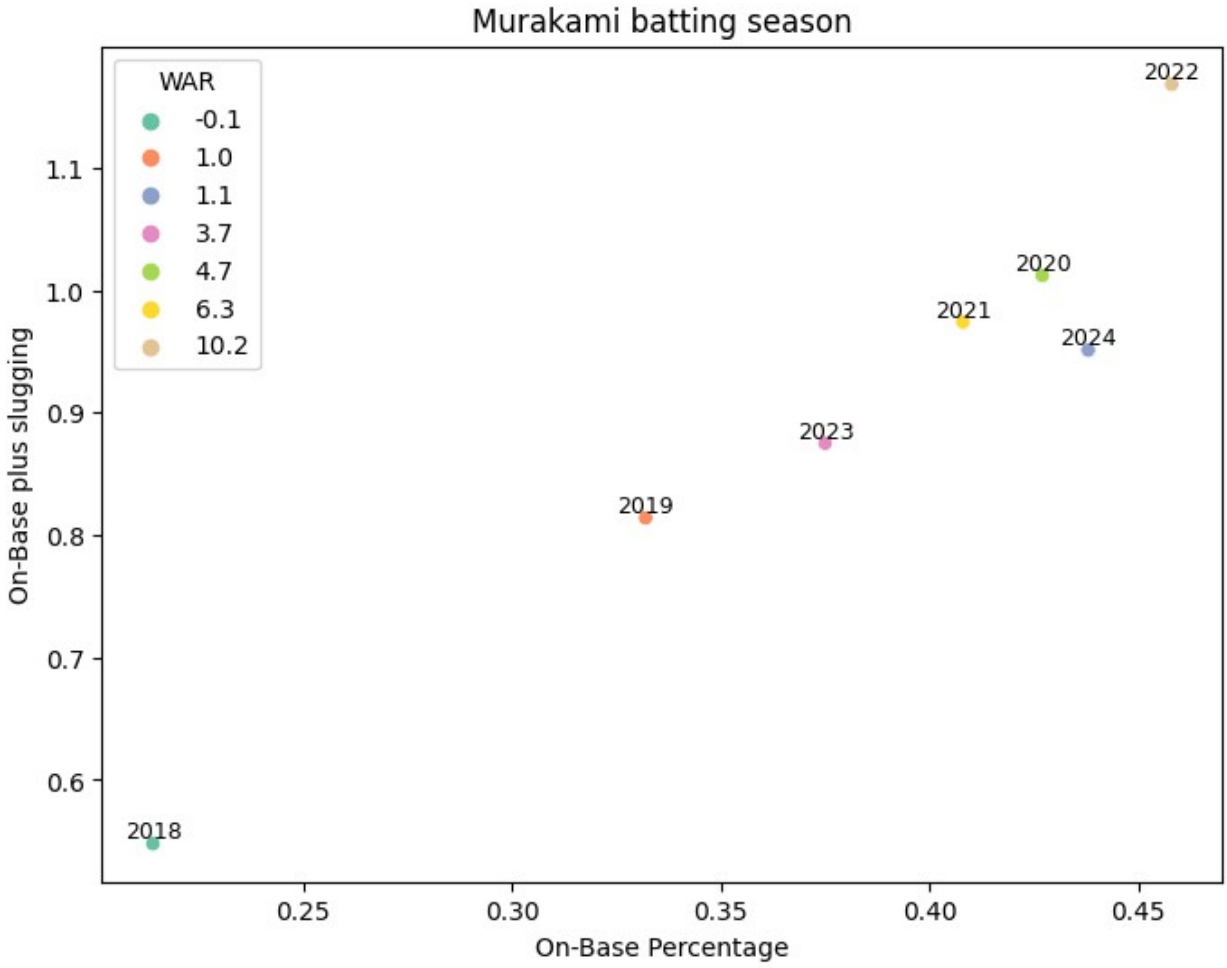
Scrub : 0-1 Role Player : 1-2 Solid Starter : 2-3 Good Player : 3-4 All-Star : 4-5 Superstar : 5-6 MVP : 6+

Source : https://library.fangraphs.com/misc/war/

Warning : This classification is approximate and is strongly influenced by MLB data, which are more numerous on the subject. In NPB, there is generally more “contact” (thus a higher “AVG” and “OPB” level) but less “power” (thus a lower “SLG” and “OPS” level). Take this classification as an arbitrary indication, which is approximate but represents the level of the athlete in this category.

```
# We have decided to create a scatterplot, with the "OBP" in x-axis
and "OPS" in y-axis
```

```
# A parameter have added to highlight the WAR value
sns.scatterplot(x= "OBP", y="OPS", data=murakami, hue = "WAR",
palette="Set2")
plt.gcf().set_size_inches(8, 6)

# Use a loop to indicate Murakami's year performance in each point
for index, row in murakami.iterrows():
    plt.text(row["OBP"], row["OPS"], row["Year"],fontsize=9,
ha='center', va='bottom')

# Creating scatterplot's labels, legend and title
plt.xlabel('On-Base Percentage')
plt.ylabel('On-Base plus slugging')
plt.title("Murakami batting season")
legend = plt.legend(loc="upper left")
legend.set_title("WAR")
plt.show()
```



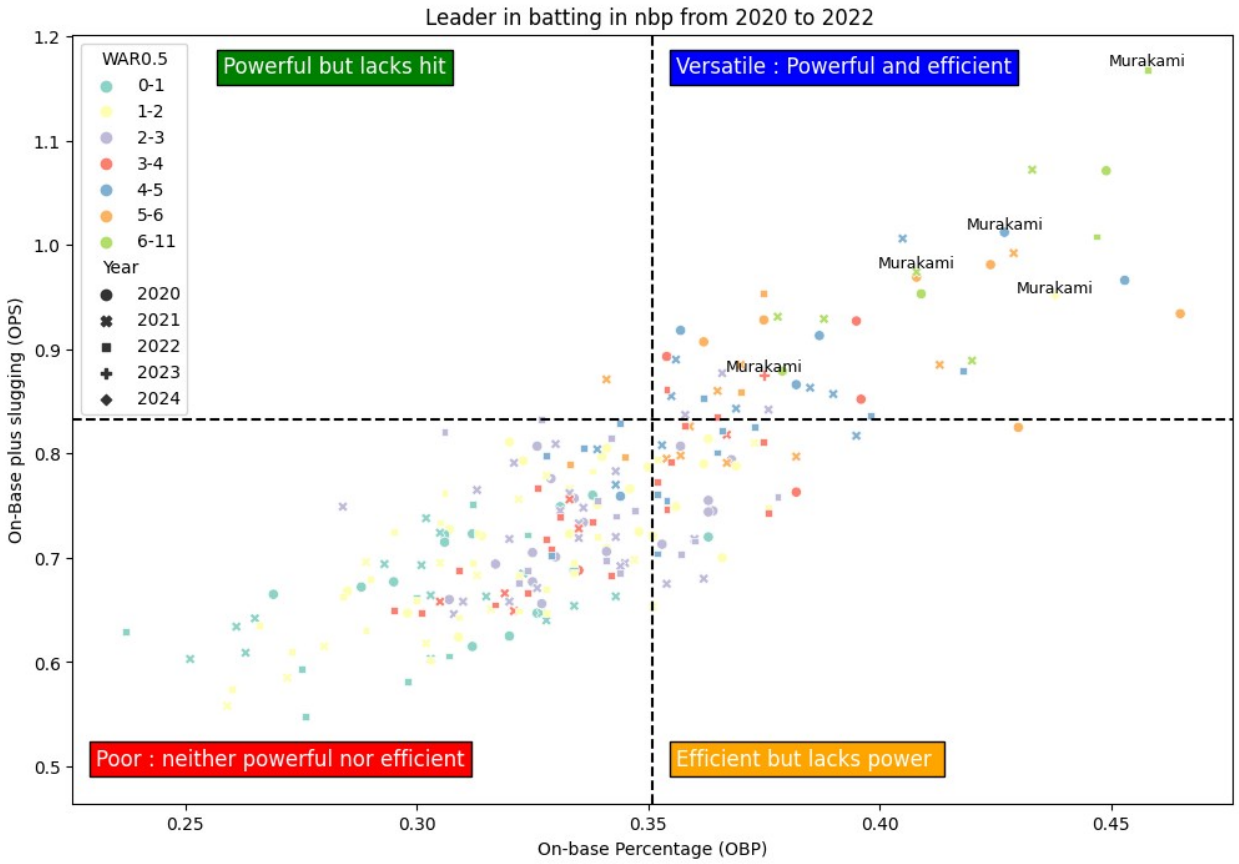
After that, We have decided to import the data of batting players from 2020 to 2022 into NPB. To compare Murakami's performances with those of his colleagues in the Japanese league.

```
# Read the Excel file
leader2022 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
NpbBatting2022.xlsx")
leader2021 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
NpbBatting2021.xlsx")
leader2020 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
NpbBatting2020.xlsx")
# Concatenate the different variables to group data from 2020 to 2022
for teams in NPB
leadernpb = pd.concat([leader2020,leader2021, leader2022, murakami23,
murakami24])
# Reset Index
leadernpb.reset_index(drop=True, inplace=True)
# We decided to aggregate the WAR value in 4 differents intervalles to
have a better vizualisation
leadernpb["WAR0.5"] = leadernpb["WAR"].astype(int)
intervalles = [0, 0.9, 1.9, 2.9, 3.9, 4.9, 5.9, 11]
labels = ['0-1','1-2', '2-3', '3-4','4-5', '5-6', '6-11']
leadernpb["WAR0.5"] = pd.cut(leadernpb["WAR0.5"], bins=intervalles,
labels=labels, include_lowest=True)
```

Warning : We can see in this dataframe that the data in the “Tm” column (representing team names) is an approximate abbreviation. We will modify this problem later to make the dataframe more comprehensible.

```
#Filter Murakami's NPB 2020 to 2022 seasons
murakami20_24 = murakami.loc[(murakami["Year"]<= 2024) &
(murakami["Year"] >2019)]
#Filter Murakami's NPB 2022 and 2023 season
murakami22 = murakami.loc[(murakami["Year"] == 2022)]

# We have decided to create a scatterplot, with the "OBP" in x-axis
and "OPS" in y-axis
# A parameter have added to highlight the WAR value
# We have decided to add a new parameter to separate the performance
of each individual over the 3 different seasons
l = sns.scatterplot(x="OBP", y="OPS", data = leadernpb, hue="WAR0.5",
style = "Year", palette="Set3")
plt.gcf().set_size_inches(12, 8)

# Use a loop to indicate Murakami's year performance in each point
for index, row in murakami20_24.iterrows():
    plt.text(row["OBP"], row["OPS"], row["Name"],fontsize=9,
ha='center', va='bottom')

# We have created a grid to better visualize the level of each athlete

plt.axhline(y=l.axes.get_ylim()[0] + (l.axes.get_ylim()[1] -
l.axes.get_ylim()[0]) / 2, color='k', linestyle='--')
```

```
plt.axvline(x=l.axes.get_xlim()[0] + (l.axes.get_xlim()[1] -
l.axes.get_xlim()[0]) / 2, color='k', linestyle='--')

# We have added differents commentary labels to better understand the
level of each baseball player
plt.text(l.axes.get_xlim()[0] + 0.13 * (l.axes.get_xlim()[1] -
l.axes.get_xlim()[0]),
         l.axes.get_ylim()[1] - 0.05 * (l.axes.get_ylim()[1] -
l.axes.get_ylim()[0]),
         "Powerful but lacks hit", fontsize=12, color='white',
bbox=dict(facecolor='green', edgecolor='black', boxstyle='square'))
plt.text(l.axes.get_xlim()[0] + 0.02 * (l.axes.get_xlim()[1] -
l.axes.get_xlim()[0]),
         l.axes.get_ylim()[0] + 0.05 * (l.axes.get_ylim()[1] -
l.axes.get_ylim()[0]),
         "Poor : neither powerful nor efficient", fontsize=12,
color='white', bbox=dict(facecolor='red', edgecolor='black',
boxstyle='square') )
plt.text(l.axes.get_xlim()[1] - 0.48 * (l.axes.get_xlim()[1] -
l.axes.get_xlim()[0]),
         l.axes.get_ylim()[1] - 0.05 * (l.axes.get_ylim()[1] -
l.axes.get_ylim()[0]),
         "Versatile : Powerful and efficient", fontsize=12,
color='white', bbox=dict(facecolor='blue', edgecolor='black',
boxstyle='square'))
plt.text(l.axes.get_xlim()[1] - 0.48 * (l.axes.get_xlim()[1] -
l.axes.get_xlim()[0]),
         l.axes.get_ylim()[0] + 0.05 * (l.axes.get_ylim()[1] -
l.axes.get_ylim()[0]),
         "Efficient but lacks power ", fontsize=12, color='white',
bbox=dict(facecolor='orange', edgecolor='black', boxstyle='square') )

# Creating scatterplot's labels, legend and title
plt.xlabel('On-base Percentage (OBP)')
plt.ylabel('On-Base plus slugging (OPS)')
plt.title("Leader in batting in nbp from 2020 to 2022 ")
plt.show()
```

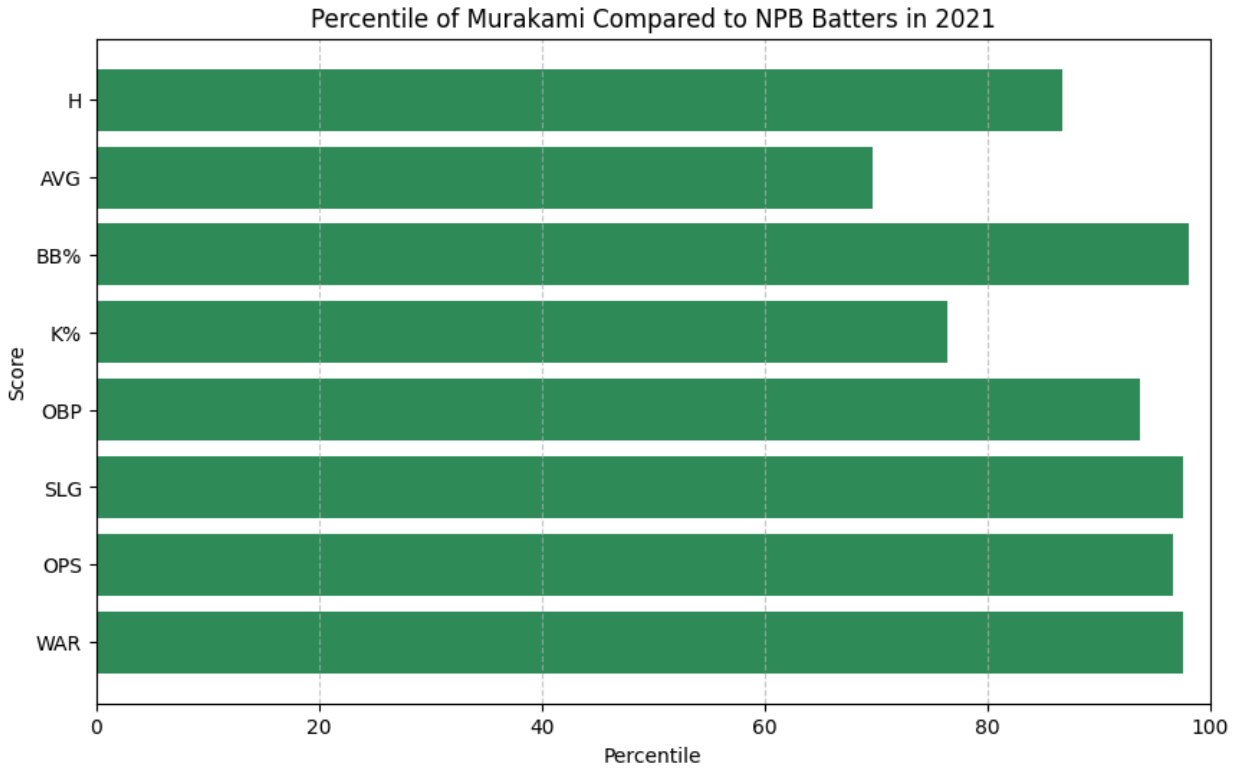


Faced with the league's best player, Murakami seems to have lost a great deal of his ability to score regular hits. Let's take a look at where Murakami stands against the competition in 2022 and 2023 in 4 categories: AVG, OPS, H, BB%. We have added the hit category to observe his gross performance in this exercise and also his percentage of base-on-balls. This last statistic is for observing whether pitchers were more or less accurate against him in 2022 and 2023.

```
# we store the values we want to analyze in a variable
scores = ['WAR', 'OPS', 'SLG', 'OBP', 'K%','BB%', 'AVG', 'H']
percentiles = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score in scores:
    murakami_score = murakami21[score].values[0]
    percentile = percentileofscore(leadernpb[score], murakami_score)
    percentiles.append(percentile)
    print(f"In 2021, Murakami was at the {percentile:.2f}th percentile
of NPB batters from 2020 to 2022 in terms of {score} score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores, percentiles, color='#2E8B57')
plt.xlabel('Percentile')
```

```
plt.ylabel('Score')
plt.title('Percentile of Murakami Compared to NPB Batters in 2021')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

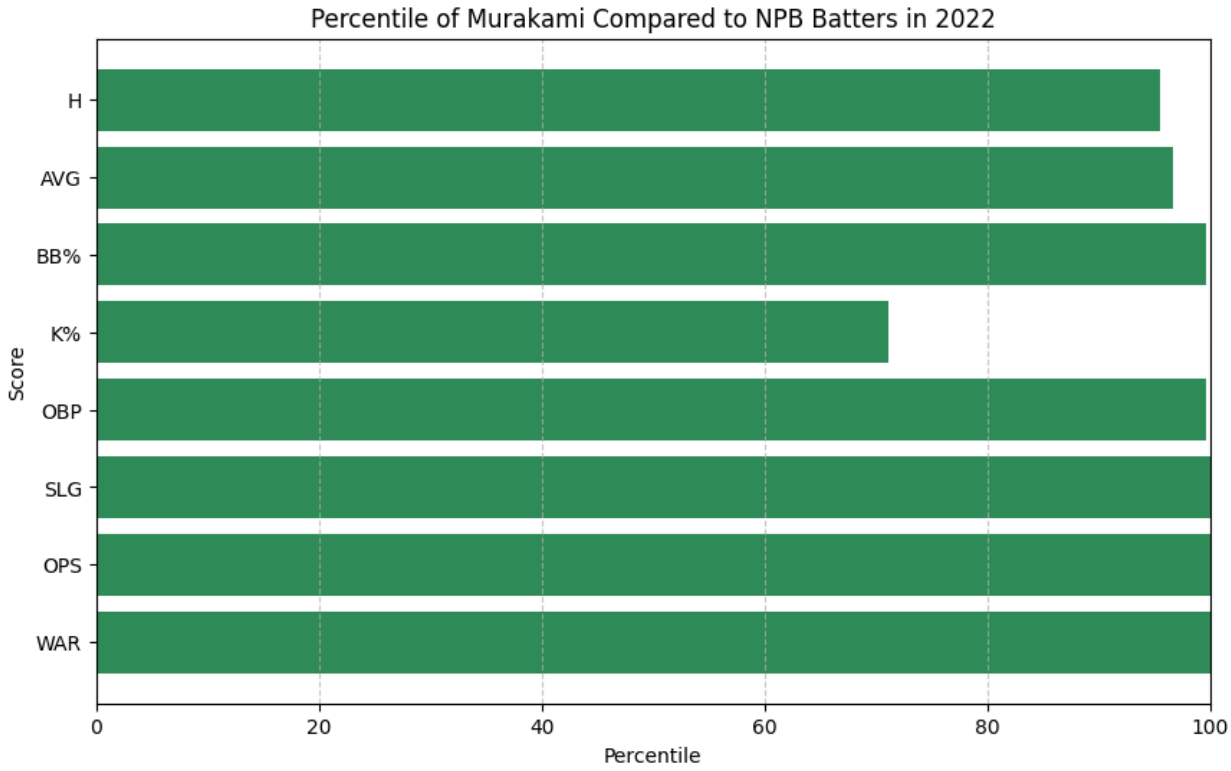
```
In 2021, Murakami was at the 97.56th percentile of NPB batters from
2020 to 2022 in terms of WAR score.
In 2021, Murakami was at the 96.75th percentile of NPB batters from
2020 to 2022 in terms of OPS score.
In 2021, Murakami was at the 97.56th percentile of NPB batters from
2020 to 2022 in terms of SLG score.
In 2021, Murakami was at the 93.70th percentile of NPB batters from
2020 to 2022 in terms of OBP score.
In 2021, Murakami was at the 76.42th percentile of NPB batters from
2020 to 2022 in terms of K% score.
In 2021, Murakami was at the 98.17th percentile of NPB batters from
2020 to 2022 in terms of BB% score.
In 2021, Murakami was at the 69.72th percentile of NPB batters from
2020 to 2022 in terms of AVG score.
In 2021, Murakami was at the 86.79th percentile of NPB batters from
2020 to 2022 in terms of H score.
```




```
# we store the values we want to analyze in a variable
scores = ['WAR', 'OPS', 'SLG', 'OBP', 'K%','BB%', 'AVG', 'H']
```

```
percentiles = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score in scores:
    murakami_score = murakami22[score].values[0]
    percentile = percentileofscore(leadernpb[score], murakami_score)
    percentiles.append(percentile)
    print(f"In 2022, Murakami was at the {percentile:.2f}th percentile
of NPB batters from 2020 to 2022 in terms of {score} score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores, percentiles, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Murakami Compared to NPB Batters in 2022')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

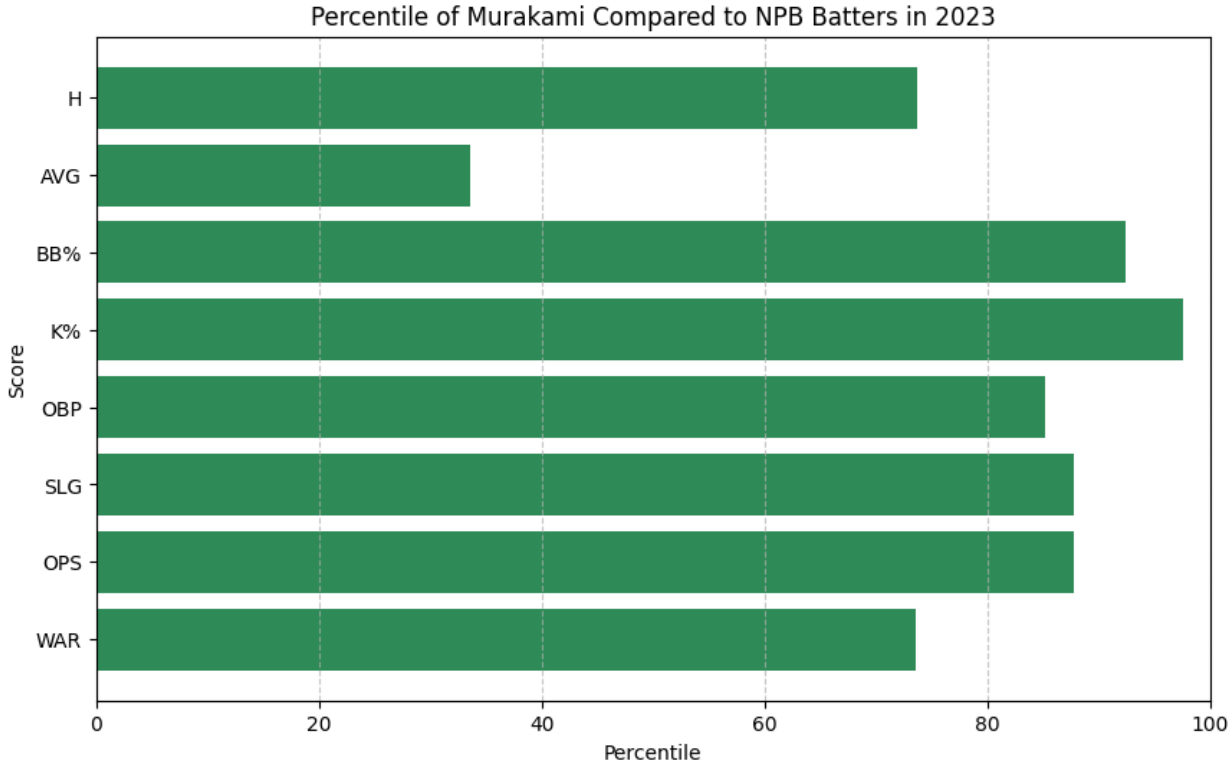
```
In 2022, Murakami was at the 100.00th percentile of NPB batters from
2020 to 2022 in terms of WAR score.
In 2022, Murakami was at the 100.00th percentile of NPB batters from
2020 to 2022 in terms of OPS score.
In 2022, Murakami was at the 100.00th percentile of NPB batters from
2020 to 2022 in terms of SLG score.
In 2022, Murakami was at the 99.59th percentile of NPB batters from
2020 to 2022 in terms of OBP score.
In 2022, Murakami was at the 71.14th percentile of NPB batters from
2020 to 2022 in terms of K% score.
In 2022, Murakami was at the 99.59th percentile of NPB batters from
2020 to 2022 in terms of BB% score.
In 2022, Murakami was at the 96.75th percentile of NPB batters from
2020 to 2022 in terms of AVG score.
In 2022, Murakami was at the 95.53th percentile of NPB batters from
2020 to 2022 in terms of H score.
```




```
# we store the values we want to analyze in a variable
scores = ['WAR', 'OPS', 'SLG', 'OBP', 'K%','BB%', 'AVG', 'H']
percentiles = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score in scores:
    murakami_score = murakami23[score].values[0]
    percentile = percentileofscore(leadernpb[score], murakami_score)
    percentiles.append(percentile)
    print(f"In 2023, Murakami was at the {percentile:.2f}th percentile
of NPB batters from 2020 to 2022 in terms of {score} score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores, percentiles, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Murakami Compared to NPB Batters in 2023')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

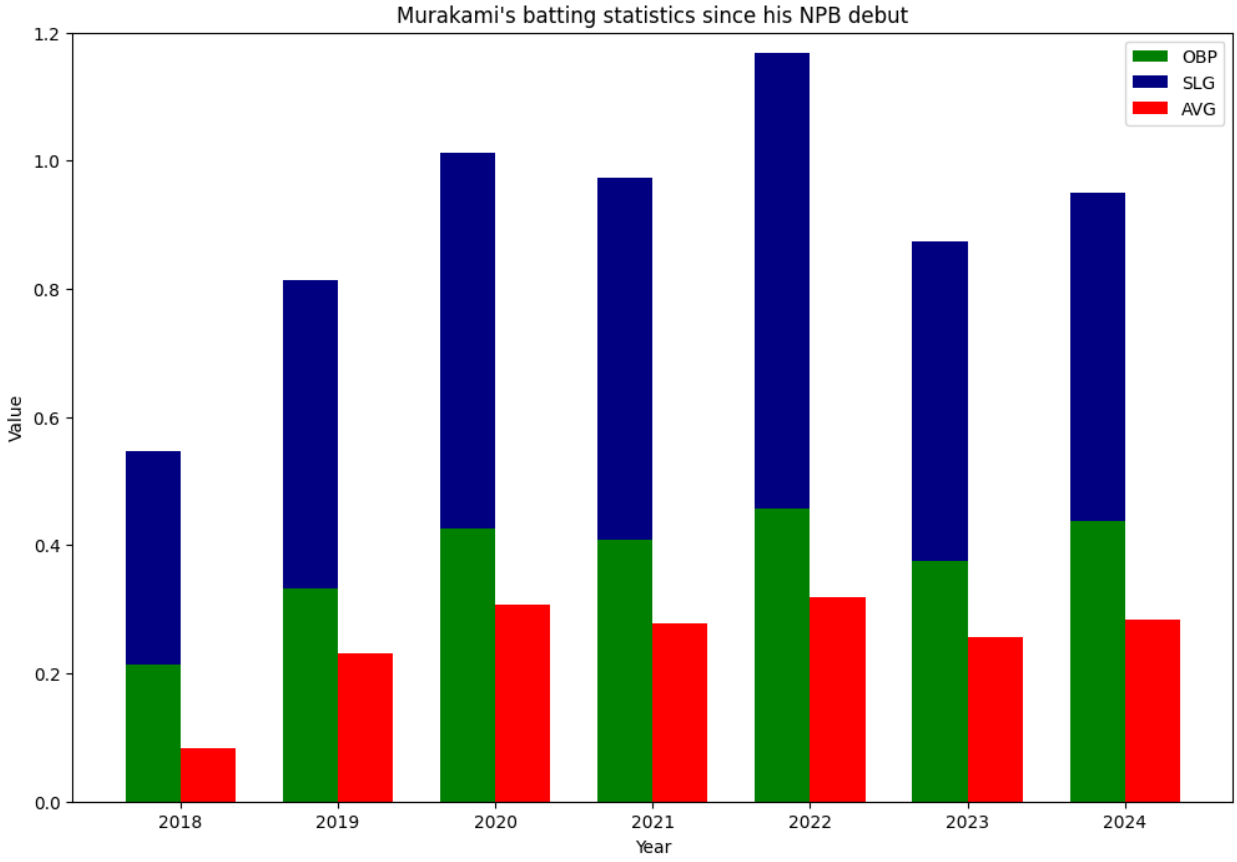
```
In 2023, Murakami was at the 73.58th percentile of NPB batters from
2020 to 2022 in terms of WAR score.
In 2023, Murakami was at the 87.80th percentile of NPB batters from
```

```
2020 to 2022 in terms of OPS score.
In 2023, Murakami was at the 87.80th percentile of NPB batters from
2020 to 2022 in terms of SLG score.
In 2023, Murakami was at the 85.16th percentile of NPB batters from
2020 to 2022 in terms of OBP score.
In 2023, Murakami was at the 97.56th percentile of NPB batters from
2020 to 2022 in terms of K% score.
In 2023, Murakami was at the 92.48th percentile of NPB batters from
2020 to 2022 in terms of BB% score.
In 2023, Murakami was at the 33.54th percentile of NPB batters from
2020 to 2022 in terms of AVG score.
In 2023, Murakami was at the 73.78th percentile of NPB batters from
2020 to 2022 in terms of H score.
```




```python
# We have decided to create a barh plot to view evolution of
Murakami's batting stats.
# The aim is to better visualize the evolution of the ability to
generate hits (“AVG” and “OBP”) and powerful swings (“SLG”) over the
years
bar_width = 0.35
x = np.arange(len(murakami['Year']))
fig, bx = plt.subplots(figsize=(12, 8))
#Here, we're going to superimpose the OBP and SLG statistics, because
the sum of the two gives the OPS, another important statistic.
obp_bar = bx.bar(x - bar_width/2, murakami['OBP'], bar_width,
```

```
label='OBP', color='green')
slg_bar = bx.bar(x - bar_width/2, murakami['SLG'], bar_width,
label='SLG', color='navy', bottom=murakami['OBP'])
avg_bar = bx.bar(x + bar_width/2, murakami['AVG'], bar_width,
label='AVG', color='red')

bx.set_ylim(0, 1.2)
bx.set_xlabel('Year')
bx.set_ylabel('Value')
bx.set_title("Murakami's batting statistics since his NPB debut")
bx.set_xticks(x)
bx.set_xticklabels(murakami['Year'])
bx.legend()
plt.show()
```



Finally, we've read about Murakami's possible superformances over the course of a season. Indeed, in sport, some athletes superform during certain games or seasons, due to mistakes by the opposing team or simply luck. In soccer and ice hockey, this is well illustrated by expected goals.

In baseball, we don't have any similar statistics, but there is one that is often used to evaluate elements outside the batting: the BABIP.

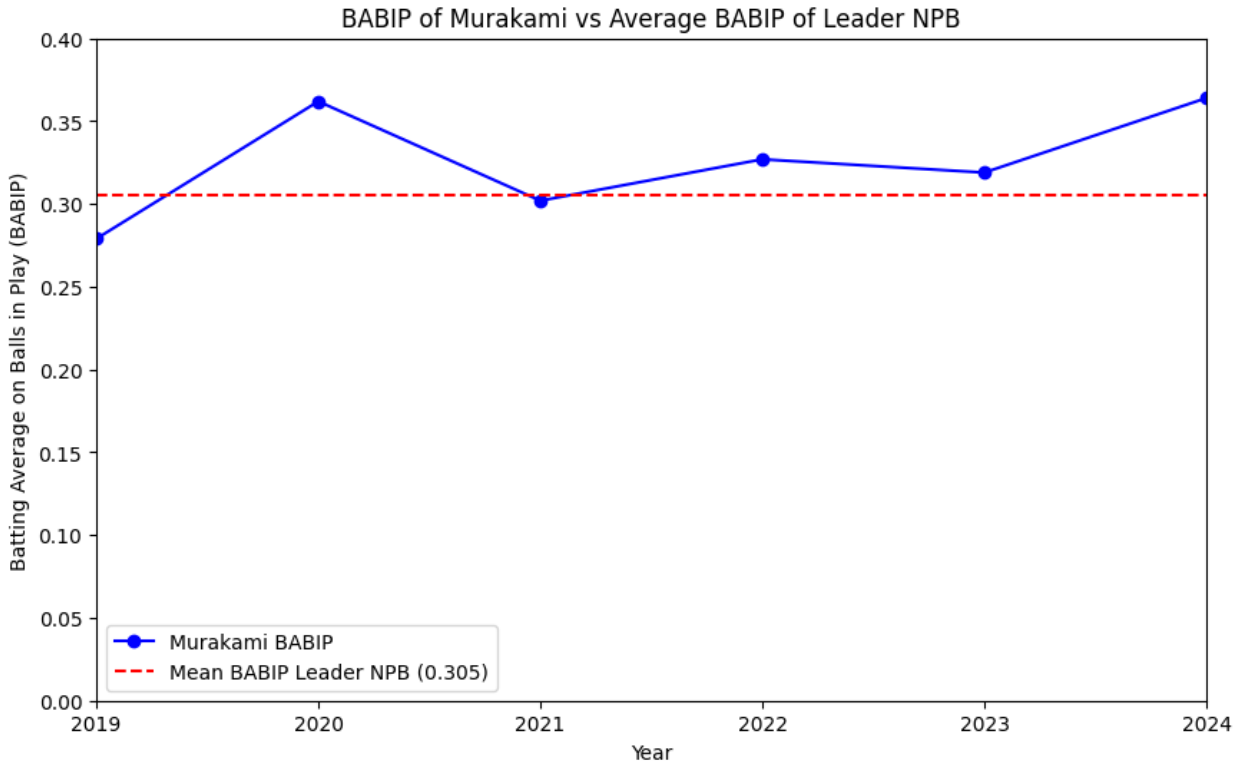
BABIP stands for Batting Average on Balls In Play in baseball. It is a statistic that measures the frequency at which a batter gets a hit on balls that are put into play, excluding home runs. BABIP provides a specific measure of a batter's ability to achieve hits on non-home run balls that are fielded. It reflects how effectively a batter manages to get hits that are not caught or turned into outs by the defense, giving insight into both the batter's skill and the defense's effectiveness.

The average score in MLB is 0.300, in NPB too (see the red horizontal line on the plot below). . From a score of 0.380, the red signals light up, as this may indicate that the opposing defences are underperforming or that the batsman has been very lucky.

Warning : This is especially true for slower players like Murakami. Skilful, fast players capable of many hits, such as Ichiro Suzuki, often score high for these reasons. Several hypotheses can be drawn from the statistics, depending on the context.

Source : https://library.fangraphs.com/offense/babip/

```
# Calculate the average of the “BABIP” column of the “LeaderNPB” df
mean_babip_leadernpb = leadernpb["BABIP"].mean()

# Plot Creation
plt.figure(figsize=(10, 6))
plt.plot(murakami["Year"], murakami["BABIP"], marker='o',
label='Murakami BABIP', color='blue')

# Add leadernpb's BABIP average line
plt.axhline(y=mean_babip_leadernpb, color='red', linestyle='--',
label=f'Mean BABIP Leader NPB ({mean_babip_leadernpb:.3f})')

# Creating plot's labels, legend and title
plt.ylim(0, 0.4)
plt.xlim(2019, 2024)
plt.title('BABIP of Murakami vs Average BABIP of Leader NPB')
plt.xlabel('Year')
plt.ylabel('Batting Average on Balls in Play (BABIP)')
plt.legend()
plt.show()
```



Looking at this plot, we can't see any variation in the level of opposing defences or luck from one year to the next. The BABIP score remains stable, with slight variations, which are quite normal.

Conclusion :

Manetaka Murakami is the more powerful japanese player in NPB, by far. Besides to its imposing size by Japanese standards (1m85 and 97 kg), he outperforms its compatriots in striking power. In addition to his homerun record, he outperforms in SLG and OPS's statistics, showing an ability to win multiple bases.

Even during his difficult 2023 season, he remained at the top of the league in terms of OPS and HR. His main concern during the 2023 season was his ability to score hits, which led to a drop in his “AVG” score. He found himself in the bottom tier of the league, which is rather worrying for his level. Warning, this could simply be the result of a difficult moment, given that his average went back up at the start of his 2024 season.

In a future chapter, we'll look back at this 2023 season in detail, to try and understand his sudden lack of ability to hit the ball.

Chapter 2 : Murakami's plate discipline

In Chapter 1, we noted that Murakami had achieved a historic season with the bat in 2022. Since then, however, he has come back to very good standards, but below his previous seasons. Soon, we could see that the problem might be his declining ability to make regular contact with the bat.

In fact, its “SLG” rate has remained relatively constant and high compared with the average, which is not the case for its number of hits, its average in “avg” or in “OBP”. In 2021, he was among the top 30% in his batting average (AVG), but that was already his lowest quality in attack. In 2023, he collapsed in this statistic, with a decline of almost 20% in his batting average.

In this chapter, we will focus on Murakami's ability to make contact with the ball and his batting average. Understand why he declined in 2023 and whether 2022 may not have been an exception.

To evaluate the plate discipline of Murakami, we will use 3 differents angles : his approach at the plate, vision and contact.

Firstly, we will begin with his approach at the plate. Behind this confusing term, we will try to understand how Murakami choose his swing and what's his style facing the pitcher. Is he more patient, waiting for the right ball to strike, or is he more aggressive? Does he go for the contact or the big hit?

For that, we will use two different metrics : BB% and K%

Glossary :

BB% stands for Walk Percentage in baseball. It is a statistic that measures the frequency at which a batter draws a walk, also known as a base on balls, in relation to their number of plate appearances. Walks are valuable for a team because they put a runner on base without the need for a hit. A high BB% indicates that the batter has good patience and discipline at the plate, as they are able to lay off pitches outside the strike zone and draw walks to get on base.

This statistic reflects a batter's ability to work the count, recognize pitches, and contribute to their team's offensive performance by extending innings and creating scoring opportunities without the need for a hit

K% stands for Strikeout Percentage in baseball. It is a statistic that measures the frequency at which a batter strikes out, meaning they are retired by the pitcher without putting the ball in play, in relation to their number of plate appearances. While strikeouts are not ideal for a batter, they are sometimes accepted as part of a hitter's profile if they come with other valuable skills, such as power hitting or a high walk rate. However, a high K% indicates that the batter struggles to make contact with the ball, either due to difficulty in making solid contact or swinging and missing at pitches.

This statistic reflects a batter's ability to make contact with the ball and avoid striking out, which can affect their ability to contribute to their team's offensive performance by putting the ball in play and advancing baserunners.

Rating :

- Walk Percentage in baseball (BB%)

Awful : Below .040 Poor : .055 Below Average : 0.07 Average : .080 Above Average : 0.100 Great : .125 Excellent : .150

Source : https://library.fangraphs.com/offense/rate-stats/

- Strikeout Percentage (K%)

Awful : Above .275 Poor : .250 Below Average : .222 Average : .200 Above Average : .160 Great : .125 Excellent : .100

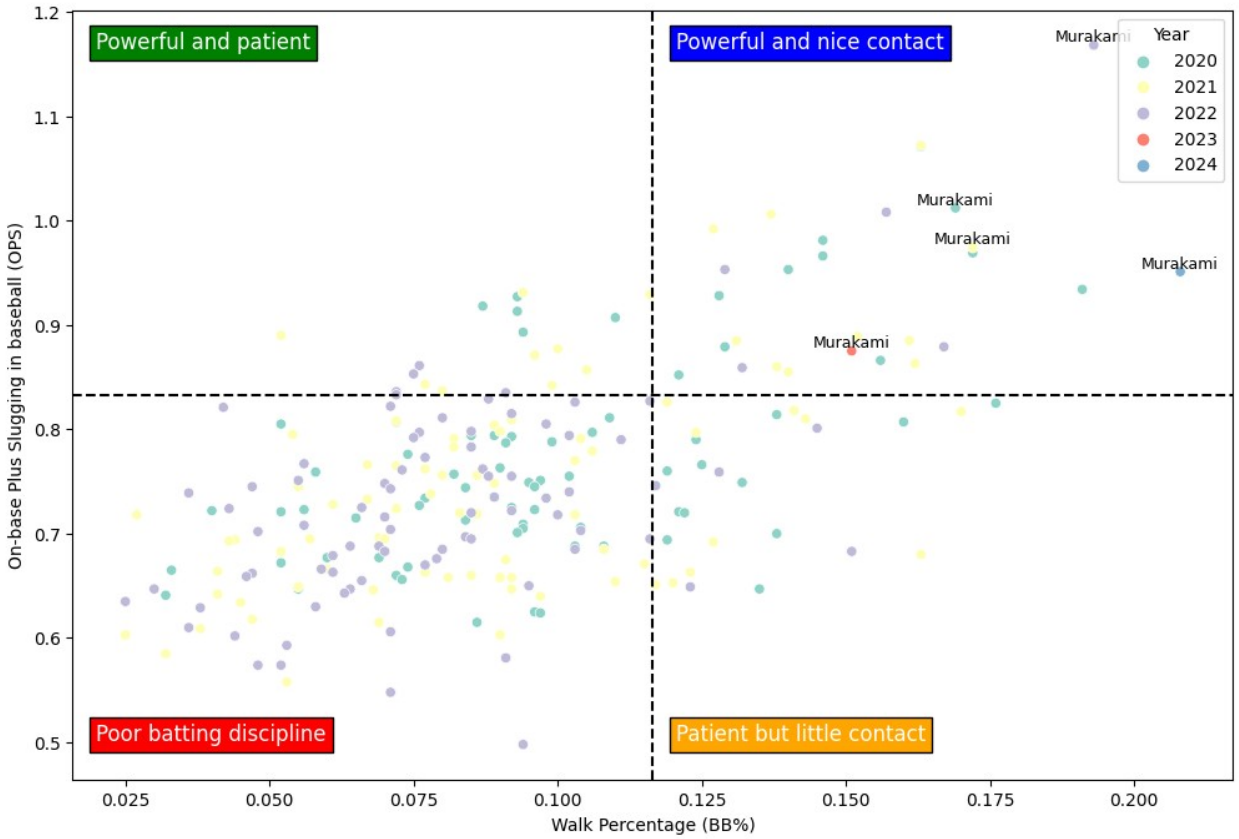
Source : https://library.fangraphs.com/offense/rate-stats/

Warning : This classification is approximate and is strongly influenced by MLB data, which are more numerous on the subject. In NPB, there is generally more “contact” (thus a higher “AVG” and “OPB” level) but less “power” (thus a lower “SLG” and “OPS” level). Take this classification as an arbitrary indication, which is approximate but represents the level of the athlete in this category.

```
# Print the data we will process below
print(murakami.loc[:, ["Name", "Age", "Year", "OPS", "BB%", "K%"]])

# We have decided to create a scatterplot, with the "BB%" in x-axis
and "K%" in y-axis
ap = sns.scatterplot(x="BB%", y="OPS", data = leadernpb, hue="Year",
palette="Set3")
plt.gcf().set_size_inches(12, 8)

# Use a loop to indicate Murakami's year performance in each point
for index, row in murakami20_24.iterrows():
    plt.text(row["BB%"], row["OPS"], row["Name"],fontsize=9,
ha='center', va='bottom')

# We have created a grid to better visualize the level of each player

plt.axhline(y=ap.axes.get_ylim()[0] + (ap.axes.get_ylim()[1] -
ap.axes.get_ylim()[0]) / 2, color='k', linestyle='--')
plt.axvline(x=ap.axes.get_xlim()[0] + (ap.axes.get_xlim()[1] -
ap.axes.get_xlim()[0]) / 2, color='k', linestyle='--')

# We have added differents commentary labels to better understand the
level of each player

plt.text(ap.axes.get_xlim()[0] + 0.02 * (ap.axes.get_xlim()[1] -
ap.axes.get_xlim()[0]),
         ap.axes.get_ylim()[1] - 0.05 * (ap.axes.get_ylim()[1] -
ap.axes.get_ylim()[0]),
         "Powerful and patient", fontsize=12, color='white',
bbox=dict(facecolor='green', edgecolor='black', boxstyle='square'))
plt.text(ap.axes.get_xlim()[0] + 0.02 * (ap.axes.get_xlim()[1] -
ap.axes.get_xlim()[0]),
         ap.axes.get_ylim()[0] + 0.05 * (ap.axes.get_ylim()[1] -
ap.axes.get_ylim()[0]),
         "Poor batting discipline", fontsize=12, color='white',
bbox=dict(facecolor='red', edgecolor='black', boxstyle='square') )
plt.text(ap.axes.get_xlim()[1] - 0.48 * (ap.axes.get_xlim()[1] -
ap.axes.get_xlim()[0]),
         ap.axes.get_ylim()[1] - 0.05 * (ap.axes.get_ylim()[1] -
```

```
ap.axes.get_ylim()[0]),
         "Powerful and nice contact", fontsize=12, color='white',
bbox=dict(facecolor='blue', edgecolor='black', boxstyle='square'))
plt.text(ap.axes.get_xlim()[1] - 0.48 * (ap.axes.get_xlim()[1] -
ap.axes.get_xlim()[0]),
         ap.axes.get_ylim()[0] + 0.05 * (ap.axes.get_ylim()[1] -
ap.axes.get_ylim()[0]),
         "Patient but little contact", fontsize=12, color='white',
bbox=dict(facecolor='orange', edgecolor='black', boxstyle='square'))


# Creating scatterplot's labels, legend and title
plt.xlabel('Walk Percentage (BB%)')
plt.ylabel('On-base Plus Slugging in baseball (OPS)')
plt.title("")
plt.show()
```

	Name	Age	Year	OPS	BB%	K%
0	Murakami	18	2018	0.548	0.143	0.357
1	Murakami	19	2019	0.814	0.125	0.310
2	Murakami	20	2020	1.012	0.169	0.223
3	Murakami	21	2021	0.974	0.172	0.216
4	Murakami	22	2022	1.168	0.193	0.209
5	Murakami	23	2023	0.875	0.151	0.281
6	Murakami	24	2024	0.951	0.208	0.271



Murakami's profile in plate is quite ambivalent. He provokes a lot of BB%, which shows his ability to be patient, but also a high rate of K%, which leads to withdrawals. In both categories, he's in the lead but at the opposite extreme, positive in BB% but negative in K%.

This could show that Murakami is a feared “Power Hitter”, pushing opposing pitchers to provoke a “walk” to avoid powerful slugging or homeruns. This also shows that Murakami prefers to generate more K%, in order to maximize his chances of achieving more powerful hits or homeruns.

Within his team, he clearly plays the role of “Power Hitter”.

Below, we will refine these first two hypotheses by taking our analysis of Murakami's Plate Discipline a step further.

Plate Discipline statistics tell us how often a hitter swings and makes contact with certain kinds of pitches or how often a pitcher induces swings or contact on certain kinds of pitches. These numbers are very useful for determining the type of hitter or pitcher at which you’re looking.

In the second part of this chapter, we'll look at how Murakami decides to swing and whether he gets more balls in or out of the zone. To do this, we'll use 7 statistics that give us a more detailed insight into the player's habits at the plate.

O-Swing% (Swing Percentage on Pitches Outside the Strike Zone) : It measures the proportion of times a player swings at pitches outside the strike zone, compared to the total number of pitches outside the strike zone he receives.

Z-Swing% (Swing Percentage on Pitches Inside the Strike Zone) : It measures the proportion of times a player swings at pitches inside the strike zone, compared to the total number of pitches in the strike zone he receives.

Swing% (Overall Swing Percentage) : It represents the percentage of all pitches a player chooses to hit, whether inside or outside the strike zone.

O-Contact% (Contact Percentage on Pitches Outside the Strike Zone) : It measures the proportion of times a player makes contact with pitches outside the strike zone, compared to the total number of pitches outside the strike zone he receives.

Z-Contact% (Contact Percentage on Pitches Inside the Strike Zone) : It measures the proportion of times a player makes contact with pitches inside the strike zone, compared to the total number of pitches in the strike zone he receives.

Contact% (Overall Contact Percentage) : It represents the percentage of all pitches a player chooses to hit that make contact, whether inside or outside the strike zone.

Zone% (Zone Percentage) : It measures the proportion of throws a player receives that are thrown into the strike zone, in relation to the total number of throws he receives.

Below, you will find the averages of all these MLB statistics. Warning : This classification is approximate and is strongly influenced by MLB data, which are more numerous on the subject. In NPB, there is generally more “contact” but less “power”. Take this classification as an arbitrary indication, which is approximate but represents the level of the athlete in this category.

- Plate Discipline Average Rating in MLB :

O-Swing : 30% Z-Swing : 65% Swing : 46% O-Contact : 66% Z-Contact : 87% Contact : 80% Zone : 45%

Source : https://library.fangraphs.com/offense/plate-discipline/

To begin our analysis of Murakami's plate discipline, we'll first extract and concatenate the NPB data set from 2020 to 2023.

Next, we'll highlight Murakami's statistics in 2021, 2022 and 2023 across the entire dataframe. We'll highlight them to see in which percentile Murakami falls on the 7 key statistics we defined above.

```
# Read the Excel file with NPB team data from 2020 to 2023
Plate_Discipline2020 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
platediscipline_2020.xlsx")
Plate_Discipline2021 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
platediscipline_2021.xlsx")
Plate_Discipline2022 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
platediscipline_2022.xlsx")
Plate_Discipline2023 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
platediscipline_2023.xlsx")

# Concatenate the different variables to group data from 2020 to 2023
Plate_Discipline_Leader =
```

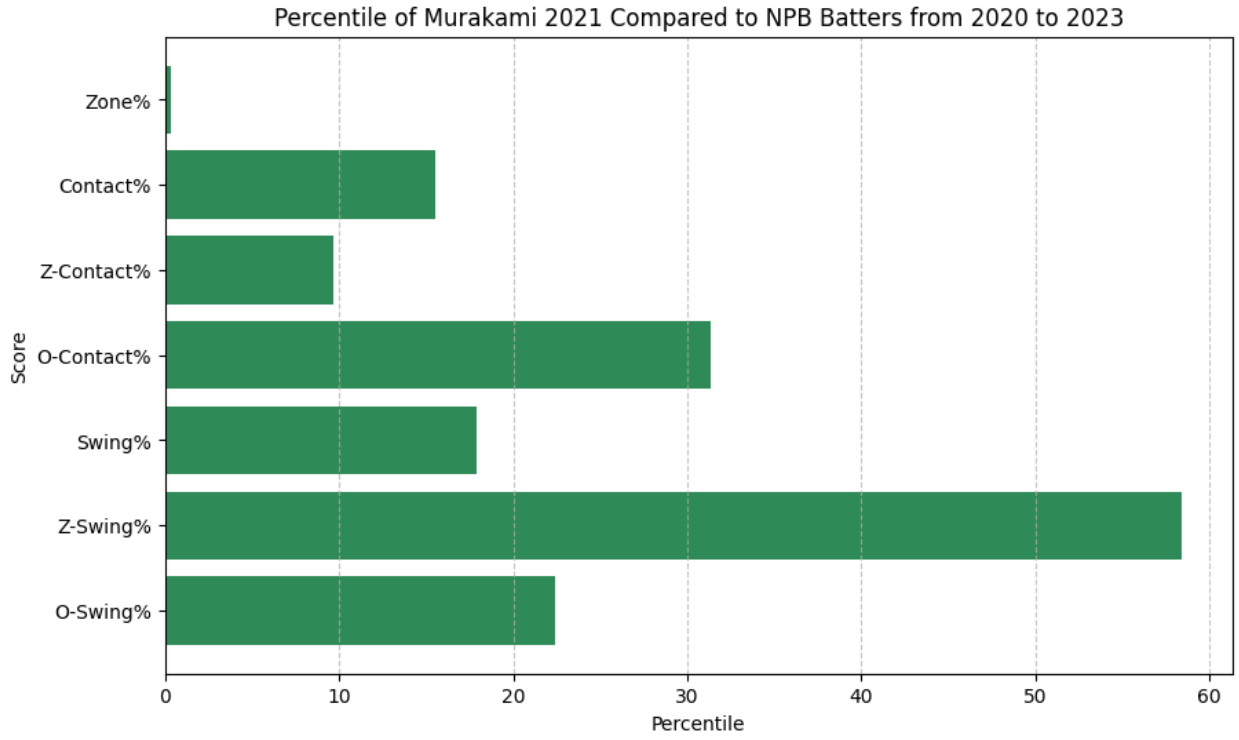
```
pd.concat([Plate_Discipline2020,Plate_Discipline2021,
Plate_Discipline2022, Plate_Discipline2023])
Plate_Discipline_Leader.reset_index(drop=True, inplace=True)

# Filter Murakami's data
Murakami_Plate_Discipline =
Plate_Discipline_Leader.loc[(Plate_Discipline_Leader["Player"] ==
"Murakami, Munetaka")]
Murakami_Plate_Discipline2023 =
Murakami_Plate_Discipline.loc[(Murakami_Plate_Discipline["Year"] ==
2023)]
Murakami_Plate_Discipline2022 =
Murakami_Plate_Discipline.loc[(Murakami_Plate_Discipline["Year"] ==
2022)]
Murakami_Plate_Discipline2021 =
Murakami_Plate_Discipline.loc[(Murakami_Plate_Discipline["Year"] ==
2021)]
Murakami_Plate_Discipline2020 =
Murakami_Plate_Discipline.loc[(Murakami_Plate_Discipline["Year"] ==
2020)]
Murakami_Plate_Discipline
```

```
                 Player  Year   PA  O-Swing%  Z-Swing%  Swing%  O-
Contact%  \
17   Murakami, Munetaka  2020  515      24.4      66.6    41.3
52.6
94   Murakami, Munetaka  2021  615      24.5      67.7    41.1
59.1
159  Murakami, Munetaka  2022  612      23.5      68.1    41.3
51.1
228  Murakami, Munetaka  2023  597      26.0      66.9    42.3
50.9

     Z-Contact%  Contact%  Zone%
17         79.9      70.2   40.0
94         80.8      72.9   38.4
159        77.1      68.3   40.0
228        74.4      65.7   39.8
```

```
# we store the values we want to analyze in a variable
scores_plate = ['O-Swing%', 'Z-Swing%', 'Swing%', "O-Contact%","Z-
Contact%","Contact%", "Zone%"]
percentiles_plate = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score_plate in scores_plate:
    Murakami_score =
Murakami_Plate_Discipline2021[score_plate].values[0]
    percentile_plate =
```

```
percentileofscore(Plate_Discipline_Leader[score_plate],
Murakami_score)
    percentiles_plate.append(percentile_plate)
    print(f"In 2021, Murakami was at the {percentile_plate:.2f}th
percentile of NPB player from 2020 to 2023 in terms of {score_plate}
score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores_plate, percentiles_plate, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Murakami 2021 Compared to NPB Batters from
2020 to 2023')
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

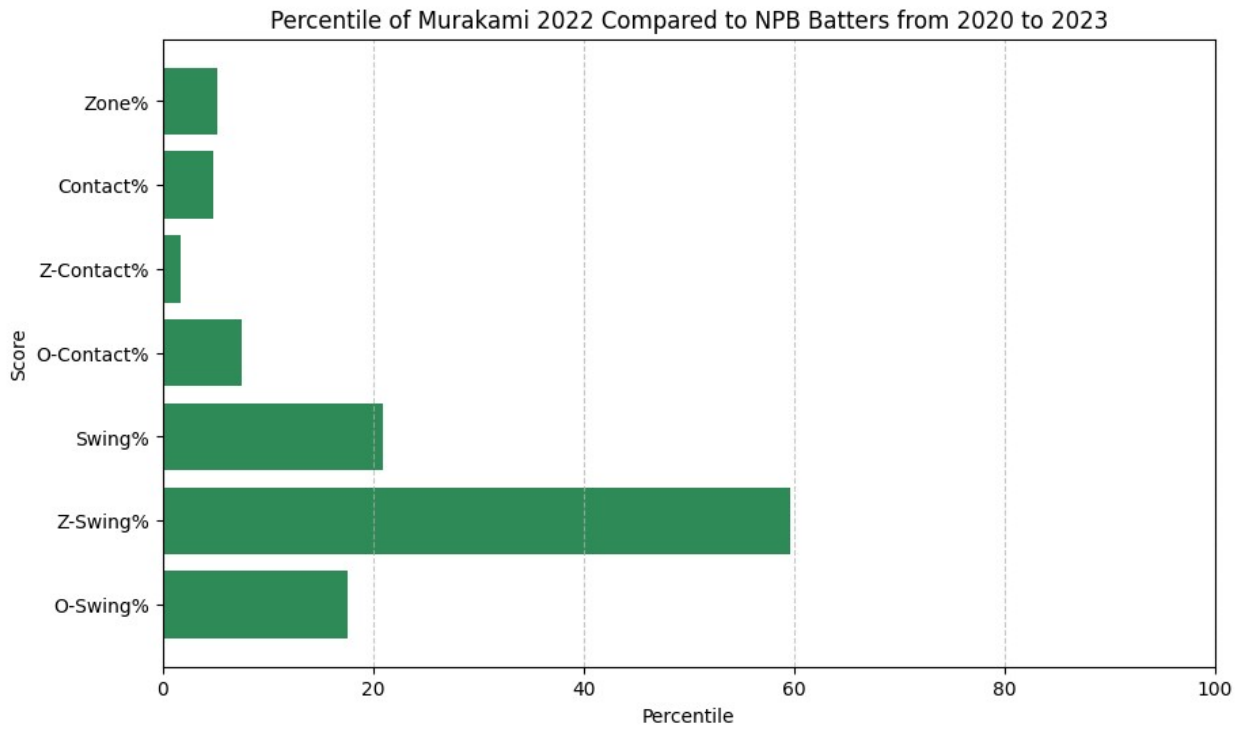
```
In 2021, Murakami was at the 22.39th percentile of NPB player from
2020 to 2023 in terms of O-Swing% score.
In 2021, Murakami was at the 58.40th percentile of NPB player from
2020 to 2023 in terms of Z-Swing% score.
In 2021, Murakami was at the 17.91th percentile of NPB player from
2020 to 2023 in terms of Swing% score.
In 2021, Murakami was at the 31.34th percentile of NPB player from
2020 to 2023 in terms of O-Contact% score.
In 2021, Murakami was at the 9.70th percentile of NPB player from 2020
to 2023 in terms of Z-Contact% score.
In 2021, Murakami was at the 15.49th percentile of NPB player from
2020 to 2023 in terms of Contact% score.
In 2021, Murakami was at the 0.37th percentile of NPB player from 2020
to 2023 in terms of Zone% score.
```




```
# we store the values we want to analyze in a variable
scores_plate = ['O-Swing%', 'Z-Swing%', 'Swing%', "O-Contact%","Z-
Contact%","Contact%", "Zone%"]
percentiles_plate = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score_plate in scores_plate:
    Murakami_score =
Murakami_Plate_Discipline2022[score_plate].values[0]
    percentile_plate =
percentileofscore(Plate_Discipline_Leader[score_plate],
Murakami_score)
    percentiles_plate.append(percentile_plate)
    print(f"In 2022, Murakami was at the {percentile_plate:.2f}th
percentile of NPB player from 2020 to 2023 in terms of {score_plate}
score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores_plate, percentiles_plate, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Murakami 2022 Compared to NPB Batters from
2020 to 2023')
plt.xlim(0, 100)
```

```
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

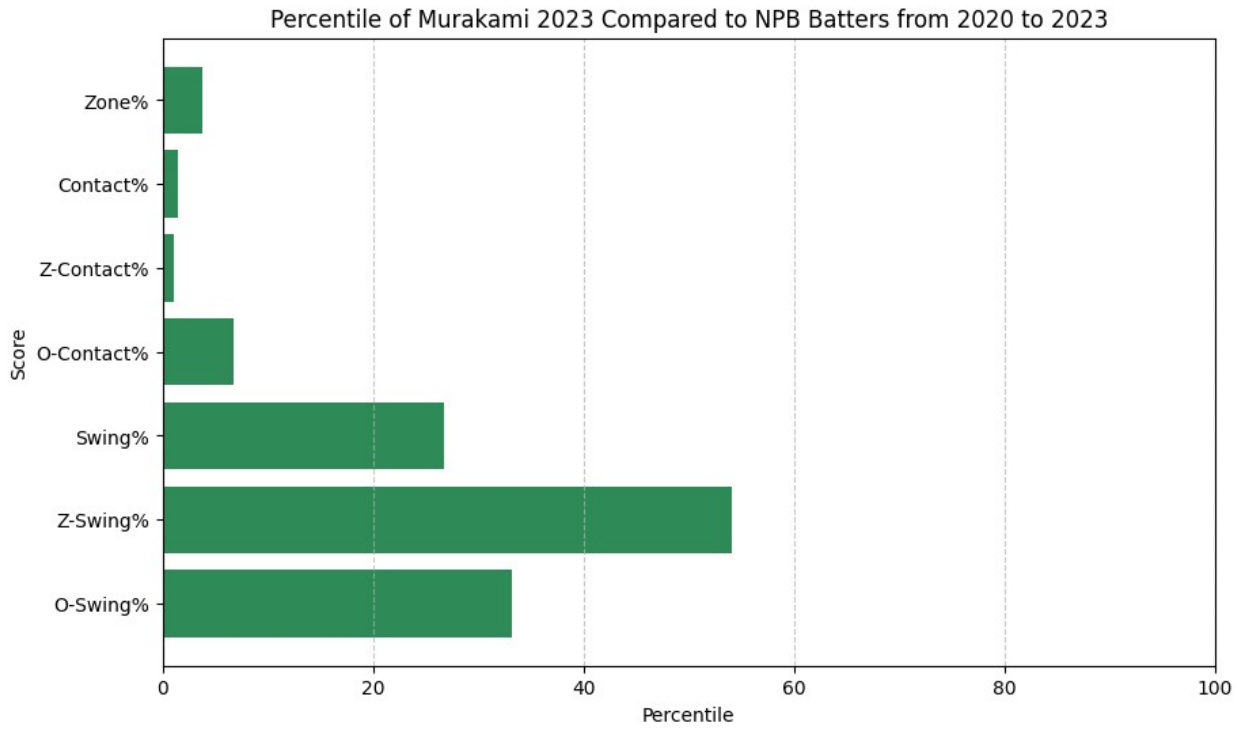
```
In 2022, Murakami was at the 17.54th percentile of NPB player from
2020 to 2023 in terms of O-Swing% score.
In 2022, Murakami was at the 59.70th percentile of NPB player from
2020 to 2023 in terms of Z-Swing% score.
In 2022, Murakami was at the 20.90th percentile of NPB player from
2020 to 2023 in terms of Swing% score.
In 2022, Murakami was at the 7.46th percentile of NPB player from 2020
to 2023 in terms of O-Contact% score.
In 2022, Murakami was at the 1.68th percentile of NPB player from 2020
to 2023 in terms of Z-Contact% score.
In 2022, Murakami was at the 4.85th percentile of NPB player from 2020
to 2023 in terms of Contact% score.
In 2022, Murakami was at the 5.22th percentile of NPB player from 2020
to 2023 in terms of Zone% score.
```




```
# we store the values we want to analyze in a variable
scores_plate = ['O-Swing%', 'Z-Swing%', 'Swing%', "O-Contact%","Z-
Contact%","Contact%", "Zone%"]
percentiles_plate = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score_plate in scores_plate:
    Murakami_score =
```

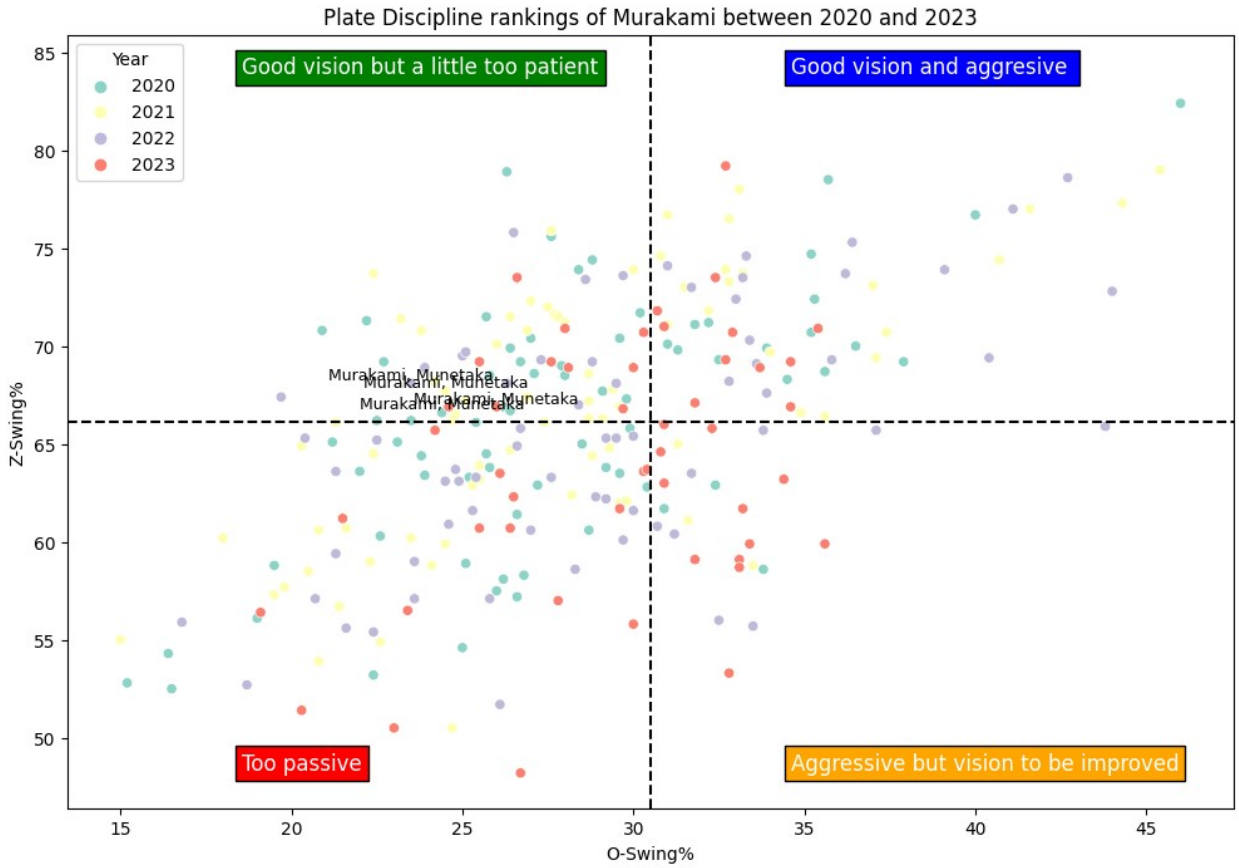
```
Murakami_Plate_Discipline2023[score_plate].values[0]
    percentile_plate =
percentileofscore(Plate_Discipline_Leader[score_plate],
Murakami_score)
    percentiles_plate.append(percentile_plate)
    print(f"In 2023, Murakami was at the {percentile_plate:.2f}th
percentile of NPB player from 2020 to 2023 in terms of {score_plate}
score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores_plate, percentiles_plate, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Murakami 2023 Compared to NPB Batters from
2020 to 2023')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

```
In 2023, Murakami was at the 33.21th percentile of NPB player from
2020 to 2023 in terms of O-Swing% score.
In 2023, Murakami was at the 54.10th percentile of NPB player from
2020 to 2023 in terms of Z-Swing% score.
In 2023, Murakami was at the 26.68th percentile of NPB player from
2020 to 2023 in terms of Swing% score.
In 2023, Murakami was at the 6.72th percentile of NPB player from 2020
to 2023 in terms of O-Contact% score.
In 2023, Murakami was at the 1.12th percentile of NPB player from 2020
to 2023 in terms of Z-Contact% score.
In 2023, Murakami was at the 1.49th percentile of NPB player from 2020
to 2023 in terms of Contact% score.
In 2023, Murakami was at the 3.73th percentile of NPB player from 2020
to 2023 in terms of Zone% score.
```




```
# We have decided to create a scatterplot, with the "O-Swing%" in x-
axis and "Z-Swing%" in y-axis
cc = sns.scatterplot(x="O-Swing%", y="Z-Swing%", data =
Plate_Discipline_Leader, hue="Year", palette="Set3")
plt.gcf().set_size_inches(12, 8)

# Use a loop to indicate Murakami's year performance in each point
for index, row in Murakami_Plate_Discipline.iterrows():
    plt.text(row["O-Swing%"], row["Z-Swing%"],
row["Player"],fontsize=9, ha='center', va='bottom')

# We have created a grid to better visualize the level of each athlete

plt.axhline(y=cc.axes.get_ylim()[0] + (cc.axes.get_ylim()[1] -
cc.axes.get_ylim()[0]) / 2, color='k', linestyle='--')
plt.axvline(x=cc.axes.get_xlim()[0] + (cc.axes.get_xlim()[1] -
cc.axes.get_xlim()[0]) / 2, color='k', linestyle='--')

# We have added differents commentary labels to better understand the
level of each team
plt.text(cc.axes.get_xlim()[0] + 0.15 * (cc.axes.get_xlim()[1] -
cc.axes.get_xlim()[0]),
         cc.axes.get_ylim()[1] - 0.05 * (cc.axes.get_ylim()[1] -
cc.axes.get_ylim()[0]),
         "Good vision but a little too patient", fontsize=12,
color='white', bbox=dict(facecolor='green', edgecolor='black',
boxstyle='square'))
```

```
plt.text(cc.axes.get_xlim()[0] + 0.15 * (cc.axes.get_xlim()[1] -
cc.axes.get_xlim()[0]),
         cc.axes.get_ylim()[0] + 0.05 * (cc.axes.get_ylim()[1] -
cc.axes.get_ylim()[0]),
         "Too passive", fontsize=12, color='white',
bbox=dict(facecolor='red', edgecolor='black', boxstyle='square') )
plt.text(cc.axes.get_xlim()[1] - 0.38 * (cc.axes.get_xlim()[1] -
cc.axes.get_xlim()[0]),
         cc.axes.get_ylim()[1] - 0.05 * (cc.axes.get_ylim()[1] -
cc.axes.get_ylim()[0]),
         "Good vision and aggresive ", fontsize=12, color='white',
bbox=dict(facecolor='blue', edgecolor='black', boxstyle='square'))
plt.text(cc.axes.get_xlim()[1] - 0.38 * (cc.axes.get_xlim()[1] -
cc.axes.get_xlim()[0]),
         cc.axes.get_ylim()[0] + 0.05 * (cc.axes.get_ylim()[1] -
cc.axes.get_ylim()[0]),
         "Aggressive but vision to be improved", fontsize=12,
color='white', bbox=dict(facecolor='orange', edgecolor='black',
boxstyle='square') )

# Creating scatterplot's labels, legend and title
plt.xlabel('O-Swing%')
plt.ylabel('Z-Swing%')
plt.title("Plate Discipline rankings of Murakami between 2020 and
2023")
plt.show()
```



Conclusion :

Murakami's batting statistics are rather ambivalent. Several analyses can be made with them, as no major trends emerge.

However, we can observe several facts.

The first is that pitchers are making an enormous number of BBs against Murakami, and have been doing so since his debut. This shows that pitchers fear the powerful hits of the Yakult Swallows player and prefer to concede a walk rather than a possible homerun. This trend is also shown by the very few pitches put in the zone, which is highlighted here by the “Zone%” statistic.

Secondly, Murakami is a patient player and would rather concede a strikeout than swing on a bad ball. Overall, he prefers to wait for the ball to arrive in his zone.

Finally, his vision at the plate is not yet perfect. He seems to lack aggressiveness, or rather he's far too patient. He's still very young, but his plate discipline remains an area for improvement.

Chapter 3 : Murakami Fielding's problems

Murakami is recognized for his wonderful performance in batting, but not for his fielding quality. Indeed, many specialists believe that he should play in first base or DH position in MLB, to hide its weaknesses.

In this chapter, we will try to highglight the fielding performances of Murakami thanks to 4 different statistics.

"FP" stands for Fielding Percentage in baseball . It is a statistic that measures the proportion of defensive chances successfully converted into outs by a fielder . Fielding percentage provides a simple measure of a fielder's reliability in making routine defensive plays . It reflects the percentage of times a fielder successfully completes defensive plays without committing errors.

"E" stands for Errors in baseball . It is a statistic that measures the number of defensive mistakes or errors committed by a fielder . Errors occur when a fielder fails to make a play that should have been made with ordinary effort, resulting in the advancement or reaching base of an opposing batter or baserunner.

"RF" stands for Range Factor in baseball . It is a statistic that measures the range or extent of a fielder's defensive coverage on the field . Range factor calculates the average number of defensive plays made by a fielder per game or inning. A higher range factor indicates a fielder who covers more ground and has a wider defensive reach on the field.

"UZR" stands for Ultimate Zone Rating in baseball . It is a statistic that measures the number of runs above or below average a fielder contributes defensively, based on their ability to make plays within their defensive zone . Ultimate zone rating takes into account factors such as a fielder's range, arm strength, and ability to convert batted balls into outs, providing a comprehensive measure of their overall defensive value. A positive UZR indicates above-average defensive performance, while a negative UZR suggests below-average performance.

Rating :

- Ultimate Zone Rating in baseball

Awful : -15 Poor : -10 Below Average : -5 Average : 0 Above Average : +5 Great : +10 Excellent : +15

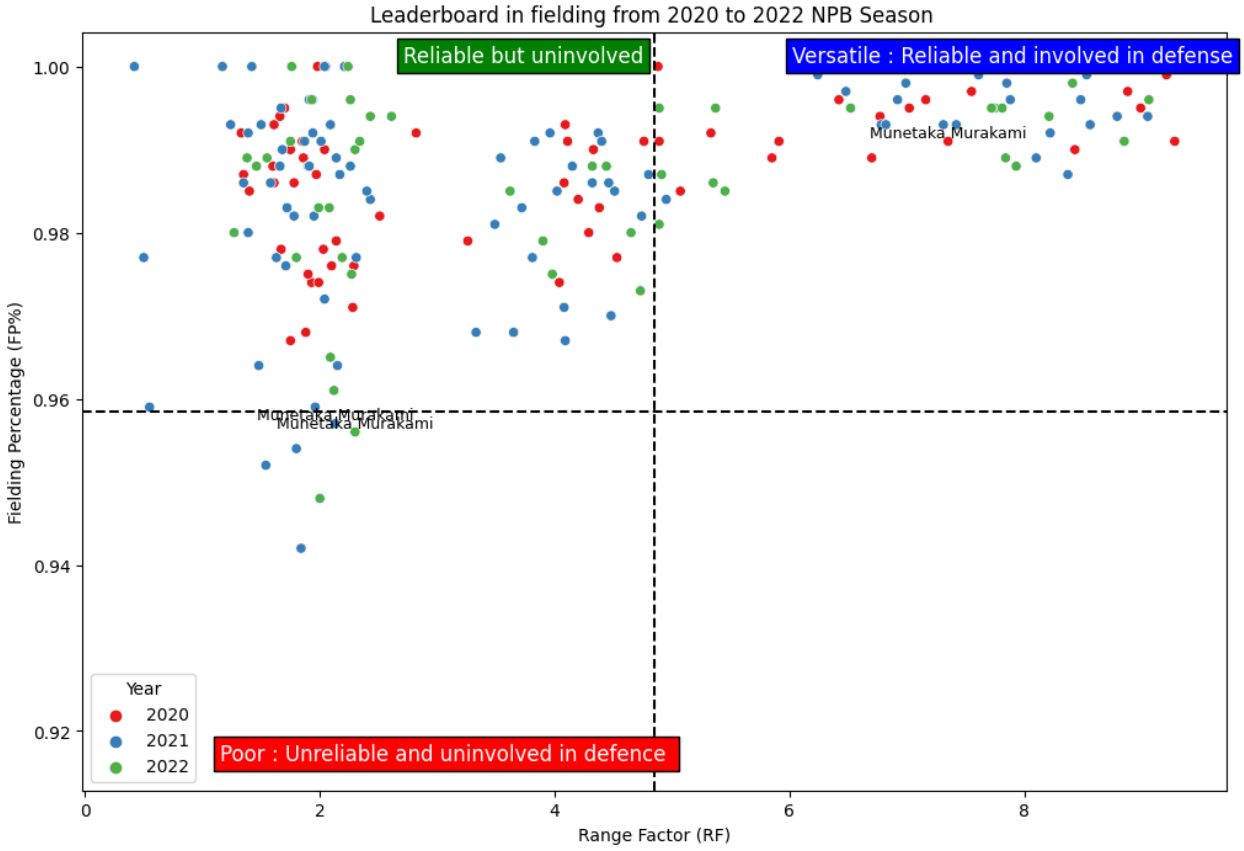
Source : https://library.fangraphs.com/defense/uzr/

```
# Read the Excel file with Fielding NPB data
fielding2022 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
fieldingnpb2022.xlsx")
fielding2021 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
fieldingnpb2021.xlsx")
fielding2020 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
fieldingnpb2020.xlsx")
# Concatenate the different variables to group data from 2020 to 2022
for teams in NPB
fielding_seasons = pd.concat([fielding2020, fielding2021,
fielding2022])
# Reset Index
fielding_seasons.reset_index(drop=True, inplace=True)
#Filter Murakami's fielding data from 2020 to 2022
murakami_fielding = fielding_seasons.loc[fielding_seasons['Player']
=='Munetaka Murakami' ]
murakami_fielding
```

```
                Player  Year  Lg   Tm Pos    G     PO    A   E
DP  ...    RF  \
49   Munetaka Murakami  2020  CL  YAK  1B   94  640.0   51   6
49  ...  7.35
137  Munetaka Murakami  2021  CL  YAK  3B  137   96.0  196  13
17  ...  2.13
179  Munetaka Murakami  2022  CL  YAK  3B  141   93.0  232  15
19  ...  2.30

     RRF  rRng  rErr    rWP    rPB   rCth  rDP   rArm  UZR
49   0.2  -4.6  -0.2 -999.0 -999.0 -999.0 -0.2 -999.0 -5.0
137  2.2  -5.6  -0.2 -999.0 -999.0 -999.0 -0.7 -999.0 -6.5
179 -4.5  -4.9  -0.1 -999.0 -999.0 -999.0 -0.2 -999.0 -5.2

[3 rows x 23 columns]
```

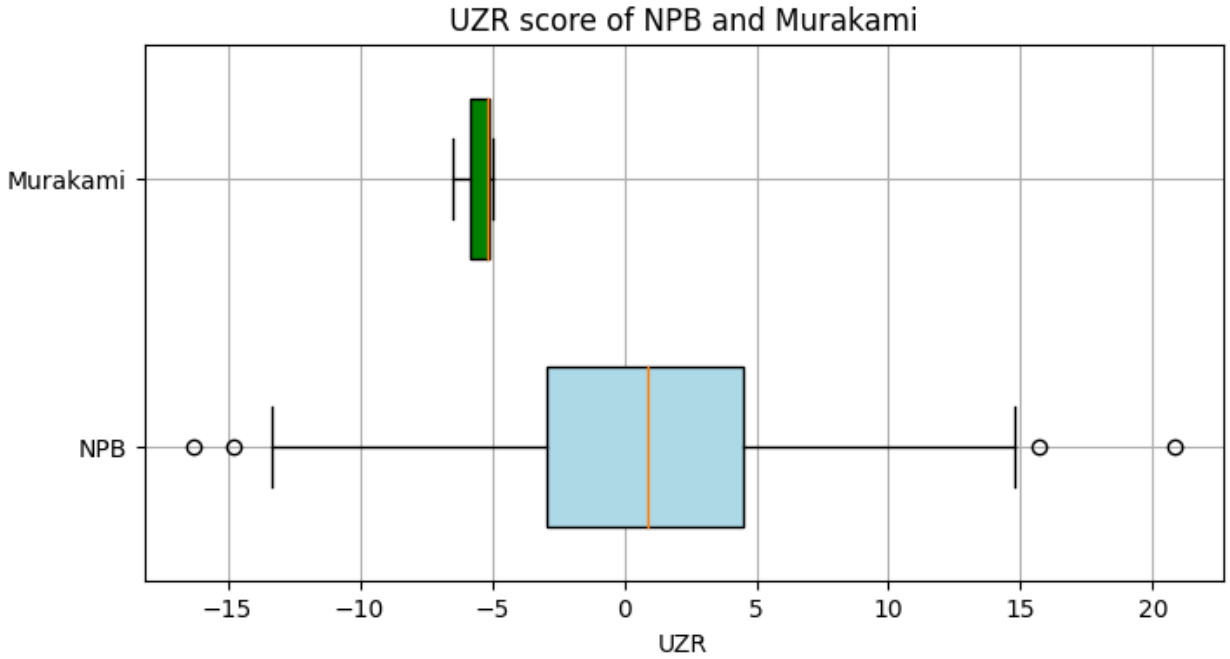
```
# We have decided to create a scatterplot, with the "Range Factor" in
x-axis and "Fielding Percentage" in y-axis
f = sns.scatterplot(x="RF", y="FP", data=fielding_seasons, hue="Year",
palette="Set1")
plt.gcf().set_size_inches(12, 8)

# Use a loop to indicate Murakami's year performance in each point
for index, row in murakami_fielding.iterrows():
    plt.text(row["RF"], row["FP"], row["Player"],fontsize=9,
ha='center', va='bottom')

# We have created a grid to better visualize the level of each athlete
plt.axhline(y=f.axes.get_ylim()[0] + (f.axes.get_ylim()[1] -
f.axes.get_ylim()[0]) / 2, color='k', linestyle='--')
plt.axvline(x=f.axes.get_xlim()[0] + (f.axes.get_xlim()[1] -
f.axes.get_xlim()[0]) / 2, color='k', linestyle='--')
# We have added differents commentary labels to better understand the
level of each baseball player
plt.text(f.axes.get_xlim()[0] + 0.28 * (f.axes.get_xlim()[1] -
f.axes.get_xlim()[0]),
         f.axes.get_ylim()[1] - 0.04 * (f.axes.get_ylim()[1] -
f.axes.get_ylim()[0]),
         "Reliable but uninvolved", fontsize=12, color='white',
bbox=dict(facecolor='green', edgecolor='black', boxstyle='square'))
plt.text(f.axes.get_xlim()[0] + 0.12 * (f.axes.get_xlim()[1] -
f.axes.get_xlim()[0]),
         f.axes.get_ylim()[0] + 0.04 * (f.axes.get_ylim()[1] -
f.axes.get_ylim()[0]),
         "Poor : Unreliable and uninvolved in defence ", fontsize=12,
color='white', bbox=dict(facecolor='red', edgecolor='black',
boxstyle='square') )
plt.text(f.axes.get_xlim()[1] - 0.38 * (f.axes.get_xlim()[1] -
f.axes.get_xlim()[0]),
         f.axes.get_ylim()[1] - 0.04 * (f.axes.get_ylim()[1] -
```

```
f.axes.get_ylim()[0]),
         "Versatile : Reliable and involved in defense", fontsize=12,
color='white', bbox=dict(facecolor='blue', edgecolor='black',
boxstyle='square'))

# Creating scatterplot's labels, legend and title
plt.xlabel('Range Factor (RF)')
plt.ylabel('Fielding Percentage (FP%)')
plt.title("Leaderboard in fielding from 2020 to 2022 NPB Season ")
plt.show()
```




```
# We have decided to create two boxplotplot, with the Murakami and NPB
players "UZR" data from 2020 to 2022
plt.figure(figsize=(8, 4))
# We choose the color green for Murakami and lighblue for NPB players
plt.boxplot(fielding_seasons['UZR'], vert=False, positions=[0],
widths=0.6, patch_artist=True, boxprops=dict(facecolor='lightblue'))
plt.boxplot(murakami_fielding['UZR'], vert=False, positions=[1],
widths=0.6, patch_artist=True, boxprops=dict(facecolor='green'))
# Creating boxplot's labels, legend and title
plt.title('UZR score of NPB and Murakami')
plt.xlabel('UZR')
plt.yticks([0, 1], ['NPB', 'Murakami'])
```

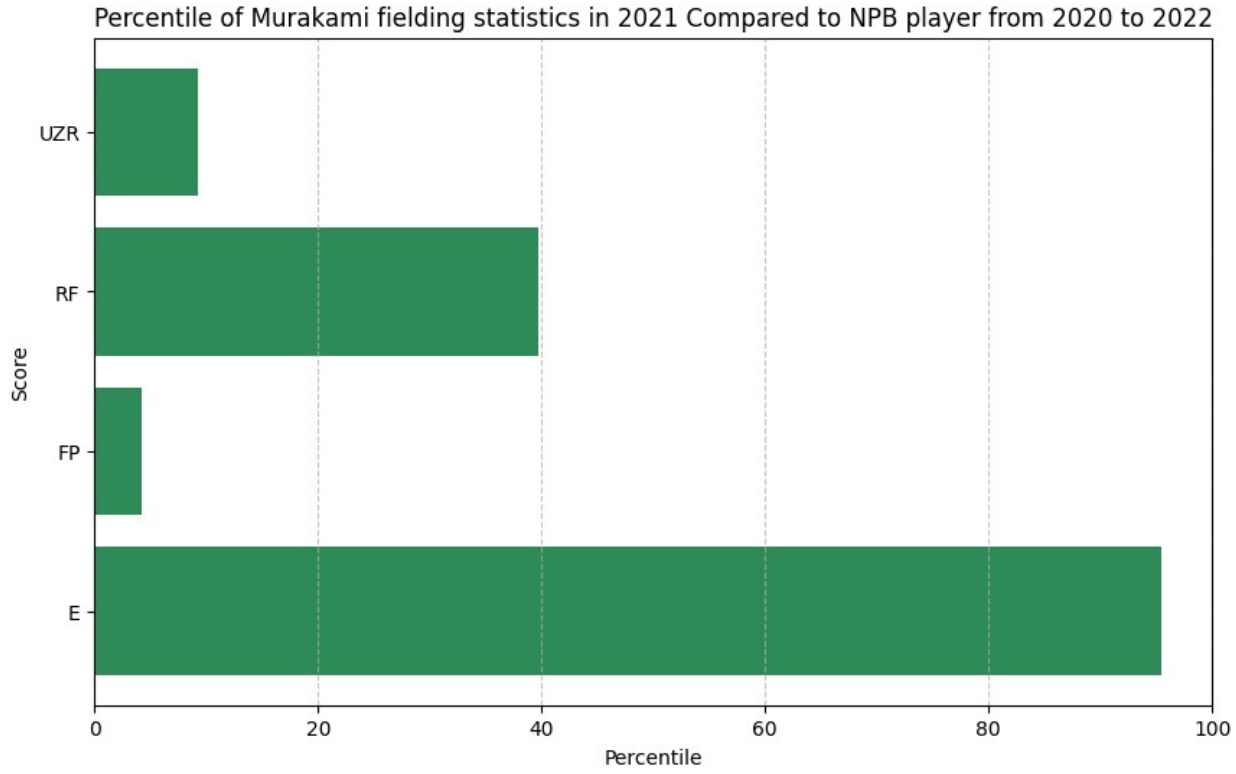
```
plt.grid(True)
plt.show()
```




```
# Filter Murakami's 2021 Fielding Data
murakami_fielding_2021 =
murakami_fielding.loc[murakami_fielding['Year'] == 2021 ]

# We store the values we want to analyze in a variable
fieldings = ['E', 'FP', 'RF', 'UZR']
percentiles_fielding = []

# We create a loop to find out in which percentiles each value of the
“scores” variable
for fielding in fieldings:
    murakami_field = murakami_fielding_2021[fielding].values[0]
    percentile_fielding =
percentileofscore(fielding_seasons[fielding], murakami_field)
    percentiles_fielding.append(percentile_fielding)
    print(f"In 2021, Murakami was at the {percentile_fielding:.2f}th
percentile of NPB player from 2020 to 2022 in terms of {fielding}
score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(fieldings, percentiles_fielding, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Murakami fielding statistics in 2021 Compared
```

```
to NPB player from 2020 to 2022')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

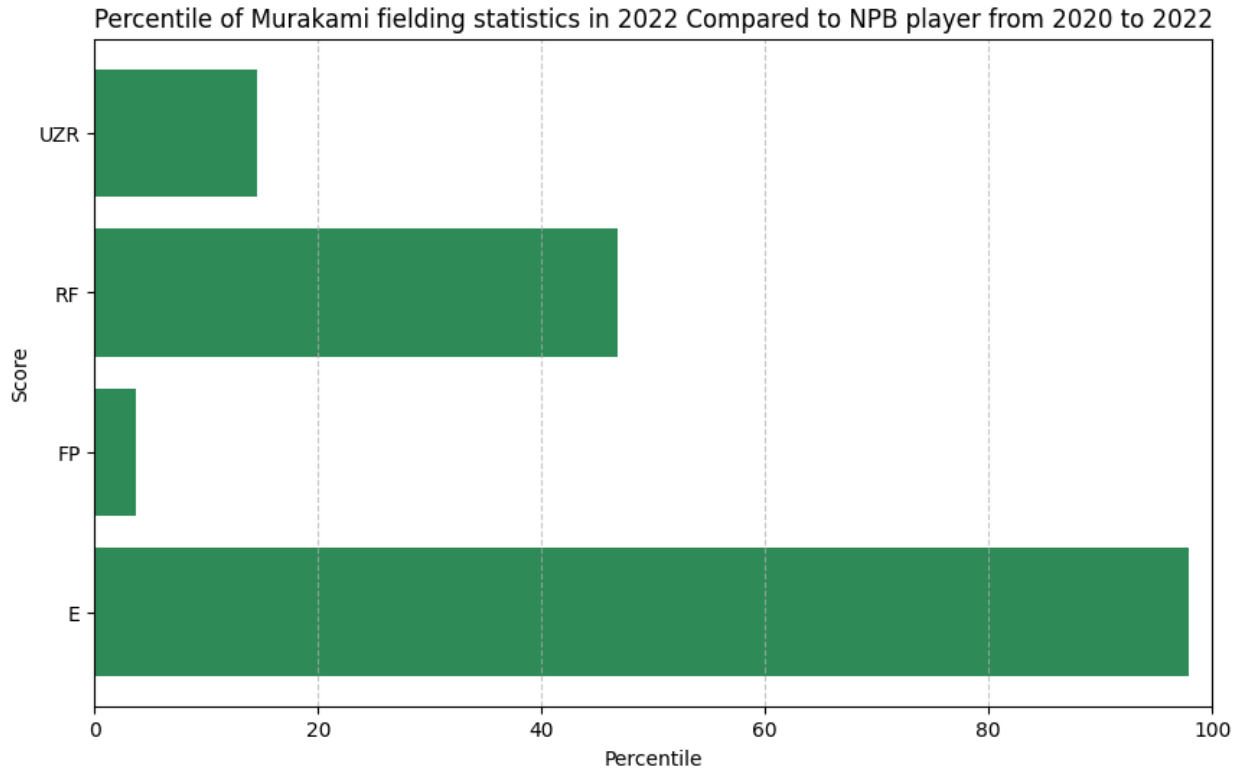
```
In 2021, Murakami was at the 95.50th percentile of NPB player from
2020 to 2022 in terms of E score.
In 2021, Murakami was at the 4.23th percentile of NPB player from 2020
to 2022 in terms of FP score.
In 2021, Murakami was at the 39.68th percentile of NPB player from
2020 to 2022 in terms of RF score.
In 2021, Murakami was at the 9.26th percentile of NPB player from 2020
to 2022 in terms of UZR score.
```




```
# Filter Murakami's 2022 Fielding Data
murakami_fielding_2022 =
murakami_fielding.loc[murakami_fielding['Year'] == 2022 ]

# We store the values we want to analyze in a variable
fieldings = ['E', 'FP', 'RF', 'UZR']
percentiles_fielding = []

# We create a loop to find out in which percentiles each value of the
“scores” variable
for fielding in fieldings:
```

```
    murakami_field = murakami_fielding_2022[fielding].values[0]
    percentile_fielding =
percentileofscore(fielding_seasons[fielding], murakami_field)
    percentiles_fielding.append(percentile_fielding)
    print(f"In 2022, Murakami was at the {percentile_fielding:.2f}th
percentile of NPB player from 2020 to 2022 in terms of {fielding}
score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(fieldings, percentiles_fielding, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Murakami fielding statistics in 2022 Compared
to NPB player from 2020 to 2022')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

```
In 2022, Murakami was at the 97.88th percentile of NPB player from
2020 to 2022 in terms of E score.
In 2022, Murakami was at the 3.70th percentile of NPB player from 2020
to 2022 in terms of FP score.
In 2022, Murakami was at the 46.83th percentile of NPB player from
2020 to 2022 in terms of RF score.
In 2022, Murakami was at the 14.55th percentile of NPB player from
2020 to 2022 in terms of UZR score.
```



In this chapter, We were able to get an overview of Murakami's defensive capabilities, and they're not great. Indeed, fielding is not one of the player's prime qualities. He makes more errors than average, and in general, he's not very effective in his offensive contributions. His fielding performances have been fairly consistent over the seasons, showing a weakness in this area of the game.

When he was positioned at 1B, he made fewer errors, but it was more a case of hiding his fielding shortcomings. It seems unlikely to us that he'll play 3B in MLB, unless the team that recruits him already has two offensive players with defensive weaknesses.

Note: he has not participated in any outfield inning. Probably due to his imposing size, but that's just a rough assumption.

Chapter 4 : Munetaka Murakami's scouting report

After analyzing Murakami's batting, plate discipline and fielding performances, we decided to highlight them in a playful way.

Inspired by the Statsbomb company's polar plot, we came up with the idea of collecting Murakami's percentile data against NPB players and producing a polar plot to observe his level in several key categories.

```
# Filter Murakami's offense data from 2020 to 2023 seasons
drop_murakami_stats = murakami[(murakami['Year'] == 2018) |
(murakami['Year'] == 2019) | (murakami['Year'] == 2024)].index
murakami_offense20to23 = murakami.drop(drop_murakami_stats)
```

```
# Selects the data and calculates the sum or average according to the
data.
g_murakami_offense20to23= murakami_offense20to23['G'].sum()
pa_murakami_offense20to23 = murakami_offense20to23['PA'].sum()
avg_murakami_offense20to23 = murakami_offense20to23['AVG'].mean()
obp_murakami_offense20to23 = murakami_offense20to23['OBP'].mean()
slg_murakami_offense20to23 = murakami_offense20to23['SLG'].mean()
ops_murakami_offense20to23 = murakami_offense20to23['OPS'].mean()
bb_murakami_offense20to23 = murakami_offense20to23['BB%'].mean()
war_murakami_offense20to23 = murakami_offense20to23['WAR'].mean()

# Creation of a dictionary for each data item collected
offense_total_dict = {
    'G': g_murakami_offense20to23,
    'PA': pa_murakami_offense20to23,
    'BB%' : bb_murakami_offense20to23,
    'AVG': avg_murakami_offense20to23,
    'OBP' : obp_murakami_offense20to23,
    'SLG' : slg_murakami_offense20to23,
    'OPS' : ops_murakami_offense20to23,
    'WAR' : war_murakami_offense20to23
}

murakami_offense_total = pd.Series(offense_total_dict)

# Creation of an average for each data item
murakami_mean_game = murakami_offense_total['G']/4
murakami_offense_total['PA'] =
(murakami_offense_total['PA']/murakami_offense_total['G'])*murakami_me
an_game
murakami_offense_total['G'] = murakami_mean_game
murakami_offense_total
```

```
G      136.00000
PA     584.75000
BB%      0.17125
AVG      0.28975
OBP      0.41700
SLG      0.59025
OPS      1.00725
WAR      6.22500
dtype: float64
```

```
# Filter Murakami's Fielding data from 2020 to 2023 seasons
uzr_murakami_fielding20to23 = murakami_fielding['UZR'].mean()
rf_murakami_fielding20to23 = murakami_fielding['RF'].mean()

# Creation of a dictionary for each data item collected
fielding_total_dict = {
```

```
    'G': murakami_mean_game,
    'UZR': uzr_murakami_fielding20to23,
    'RF' : rf_murakami_fielding20to23
}

murakami_fielding_total = pd.Series(fielding_total_dict)
murakami_fielding_total
```

```
G      136.000000
UZR     -5.566667
RF       3.926667
dtype: float64
```

```
# Filter Murakami's Plate Discipline data from 2020 to 2023 seasons
O_Swing_Murakami_Plate_Discipline = Murakami_Plate_Discipline['O-Swing
%'].mean()
Z_Swing_Murakami_Plate_Discipline = Murakami_Plate_Discipline['Z-Swing
%'].mean()
Contact_Murakami_Plate_Discipline = Murakami_Plate_Discipline['Contact
%'].mean()

# Creation of a dictionary for each data item collected
Plate_total_dict = {
    'G': murakami_mean_game,
    'O-Swing%': O_Swing_Murakami_Plate_Discipline,
    'Z-Swing%' : Z_Swing_Murakami_Plate_Discipline,
    "Contact%" : Contact_Murakami_Plate_Discipline
}

murakami_plate_discipline_total = pd.Series(Plate_total_dict)
murakami_plate_discipline_total
```

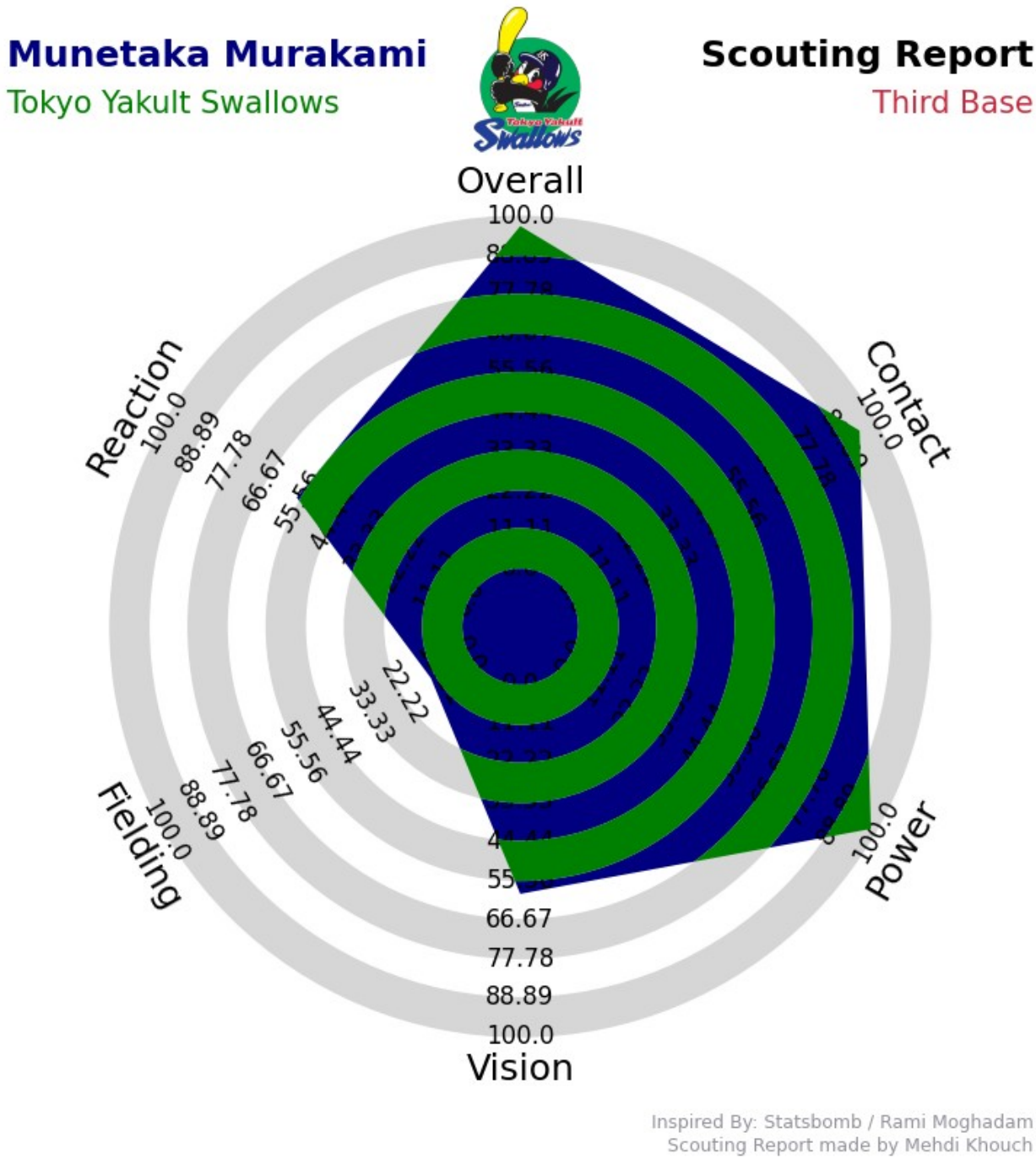
```
G           136.000
O-Swing%     24.600
Z-Swing%     67.325
Contact%     69.275
dtype: float64
```

```
# We store the values we want to analyze in a variable
scores_overall_batting = ['WAR', 'OBP', 'SLG', "BB%"]
scores_labels_batting = ['Overall',"Contact", "Power", "BB%"]
percentiles_overall_batting = []

# We create a loop to find out in which percentiles each value of the
“scores” variable
for score in scores_overall_batting:
    murakami_score = murakami_offense_total[score]
    percentile_overall_batting = percentileofscore(leadernpb[score],
murakami_score)
    percentiles_overall_batting.append(percentile_overall_batting)
```

```
# We store the values we want to analyze in a variable
scores_overall_fielding = ['UZR', 'RF']
scores_labels_fielding = ['Fielding', 'Reaction' ]
percentiles_overall_fielding = []

# We create a loop to find out in which percentiles each value of the
“scores” variable
for score in scores_overall_fielding:
    murakami_score = murakami_fielding_total[score]
    percentile_overall_fielding =
percentileofscore(fielding_seasons[score], murakami_score)
    percentiles_overall_fielding.append(percentile_overall_fielding)

# We store the values we want to analyze in a variable
scores_overall_plate_discipline = ['O-Swing%', 'Z-Swing%', "Contact
%" ]
scores_labels_plate_discipline = ['Vision', 'Vision', "Vision" ]
percentiles_overall_plate_discipline = []

# We create a loop to find out in which percentiles each value of the
“scores” variable
for score in scores_overall_plate_discipline:
    murakami_score = murakami_plate_discipline_total[score]
    percentile_overall_plate_discipline =
percentileofscore(Plate_Discipline_Leader[score], murakami_score)

percentiles_overall_plate_discipline.append(percentile_overall_plate_d
iscipline)

# We invert the percentile of the “O-Swing%” data so that Murakami's
positive performance is better illustrated in this area.
percentiles_overall_plate_discipline[0] =  100 -
percentiles_overall_plate_discipline[0]
#We add up the 4 data related to Murakami's marble vision and average
them.
percentiles_overall_plate_discipline =
(percentiles_overall_plate_discipline[0] +
percentiles_overall_plate_discipline[1] +
percentiles_overall_plate_discipline[2] +
percentiles_overall_batting[3])/4
percentiles_overall_plate_discipline =
[percentiles_overall_plate_discipline]
scores_overall_labels_plate_discipline = ['Vision']

percentiles_overall_batting.pop()
scores_labels_batting.pop()

# We concatenate the set of percentiles and labels in the two
variables
percentiles_test_combined = percentiles_overall_batting +
```

```
percentiles_overall_plate_discipline + percentiles_overall_fielding
scores_labels_total = scores_labels_batting  +
scores_overall_labels_plate_discipline + scores_labels_fielding

# We create a range from 0 to 100, to highlight Murakami's level in
each discipline according to its percentile.
ranges_overall_data = [(0, 100), (0, 100), (0, 100), (0, 100), (0,
100), (0, 100)]

# Creating polar plot's labels, legend and title
radar = Radar(label_fontsize=18, range_fontsize=12)
title_radar = dict(
    title_name='Munetaka Murakami',
    title_color='navy',
    subtitle_name='Tokyo Yakult Swallows',
    subtitle_color='green',
    title_name_2="Scouting Report",
    title_color_2="black",
    subtitle_name_2='Third Base',
    subtitle_color_2='#C72C41',
    title_fontsize=18,
    subtitle_fontsize=15,
)
endnote = "Scouting Report made by Mehdi Khouch"
# Creation of Murakami's polar plot
radar.plot_radar(ranges=ranges_overall_data,
params=scores_labels_total, values=percentiles_test_combined,
radar_color = ['navy', 'green'], title = title_radar, image=r"C:\
Users\Mehdi\Desktop\NPB\tokyoyakultpicture.png", image_coord=[0.495,
0.805, 0.04, 0.1], endnote=endnote)
```

```
(<Figure size 2000x1000 with 2 Axes>, <AxesSubplot:>)
```



Chapter 5 : Comparison between Murakami and Japanese Superstars played or is playing in MLB

For this final chapter of the analysis, we decided to analyze Murakami's performances against 6 great Japanese players who have played or are playing in MLB, having broken records in Japan during the 21st century.

We decided to take 3 contemporary players (Shohei Ohtani, Seiya Suzuki and Masataka Yoshida) and 3 retired players (Ichiro Suzuki, Hideki Matsui and Kosuke Fukudome). These 6 players have had different experiences, some of them becoming All-Stars or even Hall of Famers (Ohtani, Ichiro and Matsui), others becoming good or complementary players.

This will allow us to compare Murakami's level with his compatriots at the same age, to get a better idea of his current level and to imagine his adaptation to MLB.

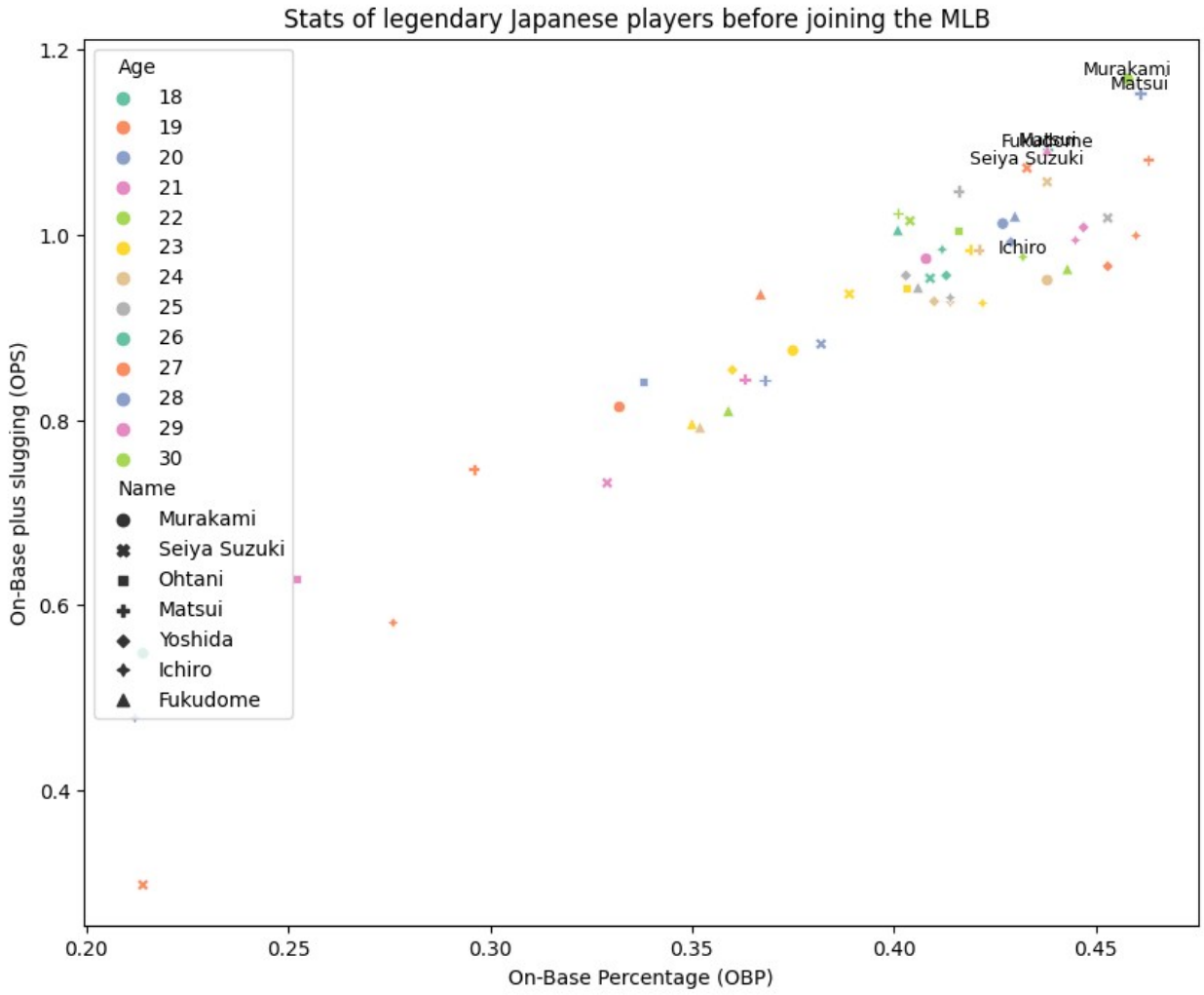
```
# Filter Japanese Superstars data in NPB
Seiya = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\seiyasuzuki.xlsx")
Ohtani =  pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\ohtani.xlsx")
Matsui = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\matsui.xlsx")
Yoshida = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\yoshida.xlsx")
Ichiro = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\ichiro.xlsx")
Fukudome = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\Fukudome.xlsx")

# Concatenate the different variables to group data for Japanese
Superstars data in NPB
japanese_legend = pd.concat([murakami, Seiya, Ohtani, Matsui, Yoshida,
Ichiro, Fukudome])
japanese_legend = japanese_legend.dropna(axis= 1)
japanese_legend.head()
```

```
   Year      Name  Age Pos    G   PA    H  HR  RBI  SB    BB%     K%
AVG  \
0  2018  Murakami   18  3B    6   14    1   1    2   0  0.143  0.357
0.083
1  2019  Murakami   19  1B  143  593  118  36   96   5  0.125  0.310
0.231
2  2020  Murakami   20  1B  120  515  130  28   86  11  0.169  0.223
0.307
3  2021  Murakami   21  3B  143  615  139  39  112  12  0.172  0.216
0.278
4  2022  Murakami   22  3B  141  612  155  56  134  12  0.193  0.209
0.318

     OBP    SLG    OPS  BABIP   WAR
0  0.214  0.333  0.548  0.000  -0.1
1  0.332  0.481  0.814  0.279   1.0
2  0.427  0.585  1.012  0.362   4.7
3  0.408  0.566  0.974  0.302   6.3
4  0.458  0.710  1.168  0.327  10.2
```

```
# Filter Murakami's Data
murakami_name = japanese_legend.loc[japanese_legend['Name'] ==
'Murakami']
# Filter WAR high score data for Japanese Superstars
filterwar = japanese_legend.sort_values(by="WAR", ascending=False)
topwar = filterwar.head(6)
topwar
```

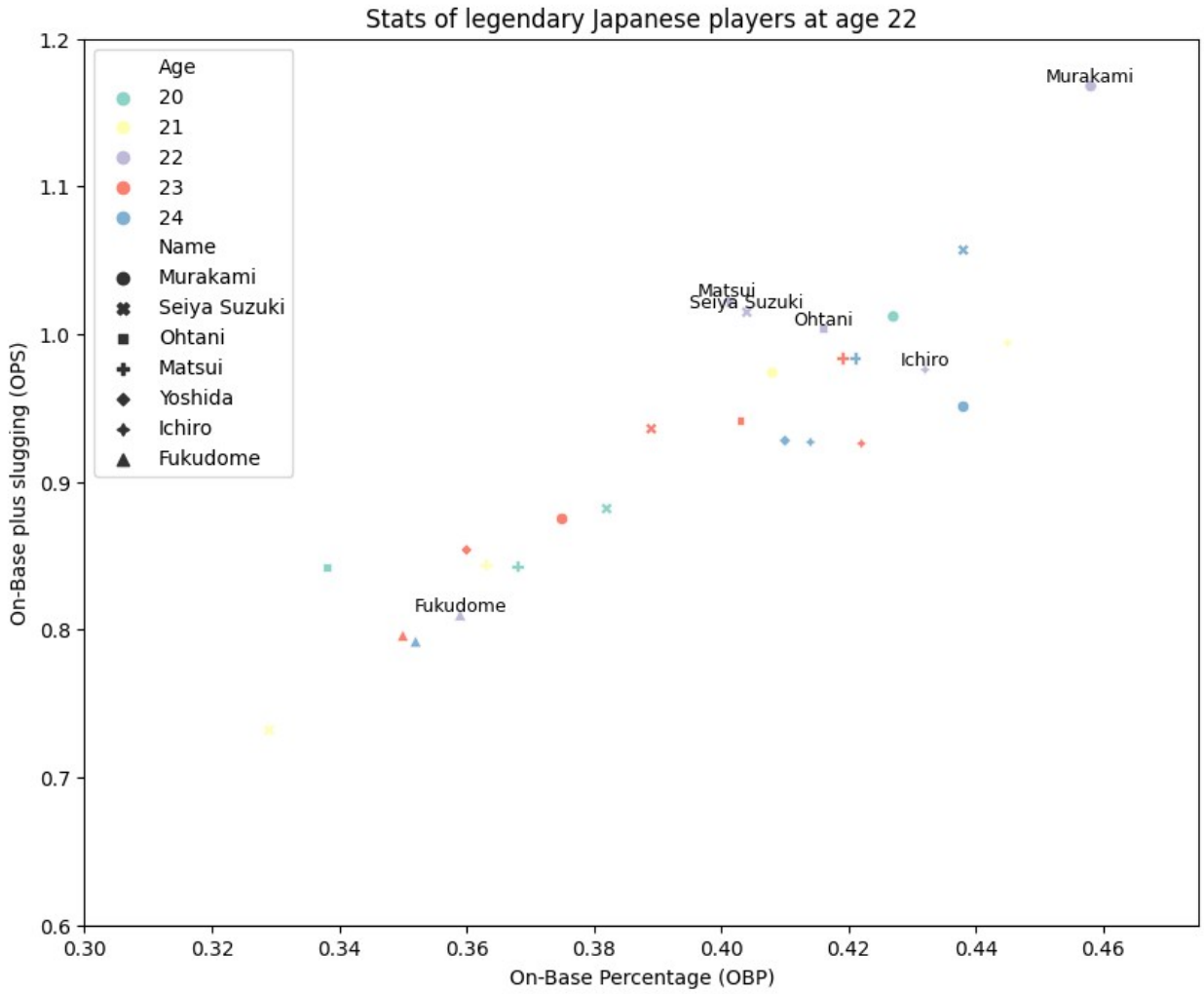
```
   Year           Name  Age Pos    G   PA    H  HR  RBI  SB    BB%
K%  \
7  2006      Fukudome   29  RF  130  578  174  31  104  11  0.131
0.163
9  2002        Matsui   28  CF  140  623  167  50  107   3  0.183
0.167
3  1995        Ichiro   22  CF  130  613  179  25   80  49  0.111
0.085
4  2022      Murakami   22  3B  141  612  155  56  134  12  0.193
0.209
7  2000        Matsui   26  CF  135  590  150  42  108   5  0.180
0.183
8  2021  Seiya Suzuki   27  RF  132  533  138  38   88   9  0.163
0.165

     AVG    OBP    SLG    OPS  BABIP   WAR
7  0.351  0.438  0.653  1.091  0.382  11.2
9  0.334  0.461  0.692  1.153  0.335  11.1
3  0.342  0.432  0.544  0.976  0.342  10.6
4  0.318  0.458  0.710  1.168  0.327  10.2
7  0.316  0.438  0.654  1.092  0.326   9.7
8  0.317  0.433  0.639  1.072  0.318   9.6
```

```
# We have decided to create a scatterplot, with the "OBP" in x-axis
and "OPS" in y-axis
sns.scatterplot(x="OBP", y="OPS", data = japanese_legend, hue="Age",
style="Name", palette="Set2")
plt.gcf().set_size_inches(10, 8)

# Use a loop to indicate Top WAR performances in each point
for index, row in topwar.iterrows():
    plt.text(row["OBP"], row["OPS"], row["Name"],fontsize=9,
ha='center', va='bottom')
# Creating scatterplot's labels, legend and title
plt.xlabel('On-Base Percentage (OBP)')
plt.ylabel('On-Base plus slugging (OPS)')
plt.title("Stats of legendary Japanese players before joining the MLB
")
plt.show()
```




```
# Filter data on Japanese Superstars between 20 and 25 years old
before25 = japanese_legend.loc[(japanese_legend['Age'] >= 20) &
(japanese_legend['Age'] <= 24)]
before25.sort_values(by="WAR", ascending=False)
#Filter Japanese Superstars data at age 22
age22 = japanese_legend.loc[japanese_legend['Age'] == 22]

# We have decided to create a scatterplot, with the "OBP" in x-axis
and "OPS" in y-axis
ux = sns.scatterplot(x="OBP", y="OPS", data = before25, hue="Age",
style="Name", palette="Set3")
plt.gcf().set_size_inches(10, 8)
# Use a loop to indicate 22 years old players performances in each
point
for index, row in age22.iterrows():
    plt.text(row["OBP"], row["OPS"], row["Name"],fontsize=9,
ha='center', va='bottom')
# Creating scatterplot's labels, legend and title
ux.set_xlim(0.30, 0.475)
```

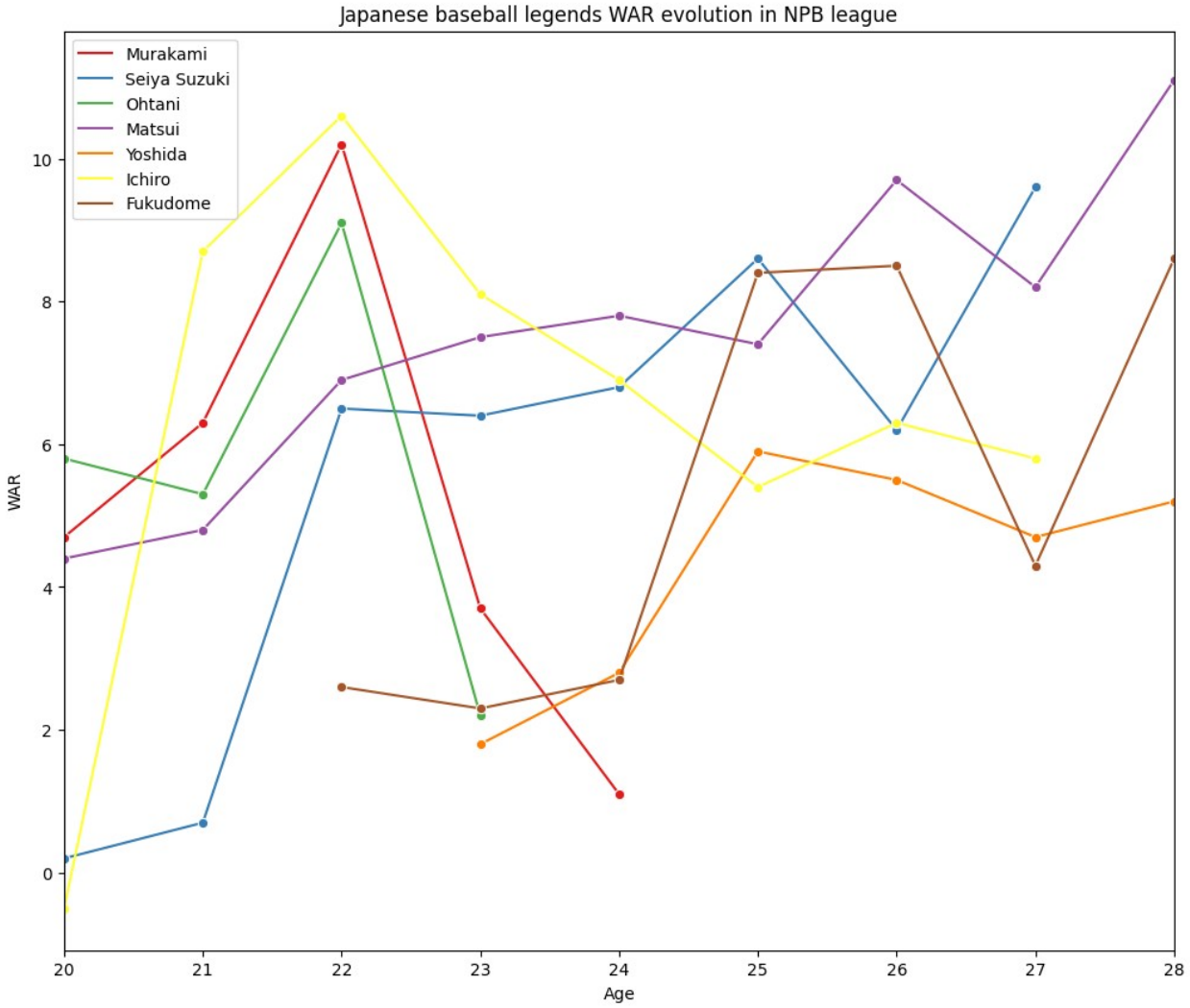
```
ux.set_ylim(0.6, 1.2)
plt.xlabel('On-Base Percentage (OBP)')
plt.ylabel('On-Base plus slugging (OPS)')
plt.title("Stats of legendary Japanese players at age 22")
plt.legend(loc="upper left")
plt.show()
plt.show()
```



Warning : It's normal that Murakami have a low WAR score in 2024 (during his 24 years old) because the WAR score climbs with each game played. So far, he has only player 32 games and 144 Plate Appearance. Same probleme with Ohtani, who didn't play as many games as the league average, due to his dual role as pitcher and batting.

```
# We have decided to create a lineplot, with the "Age" in x-axis and
"WAR" in y-axis
ax = sns.lineplot(x="Age", y="WAR", data=japanese_legend, hue="Name",
marker='o', palette="Set1" )
plt.gcf().set_size_inches(12, 10)
```

```
ax.set_xlim(20, 28)
# Creating lineplot's labels, legend and title
plt.xlabel("Age")
plt.ylabel("WAR")
plt.title("Japanese baseball legends WAR evolution in NPB league")
legend = plt.legend(loc="upper left")
plt.show()
```



In the plot "Stats of legendary Japanese players before joining the MLB ", we can see that Murakami's 2022 season have a similiar record than Hideki Matsui. Indeed, offensively, the two players have a lot in common. Like their size, but also their ability to hit home runs.

The main difference is that Murakami was 6 years younger when he reached this level of performance. Matsui was already an established 28-year-old before joining the New York Yankees.

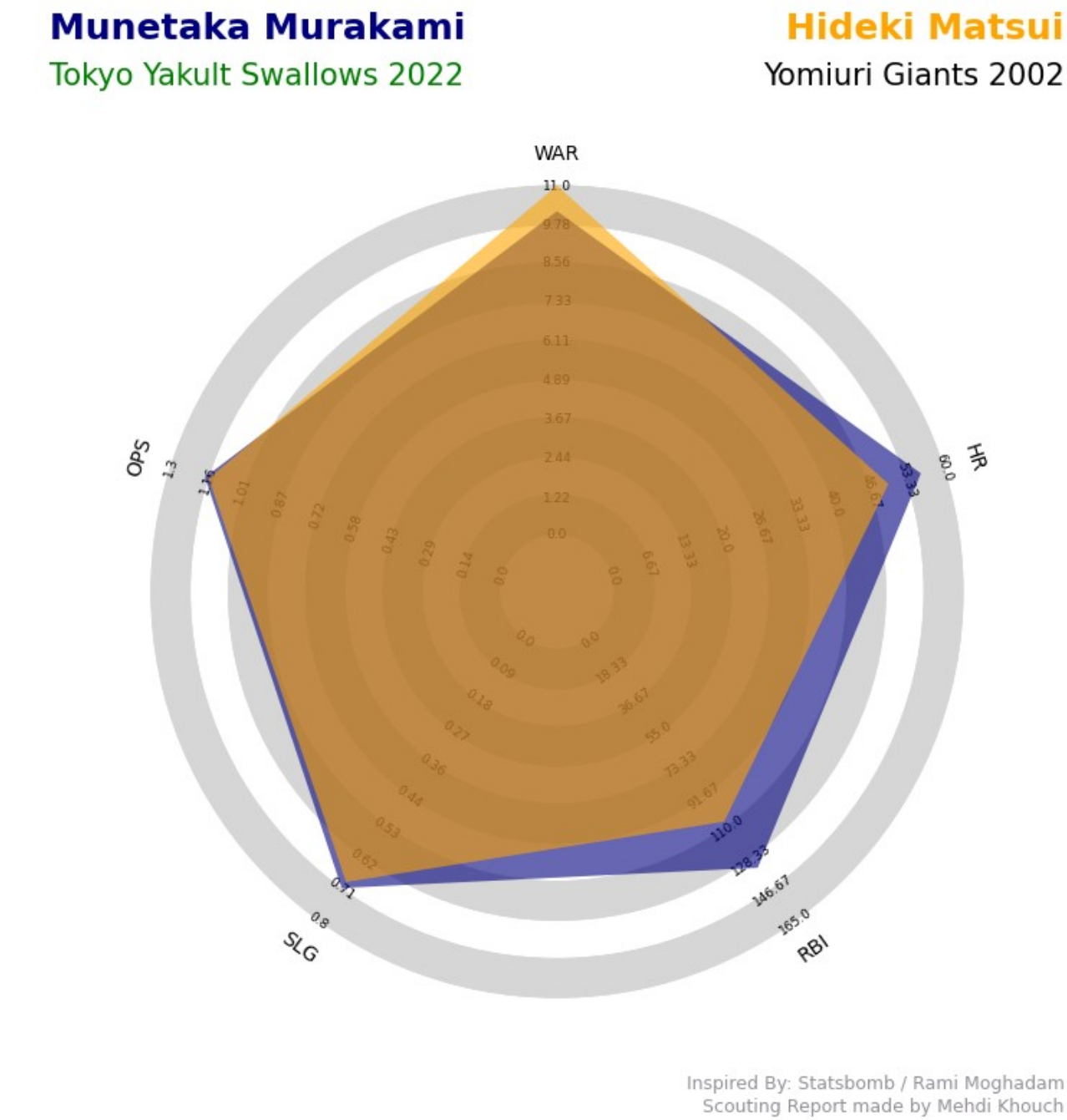
```
# Filter Murakami's 2022 season and Matsui's 2002 season data
Murakami_Matsui = topwar.loc[(topwar["Year"] == 2002) |
(topwar["Year"] == 2022)]
Murakami_Matsui
```

```
   Year      Name  Age Pos    G   PA    H  HR  RBI  SB    BB%     K%
AVG  \
9  2002    Matsui   28  CF  140  623  167  50  107   3  0.183  0.167
0.334
4  2022  Murakami   22  3B  141  612  155  56  134  12  0.193  0.209
0.318

     OBP    SLG    OPS  BABIP   WAR
9  0.461  0.692  1.153  0.335  11.1
4  0.458  0.710  1.168  0.327  10.2
```

```
# Filter Murakami's 2022 season data
Murakami_2022 = topwar.loc[(topwar["Year"] == 2022) & (topwar["Name"]
== "Murakami")].iloc[0]
# Filter Matsui's 2002 season data
Matsui_2002 = topwar.loc[(topwar["Year"] == 2002) & (topwar["Name"] ==
"Matsui")].iloc[0]

# We store the values we want to analyze in a variable
params_Mur_Mat = ["WAR", "HR", "RBI", "SLG", "OPS"]

# We create a range for each data item
ranges_Mur_Mat = [(0.0, 11.0), (0.0, 60.0), (0.0, 165.0), (0.0, 0.8),
(0.0, 1.3)]

# We create a loop to find the data for each parameter
values_Mur_Mat = [
    [Murakami_2022[param] for param in params_Mur_Mat],
    [Matsui_2002[param] for param in params_Mur_Mat]
]

# Creating polar plot's labels, legend and title
title = dict(
    title_name='Munetaka Murakami',
    title_color='navy',
    subtitle_name='Tokyo Yakult Swallows 2022',
    subtitle_color='green',
    title_name_2='Hideki Matsui',
    title_color_2='orange',
    subtitle_name_2='Yomiuri Giants 2002',
    subtitle_color_2='black',
    title_fontsize=18,
    subtitle_fontsize=15,
)
endnote = "Scouting Report made by Mehdi Khouch"
```

```
# Creation of polar plot
radar = Radar()
radar.plot_radar(ranges=ranges_Mur_Mat, params=params_Mur_Mat,
values=values_Mur_Mat,
                            radar_color=['navy', 'orange'],
                            title=title, endnote = endnote,
compare=True)
```

```
(<Figure size 2000x1000 with 1 Axes>, <AxesSubplot:>)
```



In this chapter, we could see that Murakami had put together a legendary 2022 season, even against the best Japanese players of the 21st century. At 22, he was even one level above his compatriots of the same age, even the most precocious and talented like Ohtani or Ichiro. His powerful with the bat is particularly impressive for his age.

On the other hand, his decline in 2023 marked a slight halt, but that seems to be just a rough patch.

Chapter 6 : Conclusion

During this scout report, we were able to get an overview of Munetaka Murakami's qualities and weaknesses.

His strengths include his incredible power with the bat and his patience when he comes to the plate. These are qualities that are highly sought-after in MLB, especially among Asian players who often find it hard to adapt in the USA. He also provokes an enormous amount of base-on-balls. We can clearly define him as a Power Hitter. We must not forget to mention his physical qualities, which are exceptional and perfect for the role of Power Hitter in MLB.

As for his weak points, we were able to see his shortcomings in fielding. An important shortcoming, which will certainly force him to migrate to 1st base or DH in MLB. His vision at the plate is decent to good, but he'll have to improve in MLB against better pitchers, with a higher level of velocity than in NPB.

Given his qualities and shortcomings, I like to compare Murakami with Boston Red Sox player, Rafael Devers. Rafael Devers has a very similar build to the Japanese player, and his offensive qualities are very similar too. He plays third base for the Boston team, but he also has a very low fielding percentage for his position, with similar statistics to Murakami. Two powerful players, capable of hitting a lot of homeruns, but with defensive performances below par for a 3rd baseman.

Finally, we would like to add that a lot of people have been quick to put a lot of pressure on the player, with a lot of expectation after his 2022 season. But be warned, he's still a very young player, and one of the most precocious Japanese players in history, as we saw in Chapter 5. Ichiro also had some lesser seasons, before joining the MLB in his late twenties and becoming an elite player there.

For more information on the player, I suggest you consult the work of Yakyu Cosmopolitan, which follows baseball news in Japan on a daily basis.

Bonus Chapter : Tokyo Yakult Swallows' offensive performances from 2021 to 2023 in NPB

In 2021 et 2022, the Tokyo Yakult Swallows dominates de Central Division. In 2021, they won the Japan Series and in 2022, they made it to the final against Orix Buffaloes. During these two seasons, Murakami won the MVP Title. But in 2023, Yakult Swallows finished second last position in NPB and their start to the 2024 season isn't much better.

In this chapter, we will try to highlight the Yakult Swallows batting performances to see whether the team is dependent on its star performer, or whether the drop in performance is a regular feature of the entire squad.

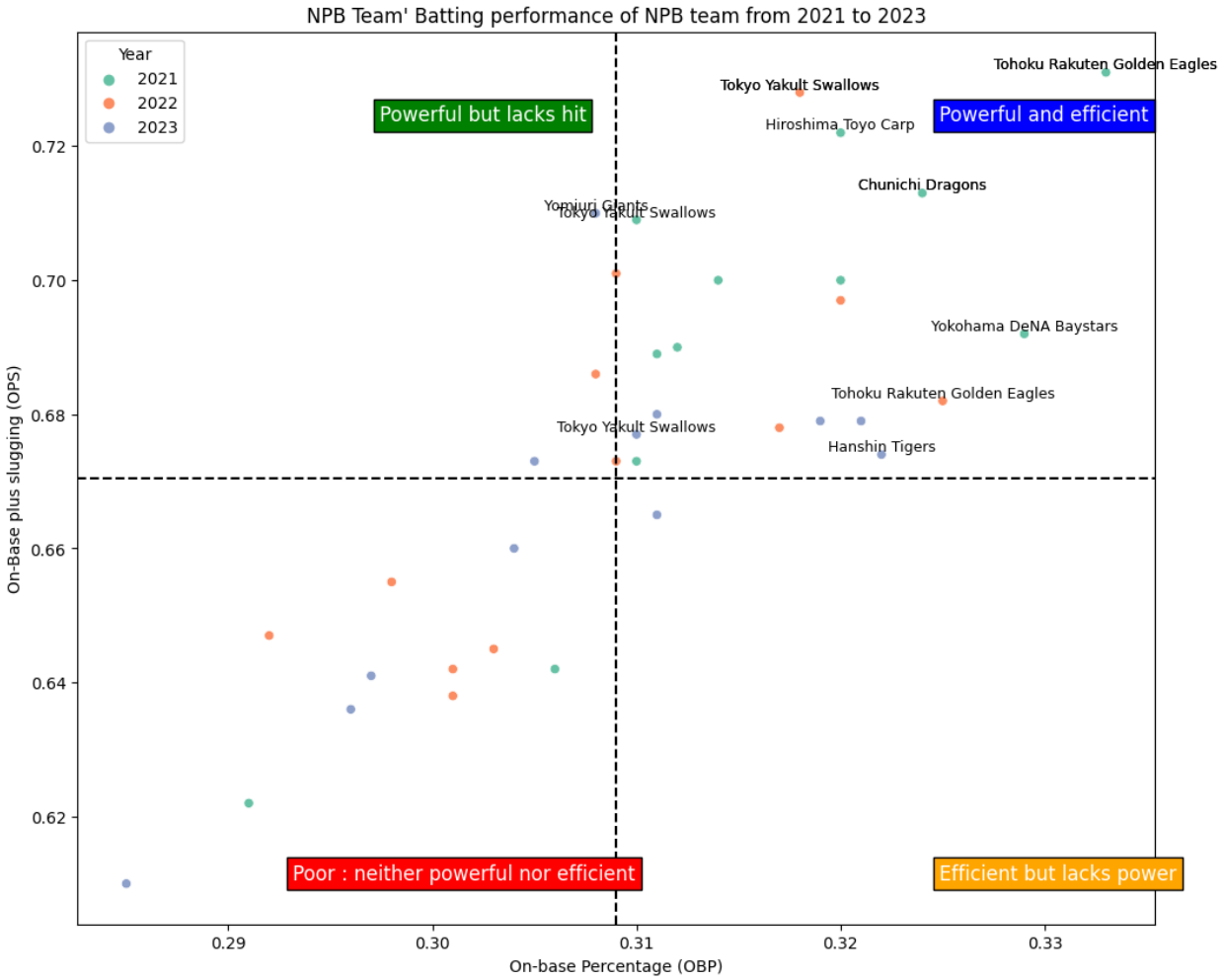
```
# Read the Excel file with NPB team data from 2022
team2021 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
NPB2021LeaderTeamBatting.xlsx")
team2022 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
NPB2022LeaderBatting.xlsx")
team2023 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
NPB2023LeaderTeamBatting.xlsx")

# Define the list of team names
team = ["Fukuoka SoftBank Hawks", "Tokyo Yakult Swallows", "Tohoku
Rakuten Golden Eagles",
        "Yomiuri Giants", "Chunichi Dragons", "Yokohama DeNA
Baystars",
        "Saitama Seibu Lions", "ORIX Buffaloes", "Hiroshima Toyo
Carp",
        "Hanshin Tigers", "Chiba Lotte Marines", "Hokkaido Nippon-Ham
Fighters"]

# Assign team names to the "Tm" column for the first 13 rows
team2021.loc[0:12, "Tm"] = team
team2022.loc[0:12, "Tm"] = team
team2023.loc[0:12, "Tm"] = team

# Concatenate the different variables to group data from 2021 to 2023
for teams in NPB
team212223 = pd.concat([team2021,team2022, team2023])
#Drop NaN values
team212223 = team212223.dropna(axis= 1)
# Reset Index
team212223.reset_index(drop=True, inplace=True)
# Filter Yakult data
Yakultdatateam = team212223.loc[team212223['Tm'] == 'Tokyo Yakult
Swallows']
Yakult21 = Yakultdatateam.loc[Yakultdatateam["Year"] == 2021]
Yakult22 = Yakultdatateam.loc[Yakultdatateam["Year"] == 2022]
Yakult23 = Yakultdatateam.loc[Yakultdatateam["Year"] == 2023]
# Filter Best team
BestTeam212223OPS = team212223.sort_values(by="OPS", ascending=False)
BestTeam212223OPS = BestTeam212223OPS.head(5)
BestTeam212223AVG = team212223.sort_values(by="OBP", ascending=False)
BestTeam212223AVG = BestTeam212223AVG.head(5)
# Display the DataFrame
team212223.head()
```

```
   Year                            Tm  Lg    PA     H   HR  RBI   SB
BB%  \
0  2021        Fukuoka SoftBank Hawks  PL  5298  1157  132  542   92
0.085
1  2021         Tokyo Yakult Swallows  CL  5241  1137  169  532   65
0.082
```

```
2  2021  Tohoku Rakuten Golden Eagles  CL  5399  1196  142  603   70
0.095
3  2021                Yomiuri Giants  CL  5265  1164  121  517  114
0.077
4  2021              Chunichi Dragons  CL  5361  1265  123  532   68
0.075

      K%    AVG    OBP    SLG    OPS
0  0.201  0.247  0.314  0.385  0.700
1  0.221  0.242  0.310  0.399  0.709
2  0.200  0.254  0.333  0.397  0.731
3  0.206  0.247  0.311  0.379  0.689
4  0.189  0.264  0.324  0.389  0.713
```

```
# We have decided to create a scatterplot, with the "OBP" in x-axis
and "OPS" in y-axis
h = sns.scatterplot(x= "OBP", y="OPS", data=team212223, hue="Year",
palette="Set2" )
plt.gcf().set_size_inches(12, 10)

# Use a loop to indicate Yakult Shallows team's year performance in
each point
for index, row in Yakultdatateam.iterrows():
    plt.text(row["OBP"], row["OPS"], row["Tm"],fontsize=9,
ha='center', va='bottom')

# Use a loop to indicate best OBP and OPS performance
for index, row in BestTeam212223OPS.iterrows():
    plt.text(row["OBP"], row["OPS"], row["Tm"],fontsize=9,
ha='center', va='bottom')
for index, row in BestTeam212223AVG.iterrows():
    plt.text(row["OBP"], row["OPS"], row["Tm"],fontsize=9,
ha='center', va='bottom')

# We have created a grid to better visualize the level of each team
plt.axhline(y=h.axes.get_ylim()[0] + (h.axes.get_ylim()[1] -
h.axes.get_ylim()[0]) / 2, color='k', linestyle='--')
plt.axvline(x=h.axes.get_xlim()[0] + (h.axes.get_xlim()[1] -
h.axes.get_xlim()[0]) / 2, color='k', linestyle='--')

# We have added differents commentary labels to better understand the
level of each team
plt.text(h.axes.get_xlim()[0] + 0.28 * (h.axes.get_xlim()[1] -
h.axes.get_xlim()[0]),
         h.axes.get_ylim()[1] - 0.10 * (h.axes.get_ylim()[1] -
h.axes.get_ylim()[0]),
         "Powerful but lacks hit", fontsize=12, color='white',
bbox=dict(facecolor='green', edgecolor='black', boxstyle='square'))
plt.text(h.axes.get_xlim()[0] + 0.20 * (h.axes.get_xlim()[1] -
h.axes.get_xlim()[0]),
```

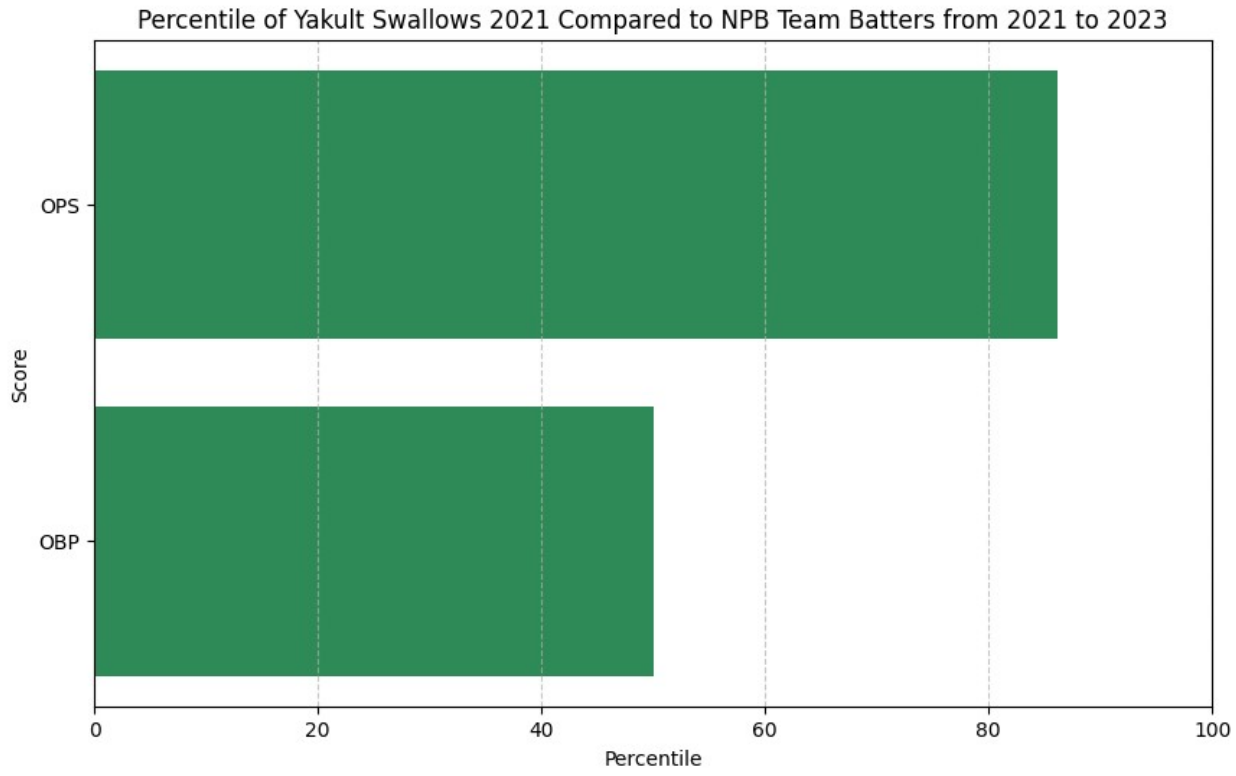
```
        h.axes.get_ylim()[0] + 0.05 * (h.axes.get_ylim()[1] -
h.axes.get_ylim()[0]),
        "Poor : neither powerful nor efficient", fontsize=12,
color='white', bbox=dict(facecolor='red', edgecolor='black',
boxstyle='square') )
plt.text(h.axes.get_xlim()[1] - 0.20 * (h.axes.get_xlim()[1] -
h.axes.get_xlim()[0]),
        h.axes.get_ylim()[1] - 0.10 * (h.axes.get_ylim()[1] -
h.axes.get_ylim()[0]),
        "Powerful and efficient", fontsize=12, color='white',
bbox=dict(facecolor='blue', edgecolor='black', boxstyle='square'))
plt.text(h.axes.get_xlim()[1] - 0.20 * (h.axes.get_xlim()[1] -
h.axes.get_xlim()[0]),
        h.axes.get_ylim()[0] + 0.05 * (h.axes.get_ylim()[1] -
h.axes.get_ylim()[0]),
        "Efficient but lacks power", fontsize=12, color='white',
bbox=dict(facecolor='orange', edgecolor='black', boxstyle='square') )

# Creating scatterplot's labels, legend and title
plt.xlabel('On-base Percentage (OBP)')
plt.ylabel('On-Base plus slugging (OPS)')
plt.title("NPB Team' Batting performance of NPB team from 2021 to
2023")
legendh = plt.legend(loc="upper left")
legendh.set_title("Year")
plt.show()
```




```
# we store the values we want to analyze in a variable
scores_team = ['OBP', 'OPS']
percentiles_team = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score_team in scores_team:
    Yakult_score = Yakult21[score_team].values[0]
    percentile_team = percentileofscore(team212223[score_team],
Yakult_score)
    percentiles_team.append(percentile_team)
    print(f"In 2021, Yakult Swallows was at the
{percentile_team:.2f}th percentile of NPB teams from 2021 to 2023 in
terms of {score_team} score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores_team, percentiles_team, color='#2E8B57')
plt.xlabel('Percentile')
```

```
plt.ylabel('Score')
plt.title('Percentile of Yakult Swallows 2021 Compared to NPB Team
Batters from 2021 to 2023')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

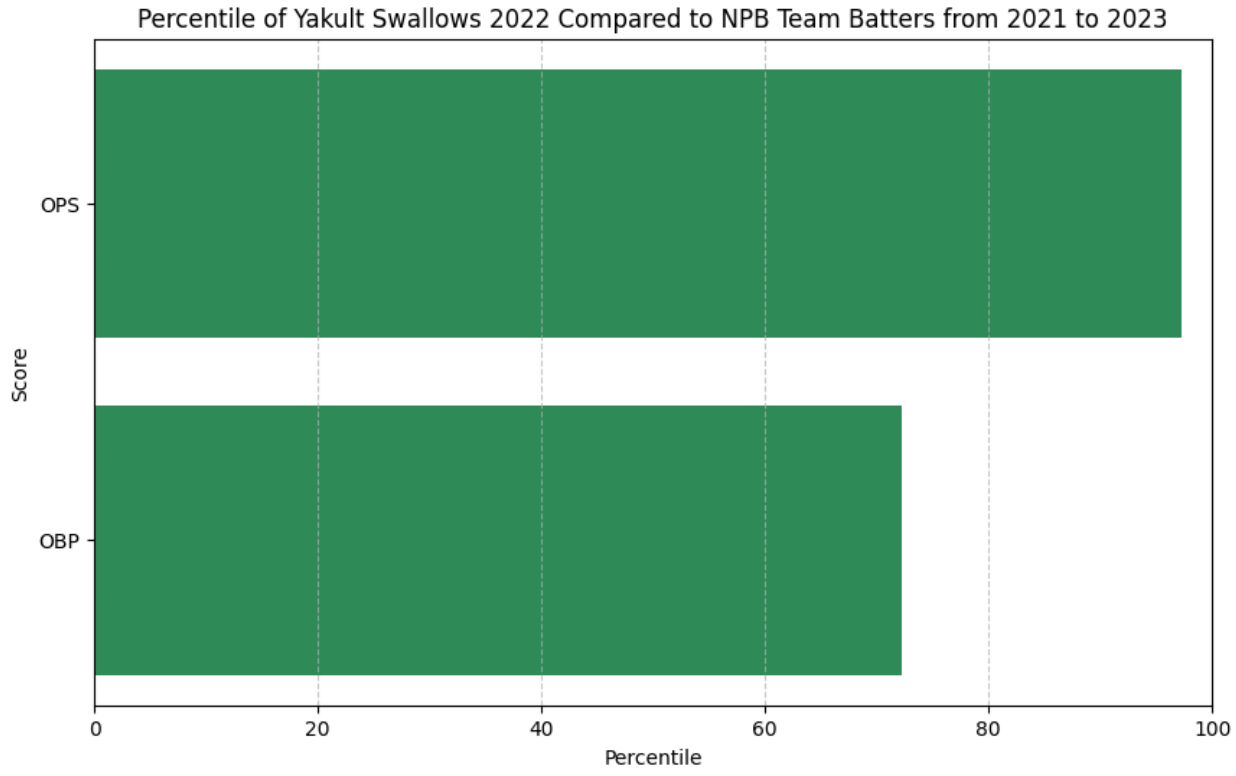
```
In 2021, Yakult Swallows was at the 50.00th percentile of NPB teams
from 2021 to 2023 in terms of OBP score.
In 2021, Yakult Swallows was at the 86.11th percentile of NPB teams
from 2021 to 2023 in terms of OPS score.
```




```
# we store the values we want to analyze in a variable
scores_team = ['OBP', 'OPS']
percentiles_team = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score_team in scores_team:
    Yakult_score = Yakult22[score_team].values[0]
    percentile_team = percentileofscore(team212223[score_team],
Yakult_score)
    percentiles_team.append(percentile_team)
    print(f"In 2022, Yakult Swallows was at the
{percentile_team:.2f}th percentile of NPB teams from 2021 to 2023 in
```

```
terms of {score_team} score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores_team, percentiles_team, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Yakult Swallows 2022 Compared to NPB Team
Batters from 2021 to 2023')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

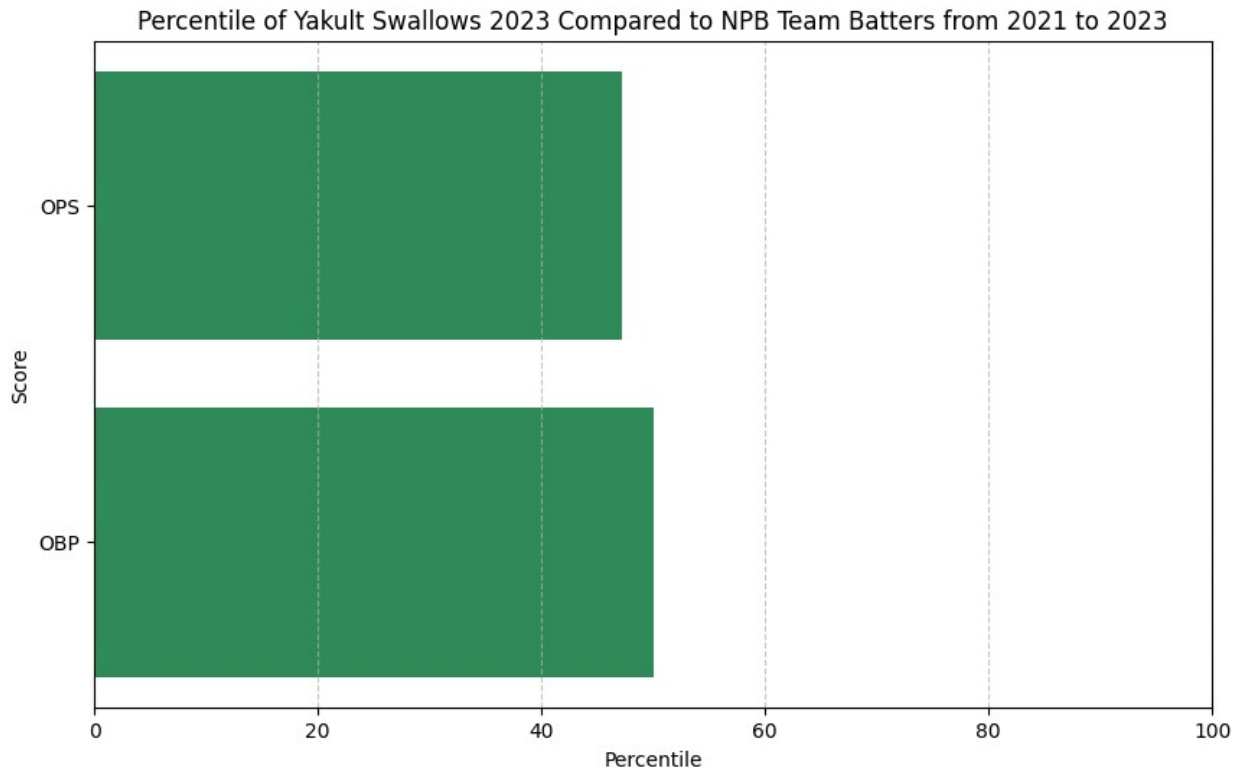
```
In 2022, Yakult Swallows was at the 72.22th percentile of NPB teams
from 2021 to 2023 in terms of OBP score.
In 2022, Yakult Swallows was at the 97.22th percentile of NPB teams
from 2021 to 2023 in terms of OPS score.
```




```
# we store the values we want to analyze in a variable
scores_team = ['OBP', 'OPS']
percentiles_team = []

# we create a loop to find out in which percentiles each value of the
“scores” variable
for score_team in scores_team:
```

```
    Yakult_score = Yakult23[score_team].values[0]
    percentile_team = percentileofscore(team212223[score_team],
Yakult_score)
    percentiles_team.append(percentile_team)
    print(f"In 2023, Yakult Swallows was at the
{percentile_team:.2f}th percentile of NPB teams from 2021 to 2023 in
terms of {score_team} score.")

# We have decided to create a barh plot to view individual data
plt.figure(figsize=(10, 6))
plt.barh(scores_team, percentiles_team, color='#2E8B57')
plt.xlabel('Percentile')
plt.ylabel('Score')
plt.title('Percentile of Yakult Swallows 2023 Compared to NPB Team
Batters from 2021 to 2023')
plt.xlim(0, 100)
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```

```
In 2023, Yakult Swallows was at the 50.00th percentile of NPB teams
from 2021 to 2023 in terms of OBP score.
In 2023, Yakult Swallows was at the 47.22th percentile of NPB teams
from 2021 to 2023 in terms of OPS score.
```

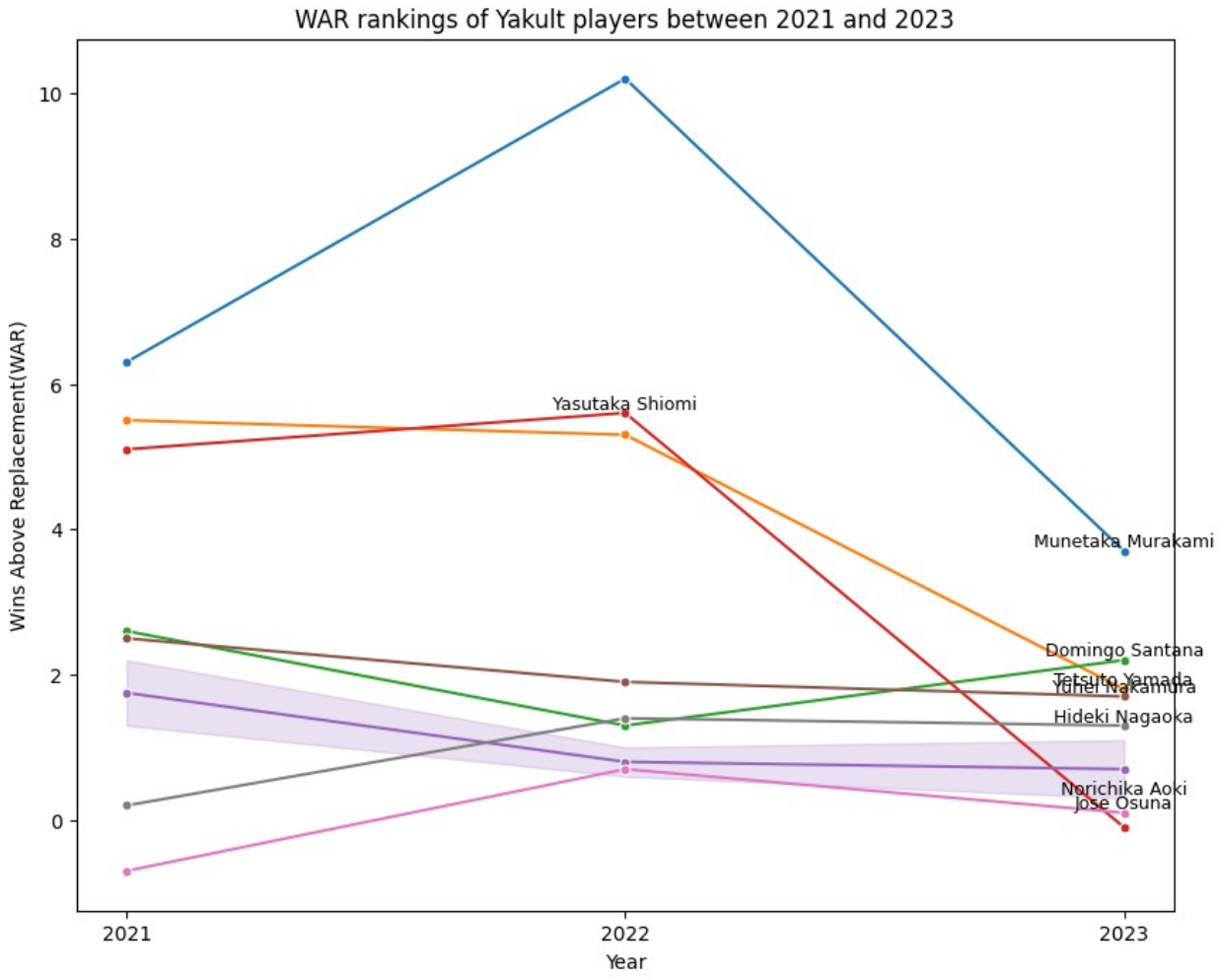


```
# Read the Excel file with Yakult players data from 2021 to 2023
Yakultplayer2021 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
Yakult2021.xlsx")
Yakultplayer2022 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
Yakult2022.xlsx")
Yakultplayer2023 = pd.read_excel(r"C:\Users\Mehdi\Desktop\NPB\
Yakult2023.xlsx")
# Concatenate the different variables to group data from 2021 to 2023
for Yakult teams in NPB
Yakultplayer = pd.concat([Yakultplayer2021,Yakultplayer2022,
Yakultplayer2023])
#Drop NaN values
Yakultplayer = Yakultplayer.dropna(axis= 1)
# Reset Index
Yakultplayer.reset_index(drop=True, inplace=True)
# Display the DataFrame
Yakultplayer.head()
```

```
              Player  Year  Lg   Tm Pos    G   PA    H  HR  RBI  SB
BB%  \
0  Munetaka Murakami  2021  CL  YAK  3B  143  615  139  39  112  12
0.172
1     Tetsuto Yamada  2021  CL  YAK  2B  137  581  134  34  101   4
0.131
2    Domingo Santana  2021  CL  YAK  RF  116  418  108  19   62   2
0.100
3    Yasutaka Shiomi  2021  CL  YAK  CF  140  534  132  14   59  21
0.090
4     Norichika Aoki  2021  CL  YAK  LF  122  501  115   9   56   0
0.086

      K%    AVG    OBP    SLG    OPS  BABIP  wRC+  WAR
0  0.216  0.278  0.408  0.566  0.974  0.302   167  6.3
1  0.172  0.272  0.370  0.515  0.885  0.273   144  5.5
2  0.246  0.290  0.366  0.511  0.877  0.355   142  2.6
3  0.292  0.278  0.357  0.441  0.798  0.388   122  5.1
4  0.088  0.258  0.335  0.383  0.719  0.268    99  2.2
```

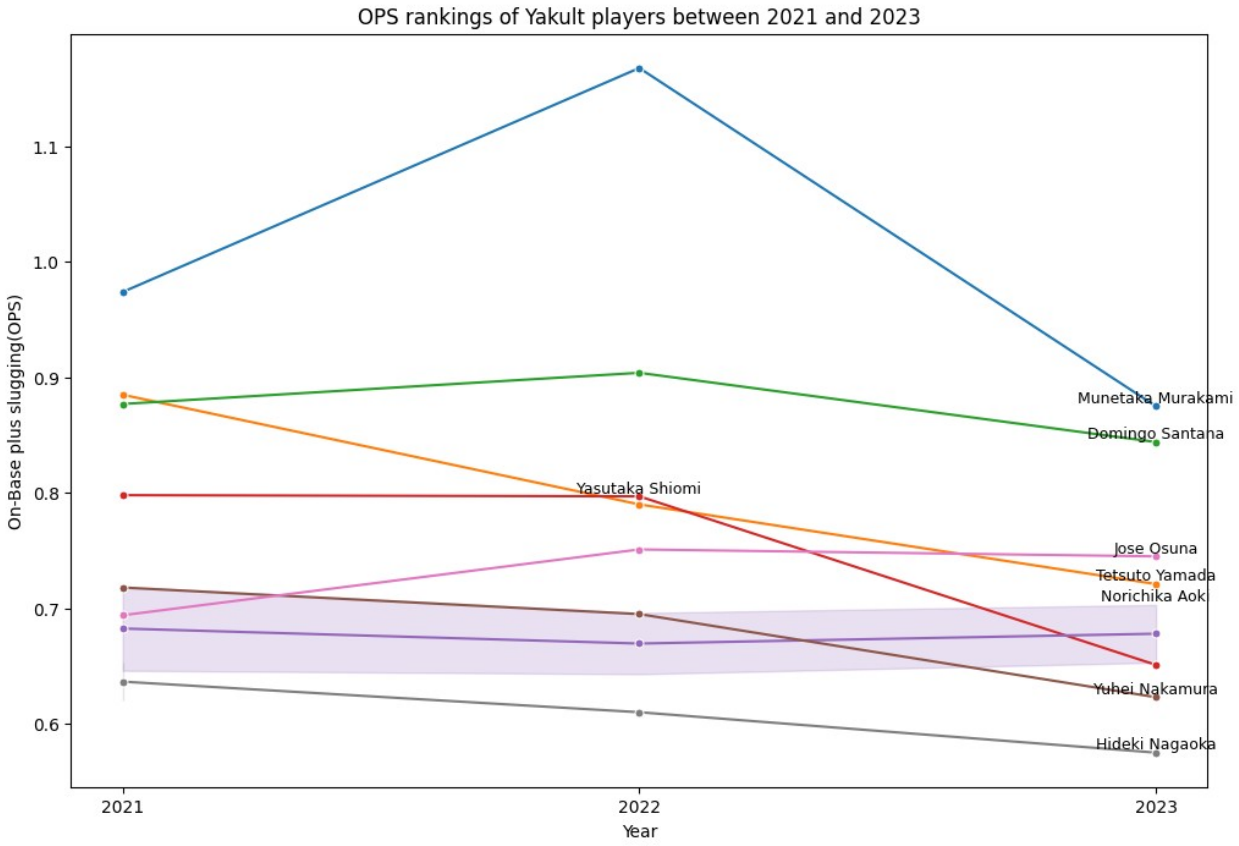
```
# Filter Yakult player's data from 2022 to 2023
Yakultplayer2022 = Yakultplayer[Yakultplayer["Year"] == 2022]
Shiomi = Yakultplayer2022[Yakultplayer2022["Player"] == 'Yasutaka
Shiomi']
Yakulttriple = Yakultplayer[Yakultplayer.duplicated(subset=['Player'],
keep=False)]
Yakulttriple = Yakulttriple[Yakulttriple['Year'] == 2023]

# We create a lineplot to see the evolution of the Yakult team's
performance between 2021 and 2023
plt.figure(figsize=(10, 8))
sns.lineplot(data=Yakultplayer, x="Year", y="WAR", hue="Pos",
```

```
marker="o", markersize=5, legend=False)
plt.xticks(range(2021, 2023 + 1, 1))
for index, row in Yakulttriple.iterrows():
    plt.text(row["Year"], row["WAR"], row["Player"],fontsize=9,
ha='center', va='bottom')
for index, row in Shiomi.iterrows():
    plt.text(row["Year"], row["WAR"], row["Player"],fontsize=9,
ha='center', va='bottom')

plt.xlabel('Year')
plt.ylabel('Wins Above Replacement(WAR)')
plt.title("WAR rankings of Yakult players between 2021 and 2023")
plt.show()
```




```
# We create a lineplot to see the evolution of the Yakult team's
performance between 2021 and 2023
plt.figure(figsize=(12, 8))
sns.lineplot(data=Yakultplayer, x="Year", y="OPS", hue="Pos",
marker="o", markersize=5, legend=False)
```

```
plt.xticks(range(2021, 2023 + 1, 1))
for index, row in Yakulttriple.iterrows():
    plt.text(row["Year"], row["OPS"], row["Player"],fontsize=9,
ha='center', va='bottom')
for index, row in Shiomi.iterrows():
    plt.text(row["Year"], row["OPS"], row["Player"],fontsize=9,
ha='center', va='bottom')

plt.xlabel('Year')
plt.ylabel('On-Base plus slugging(OPS)')
plt.title("OPS rankings of Yakult players between 2021 and 2023")
plt.show()
```



It's pretty hard to tell from these simple offensive figures, but we can point out two things. Firstly, the team was dependent on the attacking performances of its star player, Murakami, which is to be expected given his level. The team also lost a lot of performance due to Yasutaka Shiomi's absence through injury in 2023.

Apart from these two changes, the level of “WAR” remains constant among the 9 other regular players on the pitch. Given that there has been little change in terms of pure statistical performance among his teammates, it's understandable that Murakami's RBI score hasn't collapsed in 2023 compared to these other stats.