We, all three project partners have been big fan of the movies and we often had discussions about the movie awards and the Oscars. So for this project we decided to create a small database about the movies which won an Oscar since the year 1960. Along with the name of the movie, we tabulated, the director of the movie, production studio and the rating as well.

**Extract:**

Data for this ETL project was extracted from two sources.

* + <https://www.filmsite.org/bestpics4.html>
  + <http://www.omdbapi.com/>

**Transform:**

Data scraped from [www.filmsite.org](http://www.filmsite.org) website and obtained in the table format.

(See Schema below)

{“Film Year”: “int”  
, “Best Picture Academy Award Winners”: “string”  
, “Director (\* Did not also win Best Director)”: “string”  
, “Title Screen and Studio or Production Company”: “string”  
, “NOT Nominated for Best Picture”: “string”  
, “Should Have Won”: “string”  
}

From the data scrapping we obtained various different tables, we carefully analyzed tables and selected right table which has the useful information for the project such as movie year, movie name which won the Oscars that year, Director name and so on.The selected table then converted into DataFrame. From the DataFrame unwanted columns were eliminated and columns were renamed using more appropriate names.Once the “Movies” table is finalized, all the names of the movies were captured in a list. Using the names of the movies in the list, captured the all the other information about the movie by making JSON calls to the [www.omdbapi.com](http://www.omdbapi.com) website.

[{'Title': “string”,  
 'Year': int,  
 'Rated': “string”,  
 'Released': “string”,  
 'Runtime': 'string',  
 'Genre': 'string',  
 'Director': 'string',  
 'Writer': 'string',  
 'Actors': 'string',  
 'Plot': 'string',  
 'Language': 'string',  
 'Country': 'string',  
 'Awards': 'string',  
 'Poster': 'string',  
 'Ratings': [{'Source': 'string', 'Value': “float”},  
  {'Source': “string”, 'Value': 'string'},  
  {'Source': 'string', 'Value': 'string'}],

 'Metascore': 'int',  
 'imdbRating': 'float',  
 'imdbVotes': 'int',  
 'imdbID': 'string',  
 'Type': 'string',  
 'DVD': 'string',  
 'BoxOffice': 'string',  
 'Production': 'string',  
 'Website': 'string',  
 'Response': “boolean”},

Data obtained using JSON then collected in separate lists and the second table is created from the values in those lists.

**Load:**

Once, the appropriate data was fetched, cleaned and properly tabulated as DataFrames, data was then transferred to the Postgres SQL. To transfer data to Postgres SQL from jupyter notebook, we created the empty tables in Postgres, and in jupyter notebook, we created the engine to for seamless communication between SQL and python. Once, the engine was created, we created queries in jupyter to transfer DataFrames to the appropriate tables in the SQL database. At last, we created a query in jupyter to merge both tables.