

Extract - CSV (Pro Football Reference - stats) | HTML (USA Today rankings)

Transform – Web scrape → Cleaned up CSV → Jupyter Notebook → List of python dictionaries

Load – Load list of python dictionaries into Mongo DB using ‘import pymongo’ in Jupyter Notebook

Data Cleanup & Analysis:

Data Sources –

Pro Football Reference

We used csv files exported from the Pro Football Reference website. You can use the search tool below to generate the quarterback stat charts. To get the tables, you select a stat to analyze, specify career or season stats, choose all available data to date (2019), then click the green “Go!” button. Once the table is generated you can click the ‘Share & more’ toolbar to export the chart to a csv file that we can manipulate.

See leaders through past seasons

Passing Yards

Career

2019

Go!

Leaders + indicates Hall of Famer Share & more ▼ Glossary

Note that long tables like this one may be slow to sort and move on some devices

Rank	Player	Yds	Years	Tm
1	Drew Brees	75,733	2001-2019	2TM
2	Tom Brady	73,266	2000-2019	nwe
3	Peyton Manning	71,940	1998-2015	2TM

USA Today

In order to get our list of all-time quarterback rankings we used BeautifulSoup to scrape the HTML code from the USA Today sports blog ‘FOR THE WIN’ to get a list of names to then compile data for. We needed their list in order to have a name to perform lookup functions in excel (index/match) and retrieve stats. We also needed their ranking data associated with the quarterback names.

FORTHEWIN

To build this list, I used [Pro Football Reference's "approximate value" rankings](#) to get a baseline, and then made some individual calls off of

To start, I'm going to just show you the rankings of 100-20, because the article is already too long and I'm not sure anyone wants to read multi words, let alone sentences, on the greatness of Brian Sipe. (Just kidding you're wonderful, Brian Sipe. Go Brian Sipe.)

After that, we'll dive into the top 20 in more detail.

TO THE LIST!

100. Jeff George
99. Lynn Dickey
98. Jeff Blake
97. Matt Schaub
96. Babe Parilli

Type of Transformation:

For Transformation we took our csv exports from Pro Football Reference and dumped them into one master Excel file (12 files total – CSV Excel tables). In the master Excel file, we used built-in Excel functions such as “text-to-columns”, “find and replace”, “concatenate”, and “index(match())” to clean up the data and generate a final table that we could read in Jupyter notebook to further our transformation.

For example, in the web scrape from the USA Today article, ranking and quarterback name were combined in one string/column, so we separated them using space as the delimiter. We then put them in their own separate column and removed the period from the ranking string (using “find and replace”). Finally, we concatenated the first and last name back together into one column.

We did this so that the name formatting would match that of the csv tables. That way when we pulled all the statistical data across all the csv tables, the name match would allow us to retrieve the statistical data we needed. We used the index/match function in excel to add statistical data from the csv exports to our master list generated from the web scrape. We then copy and pasted that data into a separate csv file so that the file could be easily read using Python and Jupyter Notebook.

Once we appended each excel column to a python list in Jupyter notebook we used a for loop to convert a zipped list of tuples into a list of python dictionaries with key value pairs (‘stat type’ : ‘data’). We then imported our list of dictionaries into MongoDB so that you could query relevant quarterback data using their full name. We chose a non-relational database to store our data.

The final tables or collections:

```
> use nfl_qbs
switched to db nfl_qbs
> db.collection.find({"Name":"Tom Brady"})
{ "_id" : ObjectId("5dd6d7036764ecec3f9098"), "Rank" : 1, "Name" : "Tom Brady", "Passing_Yds" : 26,977, "Completion_Percentage" : "64.03%", "Passer_Ratings" : "97.30", "YPG" : "26.977" }
```

MongoDB Compass Community - localhost:27017/nfl_qbs.collection

Connect View Collection Help

My Cluster

14 DBS 12 COLLECTIONS

HOST: localhost:27017

CLUSTER: Standalone

EDITION: MongoDB 4.2.1 Community

Filter your data

admin

classDB

config

craigslist_db

fruits_db

local

nfl

nfl_qbs

collection

nfl_qbs.collection Documents

Documents Aggregations Explain Plan Indexes

FILTER

INSERT DOCUMENT VIEW LIST TABLE

	_id ObjectId	Rank Int32	Name String	Passing_Yds String
1	5dd6d7036764ecec3f9035	100	"Jeff George"	" 27,602 "
2	5dd6d7036764ecec3f9036	99	"Lynn Dickey"	" 23,322 "
3	5dd6d7036764ecec3f9037	98	"Jeff Blake"	" 21,711 "
4	5dd6d7036764ecec3f9038	97	"Matt Schaub"	" 25,412 "
5	5dd6d7036764ecec3f9039	96	"Babe Parilli"	" 22,681 "
6	5dd6d7036764ecec3f903a	95	"Andrew Luck"	" 23,671 "
7	5dd6d7036764ecec3f903b	94	"Bobby Hebert"	" 21,683 "
8	5dd6d7036764ecec3f903c	93	"Bernie Kosar"	" 23,301 "
9	5dd6d7036764ecec3f903d	92	"Neil Lomax"	" 22,771 "

```
> { "_id": ObjectId("5dd6d7036764ecec3f9035"),
  "Rank": 100,
  "Name": "Jeff George",
  "Passing_Yds": " 27,602 ",
  "TDs": "154",
  "Int": "113",
  "TD_INT_Ratio": "1.36",
  "Completions": " 2,298 ",
  "Attempts": " 3,967 ",
  "Completion_Percentage": "57.93%",
  "Passer_Ratings": "80.40",
  "YPG": "210.70",
  "Game_Played": "131",
  "Seasons_Played": "11",
  "Number_of_Teams": "5TM" }
```