**MOVIE IMDB Database**

**Extract Transform Load Project**

**Team 5**

**Beth Emborsky**

**Amy Guan**

**Shanker Nair**

August 1, 2020

**Table of Contents**

[1. Introduction 4](#_Toc47196455)

[2. Data Source 4](#_Toc47196456)

[2.1 Movies CSV File 4](#_Toc47196457)

[2.2 Names CSV File 5](#_Toc47196458)

[3. Methodology 6](#_Toc47196459)

[3.1 Extraction 6](#_Toc47196460)

[3.2 Transformation 6](#_Toc47196461)

[3.2.1. ERD Diagram 7](#_Toc47196462)

[3.2.2. Data Selection 7](#_Toc47196463)

[3.2.3. Data Aggregation 8](#_Toc47196464)

[4. Loading 10](#_Toc47196465)

**List of Figures**

[Figure 1 Kaggle IMDB Movies Dataset (https://www.kaggle.com/stefanoleone992/imdb-extensive-dataset) 4](#_Toc47196466)

[Figure 2 IMDB movies.csv Snapshot 5](#_Toc47196467)

[Figure 3 Flow Chart of ETL Methodology 6](#_Toc47196468)

[Figure 4 ERD Diagram 7](#_Toc47196469)

[Figure 5 ERD with 6 Supporting Tables Highlighted by Red Ellipses 9](#_Toc47196470)

[Figure 6 ERD with 4 Junction Tables Highlighted by Green Ellipses 10](#_Toc47196471)

**List of Tables**

[Table 1 List of Parameters Selection 8](#_Toc47196472)

[Table 2 List of Primary Keys Created using Python 9](#_Toc47196473)

# Introduction

The goal of this ETL project is to extract, transform and load the large movie IMDB datasets into a relational database. This report is organized as follows: Section 2 contains the data source; Methodologies including data extraction and transforming are detailed in Section 3; Section 4 summarizes the loading of the final relational database.

# Data Source

Two extensive IMDB dataset from Kaggle were selected for this project:

* One is a movie-oriented CSV file *IMDb movies.csv* with 81k+ records;
* The other is a cast-member-oriented file *IMDb names.csv* with 175k+ records.



Figure 1 Kaggle IMDB Movies Dataset (<https://www.kaggle.com/stefanoleone992/imdb-extensive-dataset>)

The two files are detailed in Section 2.1 and 2.2 below.

# Movies CSV File

The *IMDb movies.csv* file includes 81,273 movies with attributes such as movie id, movie title, year, country, language, production company, description, director, average rating, number of votes, genre, etc.

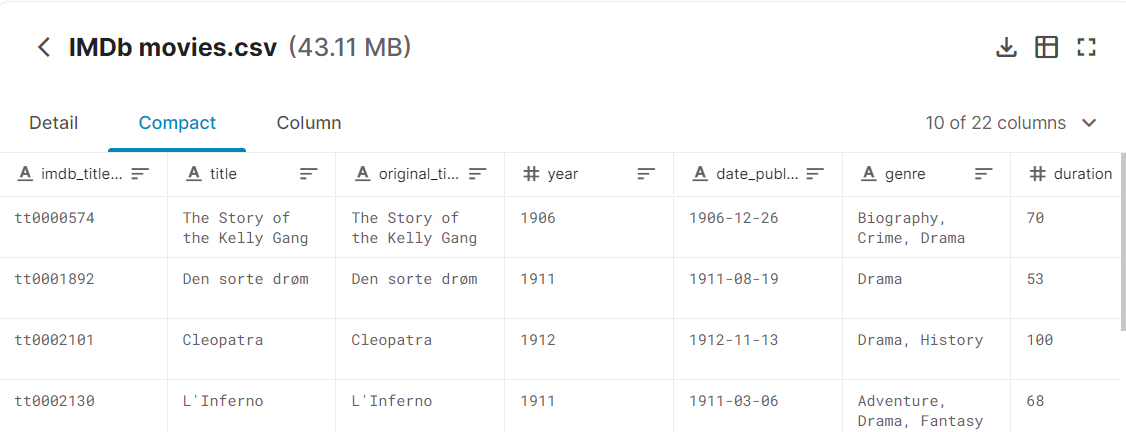


Figure 2 IMDB movies.csv Snapshot

# Names CSV File

The *IMDB names.csv* file dataset includes 175,719 cast members with personal attributes such as person id, job titles, movies participated, birth details, death details, height, spouses, children, etc.

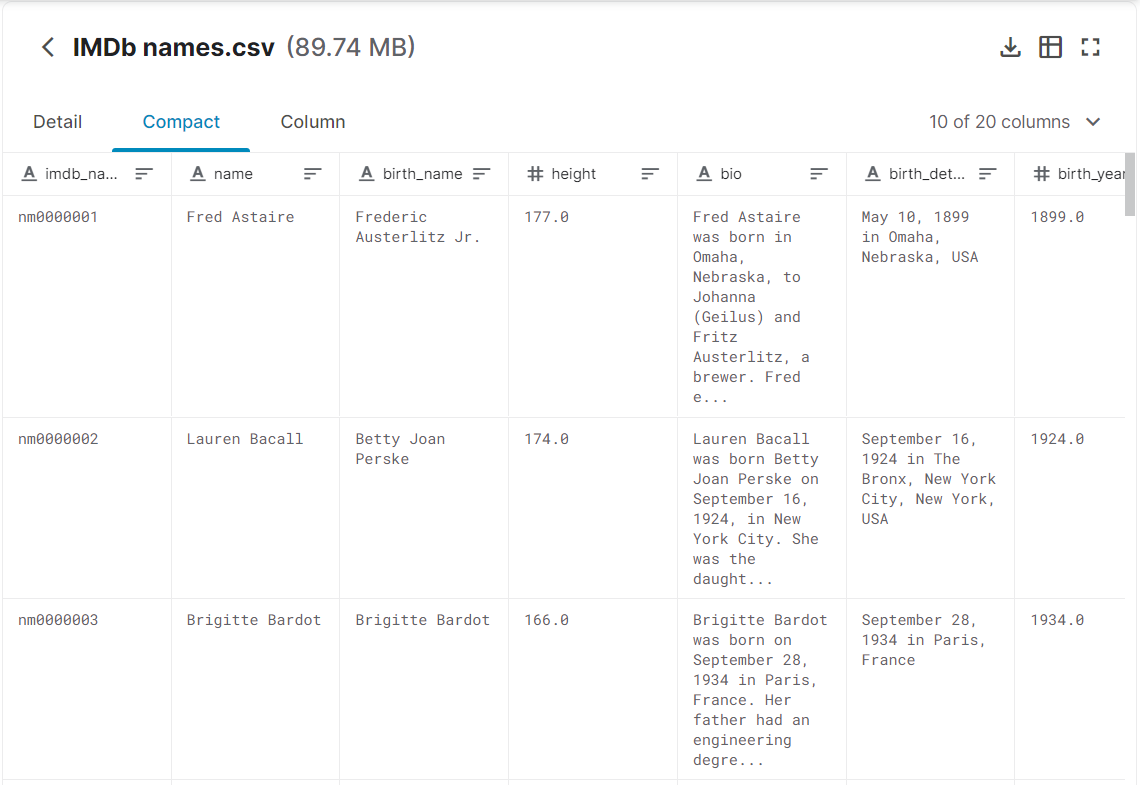


Figure 3 IMDB movies.csv Snapshot

# Methodology

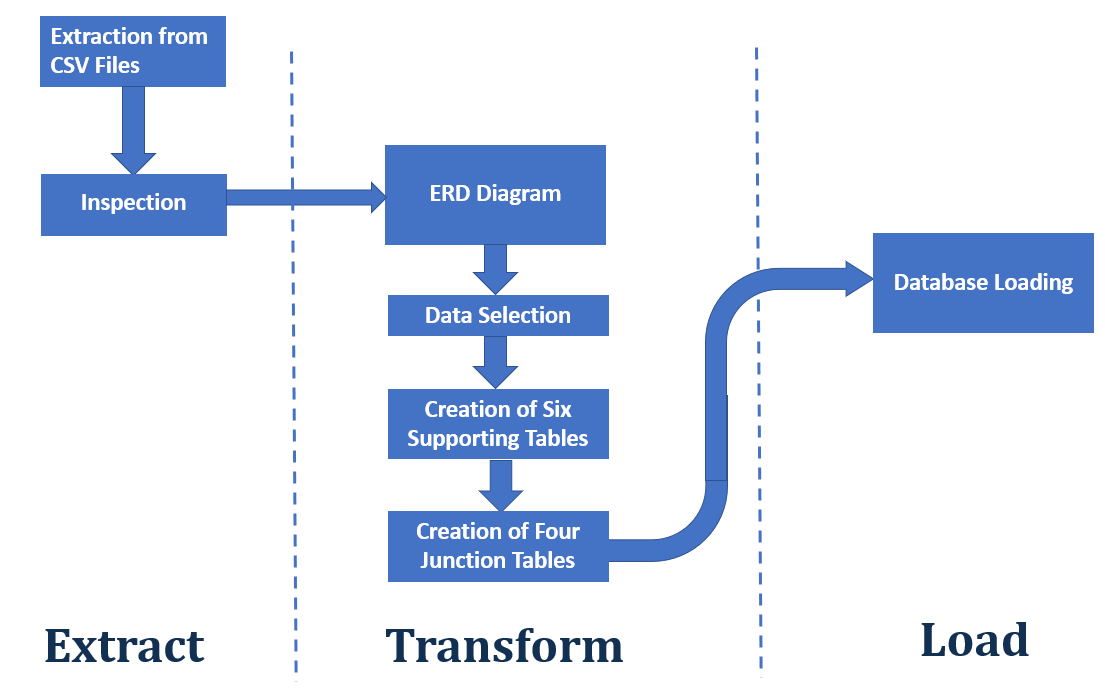


Figure 3 Flow Chart of ETL Methodology

# Extraction

After the CSV files were downloaded from Kaggle, Python was used to convert the CSVs into Panda DataFrame.

Initial inspection was conducted to find out the number of null and valid elements for all the columns.

Please refer to the ‘Extract’ Section of the master script ‘Project2.ipynb’.

# Transformation

# ERD Diagram

The following ERD diagram shows the collection of tables, their relationship, the information included in each table, as well as the primary keys and foreign keys. It provides the overview of the framework of the final database.

All the transformation steps are detailed in Sections 3.2.2 and 3.2.3.

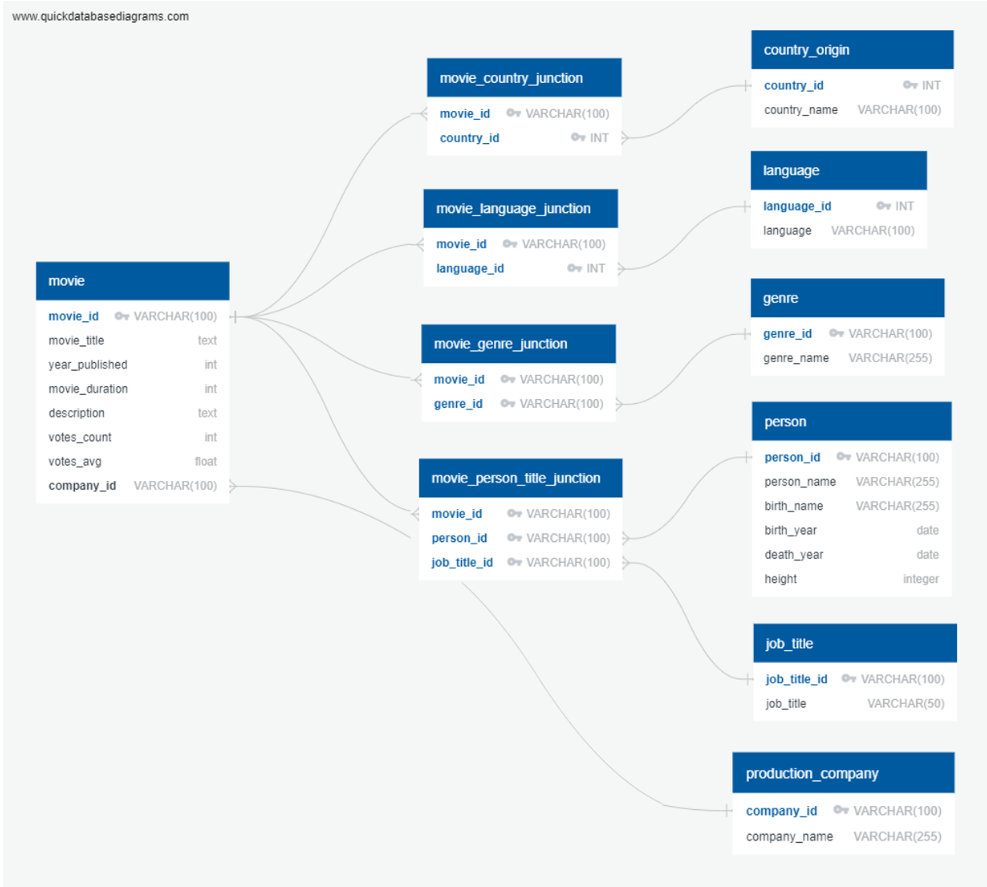


Figure 4 ERD Diagram

# Data Selection

After the CSV files were downloaded, we decided on the information of interest and selected them using Python. Please refer to the master script ‘Project2.ipynb’ for details.

**Table 1** below lists the parameters extracted, as well as their corresponding source CSV files.

**Table 1 List of Parameters Selection**

|  |  |
| --- | --- |
| **Source File** | **Parameters Selected** |
| IMDB movies.csv (81K+ record) | movie\_id |
| movie\_title |
| year\_published |
| movie\_duration |
| description |
| votes\_count |
| votes\_avg |
| **company** |
| **country** |
| **language** |
| **genre** |
| IMDB names.csv (175k+ record) | movie\_id |
| **job\_title** |
| **person\_name** |
| birth\_name |
| birth\_year |
| death\_year |
| height |

# Data Aggregation

After data inspection, it was discovered that special attention had to be paid to those parameters with one-to-many or many-to-many relationships, including company, country, language, genre, person name and job title.

The data aggregation process included two main steps, creation of primary keys for six supporting tables, and creation of four junction tables, as detailed below.

Each team member was responsible for creating two of the six supporting tables and their corresponding junction tables.

***Creation of Primary Keys for Six Supporting Tables***

In order to enable the relational dataset establishment of the six supporting tables using SQL as highlighted by red ellipse in Figure 4, the following unique primary keys were created:

**Table 2 List of Primary Keys Created using Python**

|  |  |
| --- | --- |
| **Tables** | **Created Primary Keys (absent from the CSV files)** |
| Person Table | person\_id |
| Job Title Table | job\_title\_id |
| Genre Table | genre\_id |
| Language Table | language\_id |
| Production Company Table | company\_id |
| Country Origin Table | country\_id |

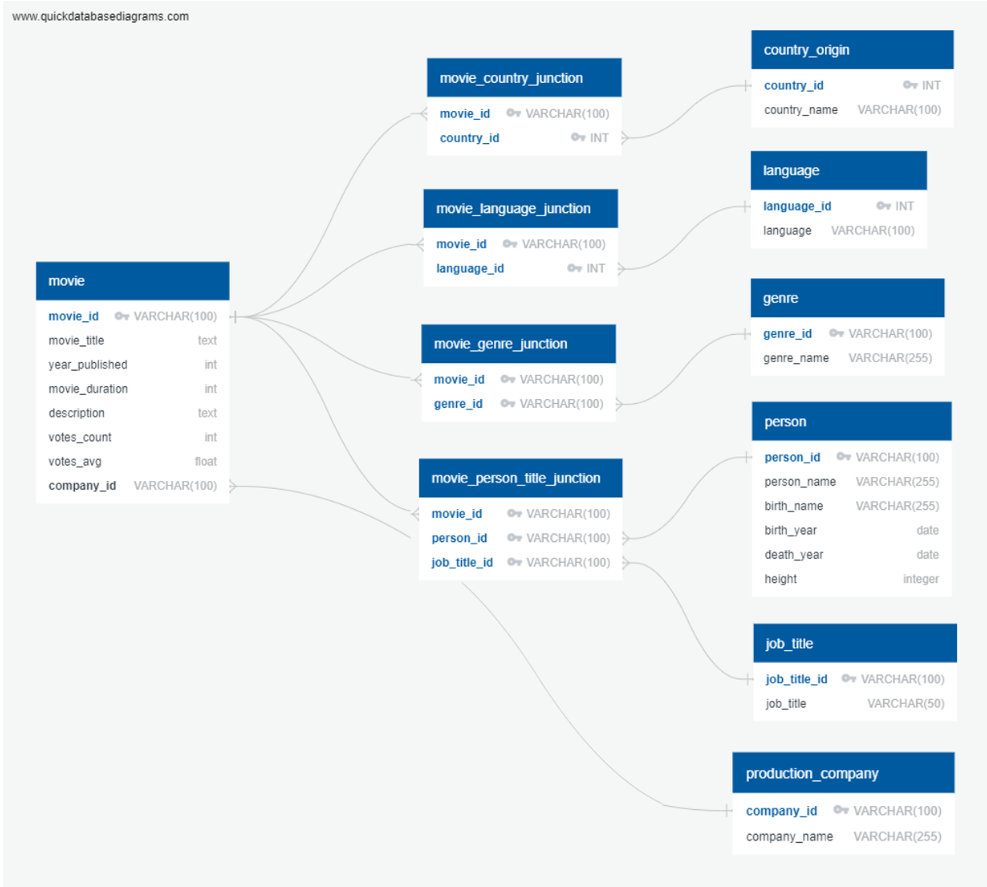


Figure 5 ERD with 6 Supporting Tables Highlighted by Red Ellipses

***Creation of Four Relational Junction Tables***

The four junction tables are essential to connect the main movie table and the supporting tables. Python was used to create the junction tables to ensure the bridging between the main movie table and the six supporting tables with the information of the country, language, genre, cast member, production company and job title.

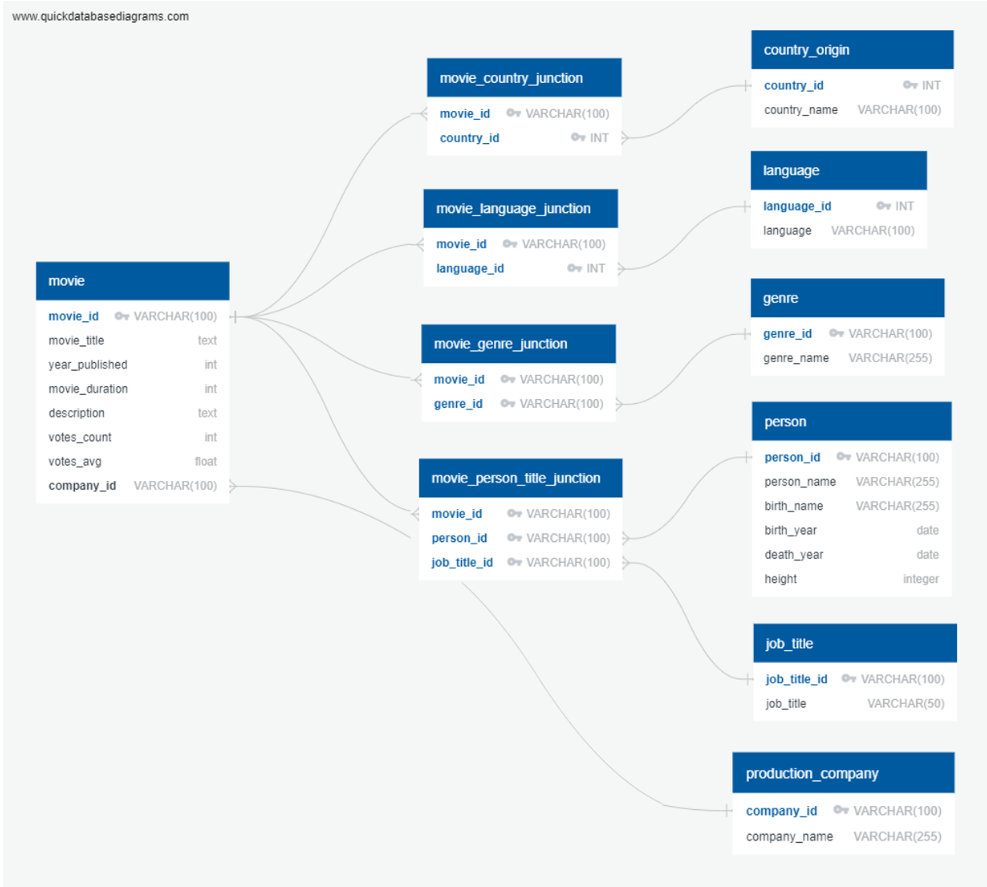


Figure 6 ERD with 4 Junction Tables Highlighted by Green Ellipses

# Load

Once the eleven pandas dataFrames were established for the main movie table, supporting tables and junction tables, SQL was used to load all into a SQL database ‘movies\_db’.