**ETL – Extract, Transform, Load**

**Looking Ahead**

In this module, you'll learn how to use the Extract, Transform, Load (ETL) process to create data pipelines. A data pipeline moves data from a source to a destination, and the ETL process creates data pipelines that also transform the data along the way. Analysis is impossible without access to good data, so creating data pipelines is often the first step before any analysis can be performed. Therefore, understanding ETL is an essential skill for data analysis.

**What You Will Learn**

By the end of this module, you will be able to:

Create an ETL pipeline from raw data to a SQL database.

Extract data from disparate sources using Python.

Clean and transform data using Pandas.

Use regular expressions to parse data and to transform text into numbers.

Load data with PostgreSQL.

**ETL – Extract, Transform, Load**

Amazing Prime Video was a platform for streaming movies and TV shows on Amazing Prime, the world’s largest online retailer. The amazing Prime video team would like to develop an algorithm to predict which low budget movies being released will become popular so that they can buy the streaming rights at a bargain. To inspire the team, have some fun, and connect with the local coding community, Amazing Prime has decided to sponsor a hackathon. Providing a clean data set of movie data and asking participants to predict the poplar pictures. Britta, a member of the Amazing Prime video team has been tasked with creating the datasets for the hackathon. There are two data sources: a scrape of Wikipedia for all movies released since 1990, and rating data from the Movie Land’s website. She’ll need to extract the data from the two sources, transform it into one clean data set, and finally load that data set into a SQL table. Let’s take a deeper look at that process.

**Challenge**

**Background**

Amazing Prime loves the dataset and wants to keep it updated on a daily basis. Britta needs your help to create an automated pipeline that takes in new data, performs the appropriate transformations, and loads the data into existing tables. You’ll need to refactor the code from this module to create one function that takes in the three files—Wikipedia data, Kaggle metadata, and the MovieLens rating data—and performs the ETL process by adding the data to a PostgreSQL database.

**What You're Creating**

This new assignment consists of four technical analysis deliverables. You will submit the following:

Deliverable 1: Write an ETL Function to Read Three Data Files

Deliverable 2: Extract and Transform the Wikipedia Data

Deliverable 3: Extract and Transform the Kaggle data

Deliverable 4: Create the Movie Database

**Overview**

After creating a movies and ratings dataset, Britta from Amazing Prime would like to keep it updated on a daily basis. To do so I have created an automated pipeline that does the following:

Takes in new, recent movie data from Wikipedia JSON file, Kaggle csv, and MovieLens csv.

Performs the appropriate transformations on the new data.

Loads the data into already existing tables.

**Analysis**

The project was successful and a movies with ratings database was created. Below are images from this created dataset

This is an image of the total amount of movies 6077 that are active in the database

This is an image of the total amount of ratings for these 6077 movies. There are 26,024,289!

Went through 26,024,289 movies in 747.4100549221039 Seconds.

By combining data from Wikipedia, Kaggle, and MovieLens and then removing any redundencies a complete database has been created with all pertinent information.

**Summary**

This database will be very helpful for the Hackathon for Britta and Amazing Prime. In the future, I would suggest adding movie genres as a column in this database. I would also suggest pulling information from additional websites such as movierankings.net, they give more of a fan rating than a critics rating. Finally, I would suggest to include the movie titles in the ratings table so that a random user knows exactly which movie is being represented instead of having to reference the movieid.

**Resources - Software**

Pandas

pgAdmin

PostgresSQL

Python

Regular Expressions

SQLAlchemy