ETL Project

# University of Phoenix Data Science Bootcamp

# 3/6/2019

Sean Roth

Richard Yarbough

Bryan Paynich

**ETL**

ETL is an abbreviation for Extract, Transform, and Load. These 3 functions are the basis for taking unorganized, various data and normalizing it into one useable format. The extraction function is used for pulling data from various sources that maybe not in a standard common format. Once the data is brought together it is organized or Transformed into a consistent workable format. The Load procedure is then used to push the data into a database that can be accessed and used for analysis and visualization.

Extract/Transform/Load

## NBA player/stadium mashup

For this project we found 3 types of data:

* NBA player data – csv file
* NBA stadium locations by year – html page
* NBA stadium name, location, and latitude/longitude coordinates – combination of google api queries

After finding the data we combined it into one consistent dataset. This data included:

* Player information (position, age, player performance data, team, year(s) played
* NBA arena (Arena name, location, address, latitude/longitude coordinates)
* NBA arena historical Data (Arena name by year for each team)

## NBA player Data

**Extract:**

This data was an extracted csv from a NBA website. <https://www.kaggle.com/drgilermo/nba-players-stats#player_data.csv> Kaggle.com NBA Players stats since 1950 3000+ Players over 60+ Seasons, and 50+ features per player

**Transform and Load:**

The transformation transformed and loaded into Pandas. The data was then loaded into a mysql table using pymysql.

## NBA stadium Data

To find the NBA team stadiums data from Wikipedia at the link below. The data set contain NBA team name and acronyms. The task with this data was to extract latitude and longitude points, so the points could be run through an API to grab the stadium locations. With this data set we also used the team acronyms to help link the table together in SQL. The following tasks were performed on that information to extract, transform, and load the data.

**Extract:**

To extract the data a simple right click Copy was used to grab the initial info from the site. Then pasted into an excel sheet and formatted as a csv. Implemented the use of jupyter notebook in pandas to read in the csv and creating list. https://en.wikipedia.org/wiki/Wikipedia:WikiProject\_National\_Basketball\_Association/National\_Basketball\_Association\_team\_abbreviations names and their acronyms from Wikipedia.

**Transform:**

The variable list was transformed into a single data frame. However, the city, state, and zip needed to be separated into separate columns from the address column. To do this a split delimiter was done to extract those into separate columns in the data frame. Last thing done was to add the team acronyms to create a new data frame. With the complete data frame complete it was then loaded into a SQL data base.

**Load:**

To start loading in the team names to look for the latitude and longitude information. A for loop in conjunction with google geocode to loop through the data use the team name as the search aspect for geocode. Team name, address, latitude, and longitude were stored in variable Empty list.

## NBA arena historical Data

The third task for retrieving data was regarding the historical arena venues each NBA team played in from 1950 to the present time.

Extracted the data from the link sourced below:

<https://nbahoopsonline.com/History/Leagues/NBA/Arenas.html>  nbahoopsonline.com NBA Arenas NBA Arena History

Method used for extracting the data, scraped via html coding and implemented using pandas, and converted to a csv file. Once the CSV file was created, converted to a dataframe then used that to transform the data to display what we needed:

For example: arena\_list = "Arena\_list.csv"

arena\_list\_pd = pd.read\_csv(arena\_list)

arena\_list\_pd.head()

Then proceeded to transfer the dataframe to show all the relevant information needed to use for our flask and using SQL Alchemy, combine that within a data table made: Year, Venue, Team, Abbreviation -

count\_year= []

count\_venue= []

count\_team= []

count\_abv = []

for index, row in arena\_list\_pd.iterrows():

Start = row['StartYear']

End = row['EndYear']

Venue = row['Venue']

Teams = row['Team']

Abv = row['ABV']

while int(Start) < int(End):

#print(Start, Venue, Teams)

count\_year.append(Start)

count\_venue.append(Venue)

count\_team.append(Teams)

count\_abv.append(Abv)

Start = Start+1

results\_list= list(zip(count\_year, count\_venue, count\_team, count\_abv))

results\_df= pd.DataFrame(results\_list)

results\_df.head()

results\_df = results\_df.rename(columns={results\_df.columns[0]: "year", \

results\_df.columns[1]: "venue", \

results\_df.columns[2]: "team", \

results\_df.columns[3]: "abv" })

results\_df.head()

engine = create\_engine("mysql+pymysql://{user}:{pw}@localhost/{db}"

.format(user="root",

pw="\*\*\*\*\*\*",

db="season\_stats\_db"))

results\_df.to\_sql(con=engine, name='arena\_venue\_year\_table', if\_exists='replace')

Loaded this data with SQL to pair alongside the NBA player data extracted, so all can be incorporated into the Python Flask.

Python Flask

# Basic functionality:

**The user selects the desired Team and Year drop down fields and presses on the submit button. The page then refreshes displaying team player and arena information on the page.**

The development of the flask app started with each of the individual MySQL tables. These three tables were then combined with and INNER JOIN query.

SELECT s.Y, s.Player, s.Pos, s.Age, s.Tm, s.G, s.GS, s.MP, s.PER, s.TS, s.PAr, s.FTr, s.STL, s.BLK, s.TOV, s.PF, s.PTS, a.teamname, a.street, a.city, a.zip, a.latitude, a.longitude, x.venue FROM season\_stats\_table as s

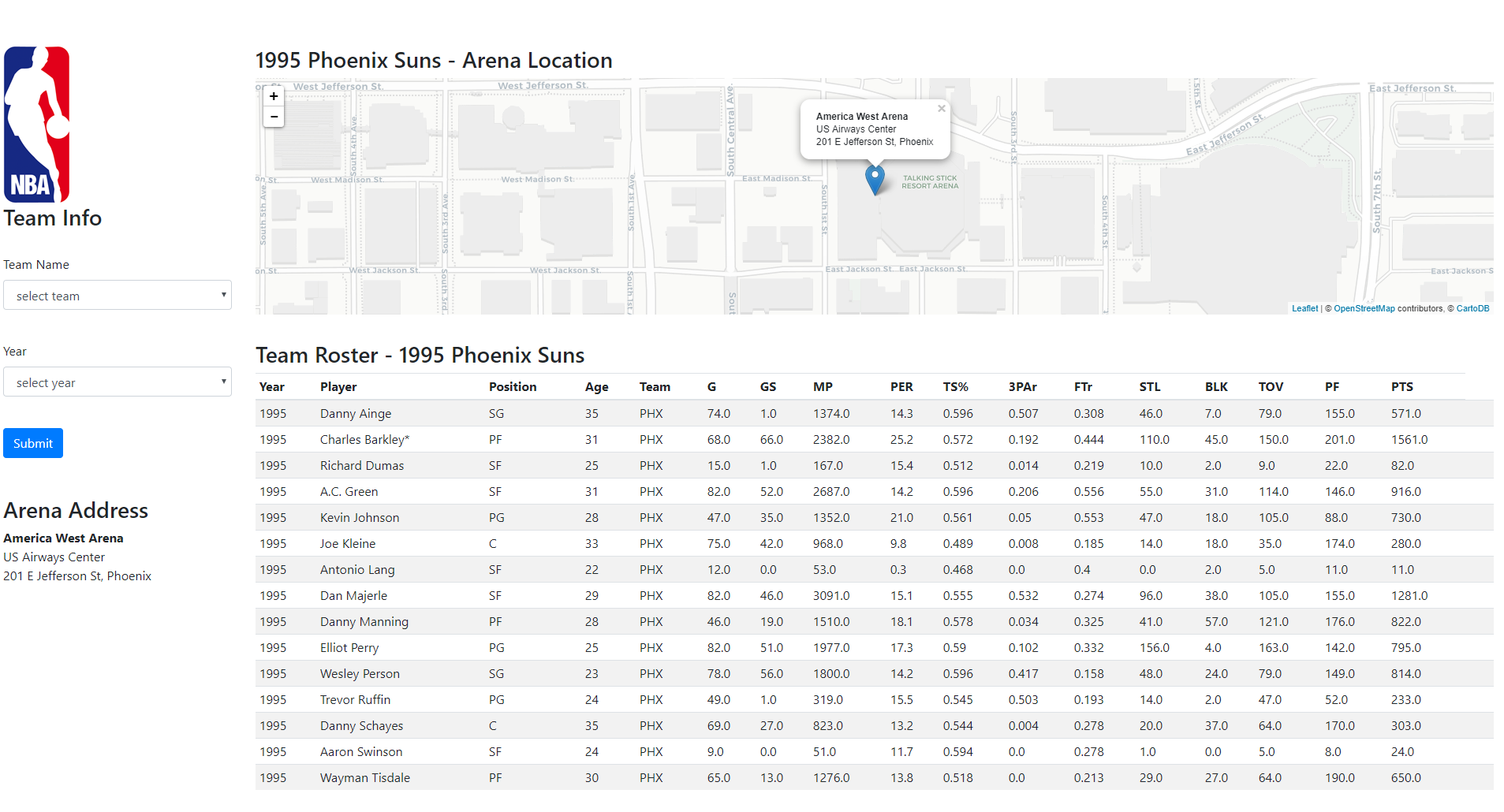
INNER JOIN address\_api\_table as a ON s.Tm = a.teamacronym

INNER JOIN arena\_venue\_year\_table as x on s.Tm = x.abv AND s.Y = x.year

This query joined all three tables together with the Team abbreviation being the main connector. This allows the user to select the team name and the year for the query on the website. Once the user selects the options through the drop down two things happen:

* The map is displayed with the latitude and longitude coordinate displayed as a marker on the stadium
* The Arena name and address are displayed in the marker
* The Address label is updated with the arena name.
* The roster is updated to reflect the players for the year and team selected.

Team drop down is populated by the Tm field from the season\_stats table which has been reduced to a single occurrence of the team name. The Year field is populated by a range of the beginning year and ending year of the season\_stats table.



Database information

# Table descriptions:

**Address\_api\_table – This table was created by using the google app to scrape the web for addresses, and coordinates of the basketball arenas.**

CREATE TABLE `address\_api\_table` (

`index` bigint(20) DEFAULT NULL,

`teamacronym` text,

`teamname` text,

`street` text,

`city` text,

`zip` text,

`latitude` double DEFAULT NULL,

`longitude` double DEFAULT NULL,

KEY `ix\_address\_api\_table\_index` (`index`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1

**Arena\_venue\_year\_table – This table was scraped from the team history site. We created a python script to increment through the history and create an entry for each year and arena.**

CREATE TABLE `arena\_venue\_year\_table` (

`index` bigint(20) DEFAULT NULL,

`year` bigint(20) DEFAULT NULL,

`venue` text,

`team` text,

`abv` text,

KEY `ix\_arena\_venue\_year\_table\_index` (`index`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1

**Season\_stats\_table – This table was extracted from the NBA season\_stats web site. The team imported the csv into a jupyter notebook. We then inserted the data into a mysql table.**

CREATE TABLE `season\_stats\_table` (

`index` bigint(20) DEFAULT NULL,

`Unnamed: 0` bigint(20) DEFAULT NULL,

`Y` int(11) DEFAULT NULL,

`Player` varchar(50) DEFAULT NULL,

`Pos` varchar(20) DEFAULT NULL,

`Age` int(11) DEFAULT NULL,

`Tm` varchar(50) DEFAULT NULL,

`G` double DEFAULT NULL,

`GS` double DEFAULT NULL,

`MP` double DEFAULT NULL,

`PER` double DEFAULT NULL,

`TS` double DEFAULT NULL,

`PAr` double DEFAULT NULL,

`FTr` double DEFAULT NULL,

`STL` double DEFAULT NULL,

`BLK` double DEFAULT NULL,

`TOV` double DEFAULT NULL,

`PF` double DEFAULT NULL,

`PTS` double DEFAULT NULL,

KEY `ix\_season\_stats\_table\_index` (`index`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1