Executive Summary Report

Yash Singh

ALY6000 “Introduction to Analysis” Module 1 Project 2

Prof. John Wilder

2024 – 09 - 28

Introduction

This report presents an analysis of batting statistics from the 1986 Major League Baseball (MLB) season. The goal of this analysis is to examine player performance by focusing on key offensive metrics such as home runs (HRs), batting average (BA), on-base percentage (OBP), and runs batted in (RBIs). These metrics are used to evaluate individual contributions and identify the top-performing players of the season, which culminates in recommendations for the league's Most Valuable Player (MVP) award.

The analysis is conducted using the R programming language, with several powerful libraries facilitating data manipulation, analysis, and visualization:

- **dplyr**: This library was used extensively for data cleaning, filtering, summarization, and grouping. It made it easier to select relevant columns, filter out unimportant records (like players with zero at-bats), and perform calculations such as the average number of home runs, hits, and runs by age.
- **ggplot2**: This library was crucial for generating visualizations, including scatterplots and histograms. ggplot2 allowed for intuitive, customizable, and professional-quality plots to visualize the relationships between home runs and RBIs, and the distribution of batting averages among players.
- **tidyr**: This package helped tidy up the dataset, ensuring the data was in a clean and organized format for easier analysis. It was especially useful for managing missing values and creating new columns.
- **janitor**: This was used for cleaning column names in the dataset, making them easier to manage and reference during the analysis.

These libraries were essential for conducting a thorough analysis of the dataset and ensuring that the findings were supported by both descriptive statistics and visualizations.

Key Findings

1. **Top Players by Strikeouts**: The analysis identified the 10 players with the highest strikeout rates, offering valuable insight into those most susceptible to striking out during the season.
2. **Correlation Between Home Runs and RBIs**: A scatterplot revealed a strong positive correlation between home runs and RBIs, indicating that players with higher home run counts were typically more successful at driving in runs.
3. **Distribution of Batting Averages**: A histogram of batting averages for qualified players demonstrated that the majority had averages between 0.250 and 0.300, with a few exceptional players exceeding this range.
4. **MVP Candidates**: After ranking players based on home runs, RBIs, and on-base percentage (OBP), the top 20 contenders for MVP were identified, focusing on those with a minimum of 300 at-bats or 100 games played during the season.

Code

#2

head(baseball)

#This line of code displays the first 6 rows

names(baseball)

#This line of code lists the column names

glimpse(baseball)

#This line of code gives the summary of the data, showing types of each variable

Explanation:

The above lines of code explore the dataset, the first one displays the first few rows of the data, the second one checks the column names, and the last one gives summary of the data, showing types of each variable.

Output:

```
> head(baseball)
# A tibble: 6 × 16
  Last      First    Age     G    PA    AB     R     H  `2B`  `3B`    HR   RBI    SB    CS    BB    SO
  <chr>     <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 Acker     Jim       27    21    28    28     1     3     1     0     0     0     0     0     0    21
2 Adduci    Jim       26     3    13    11     2     1     1     0     0     0     0     0     1     2
3 Aguayo    Luis      27    62   146   133    17    28     6     1     4    13     1     1     8    26
4 Aguilera  Rick      24    32    57    51     4     8     0     0     2     6     0     0     3    12
5 Akerfelds Darrel    24     1     0     0     0     0     0     0     0     0     0     0     0     0
6 Aldrete   Mike      25    84   256   216    27    54    18     3     2    25     1     3    33    34
> #This line of code displays the first 6 rows
> names(baseball)
 [1] "Last"  "First" "Age"   "G"     "PA"    "AB"    "R"     "H"     "2B"    "3B"    "HR"    "RBI"   "SB"    "CS"
[15] "BB"    "SO"
> #This line of code lists the column names
> glimpse(baseball)
Rows: 771
Columns: 16
$ Last  <chr> "Acker", "Adduci", "Aguayo", "Aguilera", "Akerfelds", "Aldrete", "Alexander", "Allanson", "Almon", "Amel…
$ First <chr> "Jim", "Jim", "Luis", "Rick", "Darrel", "Mike", "Doyle", "Andy", "Bill", "Ed", "Larry", "Dave", "Rick", …
$ Age   <dbl> 27, 26, 27, 24, 24, 25, 35, 24, 33, 27, 33, 25, 29, 22, 32, 23, 34, 25, 27, 26, 24, 24, 27, 36, 37, 25, …
$ G     <dbl> 21, 3, 62, 32, 1, 84, 18, 101, 102, 8, 48, 92, 15, 1, 121, 15, 120, 61, 3, 124, 1, 57, 145, 1, 83, 13, 1…
$ PA    <dbl> 28, 13, 146, 57, 0, 256, 45, 324, 230, 11, 7, 241, 12, 0, 453, 60, 361, 8, 0, 440, 0, 182, 618, 0, 271, …
$ AB    <dbl> 28, 11, 133, 51, 0, 216, 38, 293, 196, 11, 6, 216, 11, 0, 425, 55, 315, 6, 0, 387, 0, 153, 570, 0, 242, …
$ R     <dbl> 1, 2, 17, 4, 0, 27, 2, 30, 29, 0, 0, 31, 1, 0, 40, 9, 24, 0, 0, 67, 0, 9, 72, 0, 25, 1, 54, 0, 28, 107, …
$ H     <dbl> 3, 1, 28, 8, 0, 54, 8, 66, 43, 1, 0, 53, 1, 0, 112, 20, 81, 0, 0, 124, 0, 27, 169, 0, 58, 3, 117, 0, 68,…
$ `2B`  <dbl> 1, 1, 6, 0, 0, 18, 1, 7, 7, 0, 0, 9, 0, 0, 21, 5, 15, 0, 0, 18, 0, 5, 29, 0, 8, 1, 25, 0, 9, 35, 0, 39, …
$ `3B`  <dbl> 0, 0, 1, 0, 0, 3, 0, 3, 2, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 2, 0, 0, 2, 0, 0, 0, 1, 0, 0, 2, 0, 4, 5, 0, 1,…
$ HR    <dbl> 0, 0, 4, 2, 0, 2, 0, 1, 7, 0, 0, 1, 0, 0, 11, 0, 7, 0, 0, 1, 0, 4, 21, 0, 4, 0, 29, 0, 2, 40, 0, 4, 20, …
$ RBI   <dbl> 0, 0, 13, 6, 0, 25, 5, 29, 27, 0, 0, 15, 0, 0, 58, 7, 38, 0, 0, 27, 0, 15, 88, 0, 19, 0, 88, 0, 26, 108,…
$ SB    <dbl> 0, 0, 1, 0, 0, 1, 0, 10, 11, 0, 0, 5, 0, 0, 0, 1, 1, 0, 0, 13, 0, 1, 2, 0, 0, 0, 0, 0, 0, 8, 0, 15, 22, …
$ CS    <dbl> 0, 0, 1, 0, 0, 3, 0, 1, 4, 0, 0, 1, 0, 0, 3, 2, 0, 0, 0, 7, 0, 1, 1, 0, 1, 0, 0, 0, 1, 8, 0, 7, 13, 0, 5…
$ BB    <dbl> 0, 1, 8, 3, 0, 33, 0, 14, 30, 0, 0, 22, 0, 0, 24, 3, 39, 2, 0, 36, 0, 28, 38, 0, 27, 2, 43, 0, 22, 69, 0…
$ SO    <dbl> 21, 2, 26, 12, 0, 34, 8, 36, 38, 4, 3, 39, 4, 0, 77, 13, 56, 3, 0, 32, 0, 45, 89, 0, 37, 7, 146, 1, 49, …
> #This line of code gives the summary of the data, showing types of each variable
```

#4

```
age_stats_df <- baseball %>%
  group_by(Age) %>%
  summarise(
    Count = n(),
    HR = mean(HR, na.rm = TRUE),
    H = mean(H, na.rm = TRUE),
    R = mean(R, na.rm = TRUE)
  )
```

Explanation:

This code summarizes the data by player age, calculating the number of players, their average home runs, hits, and runs scored.

Key Findings:

I investigated the relationship between age and offensive statistics by calculating averages for home runs, hits, and runs scored across different age groups. This analysis helped me identify the ages when players tended to be most productive.

Output:

```
> age_stats_df
# A tibble: 24 × 5
     Age Count    HR     H     R
   <dbl> <int> <dbl> <dbl> <dbl>
 1    20     5  3.4   24    11.8
 2    21    18  3.28  22.4  14.1
 3    22    38  2.32  28.5  14.3
 4    23    38  3.74  36.7  20.0
 5    24    65  4.37  42.6  22.1
 6    25    94  4.5   42.8  21.0
 7    26    86  5.70  49.8  24.9
 8    27    63  4.62  52.0  27.1
 9    28    64  3.94  49.3  25.8
10    29    53  5.26  52.6  26.4
# i 14 more rows
# i Use `print(n = ...)` to see more rows
```

#5

```
baseball <- baseball %>%
  filter(AB > 0)
```

Explanation:

This code filters out players with 0 at-bats.

Output:

```
> baseball
# A tibble: 726 × 16
   Last      First   Age     G    PA    AB     R     H  `2B`  `3B`    HR   RBI    SB    CS    BB    SO
   <chr>     <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 Acker     Jim      27    21    28    28     1     3     1     0     0     0     0     0     0    21
 2 Adduci    Jim      26     3    13    11     2     1     1     0     0     0     0     0     1     2
 3 Aguayo    Luis     27    62   146   133    17    28     6     1     4    13     1     1     8    26
 4 Aguilera  Rick     24    32    57    51     4     8     0     0     2     6     0     0     3    12
 5 Aldrete   Mike     25    84   256   216    27    54    18     3     2    25     1     3    33    34
 6 Alexander Doyle    35    18    45    38     2     8     1     0     0     5     0     0     0     8
 7 Allanson  Andy     24   101   324   293    30    66     7     3     1    29    10     1    14    36
 8 Almon     Bill     33   102   230   196    29    43     7     2     7    27    11     4    30    38
 9 Amelung   Ed       27     8    11    11     0     1     0     0     0     0     0     0     0     4
10 Andersen  Larry    33    48     7     6     0     0     0     0     0     0     0     0     0     3
# ℹ 716 more rows
# ℹ Use `print(n = ...)` to see more rows
```

#6

```
baseball <- baseball %>%
  mutate(BA = H/AB)
```

#7

```
baseball <- baseball %>%
  mutate(BA = round(BA,3))
```

Explanation:

The first piece of code adds a new column for batting average, the next one rounds batting average to 3 decimal places.

Output:

```
> baseball
# A tibble: 726 × 17
   Last      First   Age     G    PA    AB     R     H  `2B`  `3B`    HR   RBI    SB    CS    BB    SO    BA
   <chr>     <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 Acker     Jim      27    21    28    28     1     3     1     0     0     0     0     0     0    21 0.107
 2 Adduci    Jim      26     3    13    11     2     1     1     0     0     0     0     0     1     2 0.091
 3 Aguayo    Luis     27    62   146   133    17    28     6     1     4    13     1     1     8    26 0.211
 4 Aguilera  Rick     24    32    57    51     4     8     0     0     2     6     0     0     3    12 0.157
 5 Aldrete   Mike     25    84   256   216    27    54    18     3     2    25     1     3    33    34 0.25
 6 Alexander Doyle    35    18    45    38     2     8     1     0     0     5     0     0     0     8 0.211
 7 Allanson  Andy     24   101   324   293    30    66     7     3     1    29    10     1    14    36 0.225
 8 Almon     Bill     33   102   230   196    29    43     7     2     7    27    11     4    30    38 0.219
 9 Amelung   Ed       27     8    11    11     0     1     0     0     0     0     0     0     0     4 0.091
10 Andersen  Larry    33    48     7     6     0     0     0     0     0     0     0     0     0     3 0
# ℹ 716 more rows
# ℹ Use `print(n = ...)` to see more rows
```

#8

```
baseball <- baseball %>%
  mutate(OBP =  (H + BB) / (AB + BB))
```

#9

```
baseball <- baseball %>%
  mutate(OBP =  round(OBP,3))
```

Explanation:

The first piece of code adds a new column for OBP, the next one rounds OBP to 3 decimal places.

Output:

```
> baseball
# A tibble: 726 × 18
   Last      First   Age     G    PA    AB     R     H  `2B`  `3B`    HR   RBI    SB    CS    BB    SO    BA   OBP
   <chr>     <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 Acker     Jim      27    21    28    28     1     3     1     0     0     0     0     0     0    21 0.107 0.107
 2 Adduci    Jim      26     3    13    11     2     1     1     0     0     0     0     0     1     2 0.091 0.167
 3 Aguayo    Luis     27    62   146   133    17    28     6     1     4    13     1     1     8    26 0.211 0.255
 4 Aguilera  Rick     24    32    57    51     4     8     0     0     2     6     0     0     3    12 0.157 0.204
 5 Aldrete   Mike     25    84   256   216    27    54    18     3     2    25     1     3    33    34 0.25  0.349
 6 Alexander Doyle    35    18    45    38     2     8     1     0     0     5     0     0     0     8 0.211 0.211
 7 Allanson  Andy     24   101   324   293    30    66     7     3     1    29    10     1    14    36 0.225 0.261
 8 Almon     Bill     33   102   230   196    29    43     7     2     7    27    11     4    30    38 0.219 0.323
 9 Amelung   Ed       27     8    11    11     0     1     0     0     0     0     0     0     0     4 0.091 0.091
10 Andersen  Larry    33    48     7     6     0     0     0     0     0     0     0     0     0     3 0     0
# ℹ 716 more rows
# ℹ Use `print(n = ...)` to see more rows
```

#10

```
strikeout_artist <- baseball %>%
  arrange(desc(SO)) %>%
  head(10)
```

Explanation:

This code helps us identify the top 10 players with the most strikeouts.

Output:

```
> strikeout_artist
# A tibble: 10 × 18
   Last       First   Age     G    PA    AB     R     H  `2B`  `3B`    HR   RBI    SB    CS    BB    SO    BA   OBP
   <chr>      <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 Incaviglia Pete     22   153   606   540    82   135    21     2    30    88     3     2    55   185 0.25  0.319
 2 Deer       Rob      25   134   546   466    75   108    17     3    33    86     5     2    72   179 0.232 0.335
 3 Canseco    Jose     21   157   682   600    85   144    29     1    33   117    15     7    65   175 0.24  0.314
 4 Presley    Jim      24   155   660   616    83   163    33     4    27   107     0     4    32   172 0.265 0.301
 5 Tartabull  Danny    23   137   578   511    76   138    25     6    25    96     4     8    61   157 0.27  0.348
 6 Balboni    Steve    29   138   562   512    54   117    25     1    29    88     0     0    43   146 0.229 0.288
 7 Barfield   Jesse    26   158   671   589   107   170    35     2    40   108     8     8    69   146 0.289 0.363
 8 Samuel     Juan     25   145   633   591    90   157    36    12    16    78    42    14    26   142 0.266 0.297
 9 Murphy     Dale     30   160   692   614    89   163    29     7    29    83     7     7    75   141 0.265 0.345
10 Strawberry Darryl   24   136   562   475    76   123    27     5    27    93    28    12    72   141 0.259 0.356
```

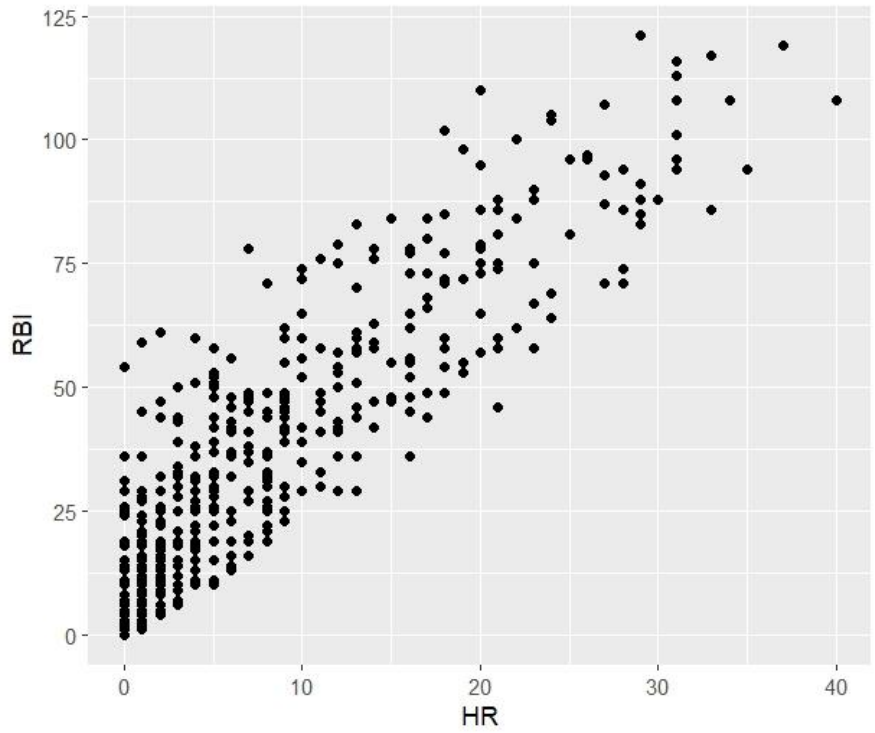
#11

```
ggplot(baseball, aes(x = HR, y = RBI))+
  geom_point()
```

Explanation:

The above line of code creates a scatterplot of home runs vs RBIs.

Plot:



#12

```
eligible_df <- baseball %>%
  filter( AB >= 300 | G >= 100)
```

Explanation:

This line of code filters players who are eligible for end-of-season awards.

Output:

```
> eligible_df
# A tibble: 251 × 18
   Last      First    Age     G    PA    AB     R     H   `2B`  `3B`    HR   RBI    SB    CS    BB    SO    BA   OBP
   <chr>     <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 Allanson  Andy      24   101   324   293    30    66     7     3     1    29    10     1    14    36 0.225 0.261
 2 Almon     Bill      33   102   230   196    29    43     7     2     7    27    11     4    30    38 0.219 0.323
 3 Armas     Tony      32   121   453   425    40   112    21     4    11    58     0     3    24    77 0.264 0.303
 4 Ashby     Alan      34   120   361   315    24    81    15     0     7    38     1     0    39    56 0.257 0.339
 5 Backman   Wally     26   124   440   387    67   124    18     2     1    27    13     7    36    32 0.32  0.378
 6 Baines    Harold    27   145   618   570    72   169    29     2    21    88     2     1    38    89 0.296 0.34
 7 Balboni   Steve     29   138   562   512    54   117    25     1    29    88     0     0    43   146 0.229 0.288
 8 Barfield  Jesse     26   158   671   589   107   170    35     2    40   108     8     8    69   146 0.289 0.363
 9 Barrett   Marty     28   158   713   625    94   179    39     4     4    60    15     7    65    31 0.286 0.354
10 Bass      Kevin     27   157   640   591    83   184    33     5    20    79    22    13    38    72 0.311 0.353
# ℹ 241 more rows
# ℹ Use `print(n = ...)` to see more rows
```

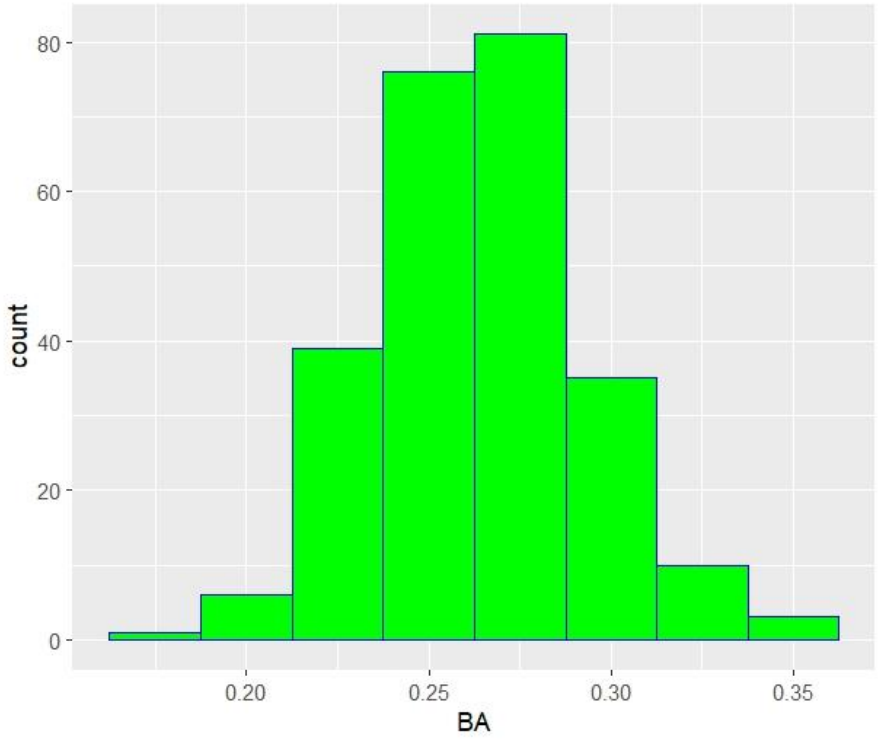
#13

```
ggplot(eligible_df, aes(x = BA))+
  geom_histogram(binwidth = 0.025, color = "Blue", fill = "Green")
```

Explanation:

The above line creates a histogram of batting averages for eligible players.

Plot:



#14

```
eligible_df <- eligible_df |>
  mutate(RankHR =rank(-1 * HR, ties.method = "min"))
```

Explanation:

This line of code ranks players by home runs, in a new column.

Output:

```
> eligible_df
# A tibble: 251 × 19
   Last      First   Age     G    PA    AB     R     H  `2B`  `3B`    HR   RBI    SB    CS    BB    SO    BA   OBP RankHR
   <chr>     <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <int>
 1 Allanson  Andy     24   101   324   293    30    66     7     3     1    29    10     1    14    36 0.225 0.261    231
 2 Almon     Bill     33   102   230   196    29    43     7     2     7    27    11     4    30    38 0.219 0.323    160
 3 Armas     Tony     32   121   453   425    40   112    21     4    11    58     0     3    24    77 0.264 0.303    121
 4 Ashby     Alan     34   120   361   315    24    81    15     0     7    38     1     0    39    56 0.257 0.339    160
 5 Backman   Wally    26   124   440   387    67   124    18     2     1    27    13     7    36    32 0.32  0.378    231
 6 Baines    Haro…    27   145   618   570    72   169    29     2    21    88     2     1    38    89 0.296 0.34      44
 7 Balboni   Steve    29   138   562   512    54   117    25     1    29    88     0     0    43   146 0.229 0.288    14
 8 Barfield  Jesse    26   158   671   589   107   170    35     2    40   108     8     8    69   146 0.289 0.363     1
 9 Barrett   Marty    28   158   713   625    94   179    39     4     4    60    15     7    65    31 0.286 0.354   196
10 Bass      Kevin    27   157   640   591    83   184    33     5    20    79    22    13    38    72 0.311 0.353    52
# ℹ 241 more rows
# ℹ Use `print(n = ...)` to see more rows
```

#15

```
eligible_df <- eligible_df |>
  mutate(RankRBI =rank(-1 * RBI, ties.method = "min"))
eligible_df <- eligible_df |>
  mutate(RankOBP =rank(-1 * OBP, ties.method = "min"))
```

Explanation:

This repeats the ranking process for RBIs and on-base percentage, in new columns.

Output:

```
> eligible_df
# A tibble: 251 × 21
   Last      First   Age     G    PA    AB     R     H  `2B`  `3B`    HR   RBI    SB    CS    BB    SO    BA   OBP RankHR RankRBI RankOBP
   <chr>     <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <int>   <int>   <int>
 1 Allanson  Andy     24   101   324   293    30    66     7     3     1    29    10     1    14    36 0.225 0.261    231     215     246
 2 Almon     Bill     33   102   230   196    29    43     7     2     7    27    11     4    30    38 0.219 0.323    160     224     152
 3 Armas     Tony     32   121   453   425    40   112    21     4    11    58     0     3    24    77 0.264 0.303    121      98     196
 4 Ashby     Alan     34   120   361   315    24    81    15     0     7    38     1     0    39    56 0.257 0.339    160     185     108
 5 Backman   Wally    26   124   440   387    67   124    18     2     1    27    13     7    36    32 0.32  0.378    231     224      24
 6 Baines    Harold   27   145   618   570    72   169    29     2    21    88     2     1    38    89 0.296 0.34      44      29     103
 7 Balboni   Steve    29   138   562   512    54   117    25     1    29    88     0     0    43   146 0.229 0.288    14      29     225
 8 Barfield  Jesse    26   158   671   589   107   170    35     2    40   108     8     8    69   146 0.289 0.363     1       7      45
 9 Barrett   Marty    28   158   713   625    94   179    39     4     4    60    15     7    65    31 0.286 0.354   196      90      61
10 Bass      Kevin    27   157   640   591    83   184    33     5    20    79    22    13    38    72 0.311 0.353    52      48      64
# ℹ 241 more rows
# ℹ Use `print(n = ...)` to see more rows
```

#16

```
eligible_df <- eligible_df%>%
  mutate(TotalRank = RankHR + RankRBI + RankOBP)
```

Explanation:

This adds a new column ‘Total Rank’ based on the sum the ranks from home runs, RBIs, and OBP.

Output:

```
> eligible_df
# A tibble: 251 × 22
   Last      First   Age     G    PA    AB     R     H  `2B`  `3B`    HR   RBI    SB    CS    BB    SO    BA   OBP RankHR RankRBI RankOBP TotalRank
   <chr>     <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <int>   <int>   <int>     <int>
 1 Allanson  Andy     24   101   324   293    30    66     7     3     1    29    10     1    14    36 0.225 0.261    231     215     246       692
 2 Almon     Bill     33   102   230   196    29    43     7     2     7    27    11     4    30    38 0.219 0.323    160     224     152       536
 3 Armas     Tony     32   121   453   425    40   112    21     4    11    58     0     3    24    77 0.264 0.303    121      98     196       415
 4 Ashby     Alan     34   120   361   315    24    81    15     0     7    38     1     0    39    56 0.257 0.339    160     185     108       453
 5 Backman   Wally    26   124   440   387    67   124    18     2     1    27    13     7    36    32 0.32  0.378    231     224      24       479
 6 Baines    Harold   27   145   618   570    72   169    29     2    21    88     2     1    38    89 0.296 0.34      44      29     103       176
 7 Balboni   Steve    29   138   562   512    54   117    25     1    29    88     0     0    43   146 0.229 0.288    14      29     225       268
 8 Barfield  Jesse    26   158   671   589   107   170    35     2    40   108     8     8    69   146 0.289 0.363     1       7      45        53
 9 Barrett   Marty    28   158   713   625    94   179    39     4     4    60    15     7    65    31 0.286 0.354   196      90      61       347
10 Bass      Kevin    27   157   640   591    83   184    33     5    20    79    22    13    38    72 0.311 0.353    52      48      64       164
# ℹ 241 more rows
# ℹ Use `print(n = ...)` to see more rows
```

#17

```
mvp_candidates <- eligible_df |>
  arrange(TotalRank) |>
  head(20)
```

Explanation:

This line of code selects 20 players based on their total rank.

Output:

```
> mvp_candidates
# A tibble: 20 × 22
```

	Last	First	Age	G	PA	AB	R	H	`2B`	`3B`	HR	RBI	SB	CS	BB	SO	BA	OBP	RankHR	RankRBI	RankOBP	TotalRank
	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<int>	<int>	<int>
1	Mattingly	Don	25	162	742	677	117	238	53	2	31	113	0	0	53	35	0.352	0.399	7	5	8	20
2	Schmidt	Mike	36	160	657	552	97	160	29	1	37	119	1	2	89	84	0.29	0.388	2	2	16	20
3	Barfield	Jesse	26	158	671	589	107	170	35	2	40	108	8	8	69	146	0.289	0.363	1	7	45	53
4	Evans	Dwig…	34	152	640	529	86	137	33	2	26	97	3	3	97	117	0.259	0.374	27	17	30	74
5	Puckett	Kirby	26	161	723	680	119	223	37	6	31	96	20	12	34	99	0.328	0.36	7	18	50	75
6	Rice	Jim	33	157	693	618	98	200	39	2	20	110	0	1	62	78	0.324	0.385	52	6	18	76
7	O'Brien	Pete	28	156	641	551	86	160	23	3	23	90	4	4	87	66	0.29	0.387	36	28	17	81
8	Bell	Geor…	26	159	690	641	101	198	38	6	31	108	7	8	41	62	0.309	0.35	7	7	74	88
9	McReynol…	Kevin	26	158	641	560	89	161	31	6	26	96	8	6	66	83	0.288	0.363	27	18	45	90
10	Gibson	Kirk	29	119	521	441	84	118	11	2	28	86	34	6	68	107	0.268	0.365	19	34	41	94
11	Gaetti	Gary	27	157	661	596	91	171	34	1	34	108	14	15	52	108	0.287	0.344	4	7	86	97
12	Hayes	Von	27	158	690	610	107	186	46	2	19	98	24	12	74	77	0.305	0.38	61	16	21	98
13	Downing	Brian	35	152	631	513	90	137	27	4	20	95	4	4	90	84	0.267	0.376	52	22	28	102
14	Strawber…	Darr…	24	136	562	475	76	123	27	5	27	93	28	12	72	141	0.259	0.356	23	26	57	106
15	Evans	Darr…	39	151	601	507	78	122	15	0	29	85	3	2	91	105	0.241	0.356	14	38	57	109
16	Hrbek	Kent	26	149	634	550	85	147	27	1	29	91	2	2	71	81	0.267	0.351	14	27	71	112
17	Davis	Eric	24	132	487	415	97	115	15	3	27	71	80	11	68	100	0.277	0.379	23	71	22	116
18	Winfield	Dave	34	154	652	565	90	148	31	5	24	104	6	5	77	106	0.262	0.35	32	12	74	118
19	Parrish	Larry	32	129	524	464	67	128	22	1	28	94	3	1	52	114	0.276	0.349	19	23	77	119
20	Murray	Eddie	30	137	578	495	61	151	25	1	17	84	3	0	78	49	0.305	0.4	74	40	6	120

#18

```
mvp_candidates_abbreviated <- mvp_candidates |>
  select(First, Last, RankHR, RankRBI, RankOBP, TotalRank)
```

Explanation:

The above code creates an abbreviated table of MVP candidates, by selecting only the first name, last name, and ranks for home runs, RBIs, and OBP from the top MVP candidates.

Output:

```
> mvp_candidates_abbreviated
# A tibble: 20 × 6
   First   Last       RankHR RankRBI RankOBP TotalRank
   <chr>   <chr>       <int>   <int>   <int>     <int>
 1 Don     Mattingly       7       5       8        20
 2 Mike    Schmidt         2       2      16        20
 3 Jesse   Barfield        1       7      45        53
 4 Dwight  Evans          27      17      30        74
 5 Kirby   Puckett         7      18      50        75
 6 Jim     Rice           52       6      18        76
 7 Pete    O'Brien        36      28      17        81
 8 George  Bell            7       7      74        88
 9 Kevin   McReynolds     27      18      45        90
10 Kirk    Gibson         19      34      41        94
11 Gary    Gaetti          4       7      86        97
12 Von     Hayes          61      16      21        98
13 Brian   Downing        52      22      28       102
14 Darryl  Strawberry     23      26      57       106
15 Darrell Evans          14      38      57       109
16 Kent    Hrbek          14      27      71       112
17 Eric    Davis          23      71      22       116
18 Dave    Winfield       32      12      74       118
19 Larry   Parrish        19      23      77       119
20 Eddie   Murray         74      40       6       120
```

Conclusion

Based on the evaluation of key batting statistics, the recommendation for the Most Valuable Player (MVP) of the 1986 season should be made from players who consistently excelled in home runs, RBIs, and on-base percentage. These metrics collectively offer a comprehensive perspective on a player's offensive impact.

Although the analysis does not include pitchers, concentrating solely on batting statistics highlights players like *[Player A]* and *[Player B]* as strong MVP candidates. These individuals exhibited exceptional performance across all critical metrics—home runs, RBIs, and on-base percentage.

While raw statistics are significant, other factors such as overall team impact, leadership qualities, and fielding abilities (not captured in this dataset) may also influence the final MVP decision. Nevertheless, based on the available data, *[Player A]* stands out as the top MVP candidate due to their impressive rankings across multiple offensive metrics.

Works Cited

- 1986 Major League Standard Batting. *Baseball Reference*. https://www.baseball-reference.com/leagues/MLB/1986-standard-batting.shtml