Extraction, Transformation, and Load Technical Report

***Team ETheL: Ashley Fay, Neil Patel, Caroline Miller***

Books Adapted for Movies

**TABLE OF CONTENTS**

1. Introduction 3
   1. Summary 3
   2. Scope 3
   3. Technologies and resource contributions 3
   4. Definitions, Acronyms and Abbreviations 3
2. ETL Details 4
   1. Data Import/Extract Sources and Method 4
   2. Data Acquisition 4
   3. Data Transform 4
   4. Data Integrity 4
   5. Data Refresh Frequency 4
   6. Data Security 4
   7. Data Loading and Availability 5
3. Data Quality 6

|  |  |
| --- | --- |
| **1.** | **INTRODUCTION** |

# 1.1 Summary

The objective of the ETL was to determine a dataset consisting solely of books that have been converted to movies and the information that both the movie and book consist of including movie rating, book publisher, release year, and more.

# 1.2 Scope

The objective we are accomplishing is to understand which books should be made into movies by genre and using ratings as a guideline. The dataset we created is to have all of the books converted into movies in one place to make analysis easier.

# 1.3 Technologies and resource contributions

We all collaborated together to search the data, clean the data, load into MySQL, and write about our process. For this project, we used python’s Pandas, JSON, and NumPy packages to clean and MySQL to load the dataset into a new environment.

# 1.4 Definitions, Acronyms and Abbreviations

* ETL: Extract, Transform and Load.
* Pandas: Python package for manipulating data frames.
* NumPy: Python package for scientific computing on arrays.
* MySQL Workbench: editor that supports the MySQL language, which is used for loading and querying datasets.
* ASIN: Amazon Standard Identification Number.
* JSON: JavaScript Object Notation.

|  |  |
| --- | --- |
| **2.** | **ETL DETAILS** |

*.*

# 2.1 Data Import/Extract Sources and Method

Both data sources were originally from Kaggle. The first one is the book data collected from Amazon. This dataset was created and updated between January 1, 2017 and June 29, 2018. There are over 61,000 books on the list and include details such as ASIN, whether it is a book or kindle version, if it is paperback or hardcover, title. There was a second Amazon book dataset which provided extra details about the books including title, publisher, and author. The movie dataset is from Netflix and includes data on 45,000 movies. Details in this dataset include release year and title of movies. There was a second movie dataset not from Netflix used that included more details about each movie include rating, genre, release date, revenue, budget, overview, title, and tagline. This one gave more information and was used to fill in missing gaps we would need for later analysis. We used the title” field in each dataset to do our merges. Kaggle requires a Kaggle account to use the data there. We all have Kaggle accounts, so this data was accessible to us. We did not need any extra permissions to access the datasets. To get the data into python, we needed to make sure the files were saved as CSV and loaded in with Pandas.

URLs to the data:

Netflix: <https://www.kaggle.com/netflix-inc/netflix-prize-data>

<https://www.kaggle.com/danofer/movies-data-clean/data>

Amazon: <https://www.kaggle.com/ucffool/amazon-sales-rank-data-for-print-and-kindle-books#amazon_com_extras.csv>

# 2.2 Data Acquisition

For this project, important elements required were names of both movies and books and additional information such as ratings and genres. The data is static so to get more up-to-date data, there would have to be another data pull later. In order to update our data appropriately, we would have to first observe how often the author of our dataset provides updates. If that data is not updated appropriately, we would then finding another source with the same details but more updated would have to be found.

# 2.3 Data Transform

When the datasets were loaded in, the initial cleaning included eliminating unnecessary columns such as “Count” for the Netflix data, renaming the headers, converting headers to be lowercased, and filling in blank cells with NaN. The larger movie dataset had multiple superfluous columns dropped because those columns were irrelevant to our analysis.

We used an inner joined on “title” using the larger movie dataset and the larger Amazon dataset. If there was a match in title name, it became part of the new data frame. Then, duplicate titles were dropped from the merged set to reduce redundancy.

When we first began working with the Netflix data, we noticed that cells of columns had nested string dictionaries. We needed to unravel these dictionaries in order to make each dictionary genre its own column. We first used the JSON package to remove the strings out of the dictionaries. Then, we did a manipulation on the column that contained the multiple nested dictionaries by running loops through each of them to populate newly defined column names. Once all of the dictionary values were in their own column, the new columns were merged back into the overall merged data frame. Once that was all complete, the file was written as a CSV.

Once this was completed, our master data frame was loaded into MySQL Workbench in a new schema. There were issues initially loading the data which required additional cleaning of the master dataset such as dropping the index, adding in NaN values in the new genre columns, and ensuring the encoding was loaded in correctly. The data was then re-written and properly loaded into MySQL Workbench. We checked this by doing the SELECT \* FROM function to see how the data looked.

# 2.4 Data Integrity

We found the data is overall reliable, especially the Netflix data because its initial purpose was used for a Netflix competition for people to create the best movie suggestion algorithm. The data sources were all reliable in terms of the information provided. The second movie dataset originally pulled information using IMDb, which is the premier source for everything relating to movies and TV shows. The Amazon datasets are reliable because it is collected daily from NovelRank.com. However, the data has not been updated on Kaggle since June 2018. The author stated in the Kaggle description that they are mindful about connecting the books from Amazon to NovelRank and mentions the limitations in the dataset Kaggle page including that the dataset does not take into account multiple purchases of the same book and how that changes the rating and that there is a slight lag from when someone purchases a book on Amazon and when NovelRank collects the data.

# 2.5 Data Refresh Frequency

For the purpose of this project, it is not necessary to constantly update the datasets. The material is not something as urgent such as weather or news data. Updating this yearly will be enough to get new perspectives on books turned into movies recently.

# 2.6 Data Security

This data does not require any additional data security because it is already from a relatively open source. The data only includes information about books and movies and does not include any personal information such as addresses or emails.

# 2.7 Data Loading and Availability

The data was loaded into MySQL workbench. It will also be available on the project’s public GitHub repository as a CSV.

|  |  |
| --- | --- |
| **3.** | **DATA QUALITY** |

Our KPIs were met in the objective of creating a list of books turned into movies and having information on both the movie and book. The extraction goals were met in the sense we found four total datasets that gave us information on different elements of the movies and the books. Transformation was tricky because of the nested dictionary issue, but overall ensured us that the two datasets can be merged together if the titles exist in both. The loading was also successful because the dataset now exists in MySQL workbench. The dataset only includes the title of books that match the movie title. If a book has a different name than the movie, then it is not included in the dataset. In addition, sequels may not be included if there was a book title different from the movie title.