Module 8 Challenge

New Attempt

Due Feb 28 by 12:59am **Points** 100 **Submitting** a text entry box or a website url

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

Files

Use the following links to download the Challenge starter codes.

ETL Deliverable 1 starter code (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_8/ETL_Deliverable1_starter_code.ipynb)

ETL Deliverable 2 starter code (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_8/ETL_Deliverable2_starter_code.ipynb)

ETL Deliverable 3 starter code (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_8/ETL_Deliverable3_starter_code.ipynb)

Deliverable 1: Write an ETL Function to Read Three Data Files (25 points)

Deliverable 1 Instructions

Using your knowledge of Python, Pandas, the ETL process, and code refactoring, write a function that reads in the three data files and creates three separate DataFrames.

REWIND

For this deliverable, you've already done the following in this module:

- **Lesson 8.2.1:** Load and extract the Wikipedia data
- **Lesson 8.2.2:** Extract the Kaggle Data

Download the `ETL_Deliverable1_starter_code.ipynb` file, add it to your Movies-ETL GitHub folder, and rename the file `ETL_function_test.ipynb`. Follow the instructions below to refactor the code from this module as indicated by the numbered comments in the starter code file.

1. In Step 1, create a function to read in the three files and give it a name.

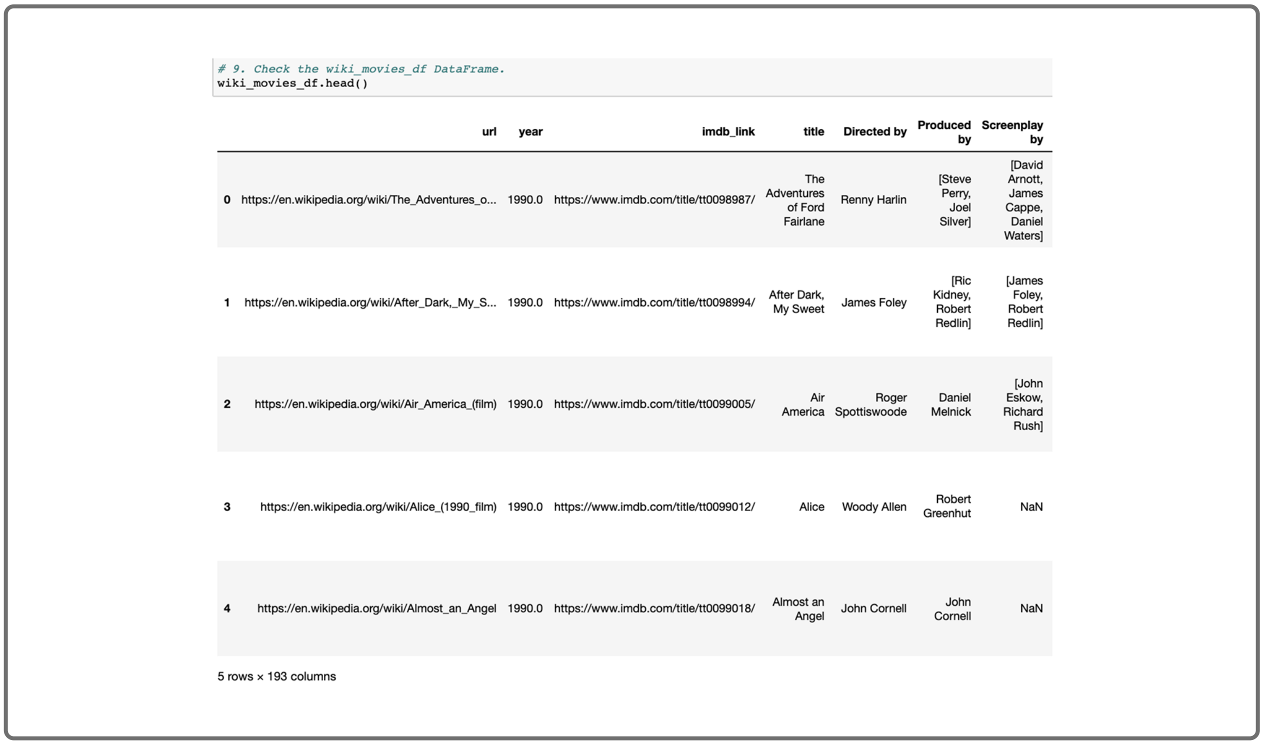
NOTE

> You do not need to pass any variables inside this function yet. In Step 7, you'll use the provided code to set the three variables for the files you'll use equal to the function created in Step 1. This is done in order to reassign the variable names in Step 8, which will allow you to display each DataFrame in steps 9-11.

2. In Step 2, read in the Kaggle metadata and MovieLens ratings CSV files as Pandas DataFrames.
3. In Step 3, open the Wikipedia JSON file and use the `json.load()` function to convert the JSON data to raw data.
4. In Step 4, read in the raw Wikipedia movie data as a Pandas DataFrame.
5. In Step 5, use the code provided to return the three DataFrames.
6. In Step 6, use the variables provided to create a path to the Wikipedia data, the Kaggle metadata, and the MovieLens rating data files.
7. In Step 7, set the three variables in Step 6 equal to the function created in Step 1.

8. In Step 8, set the DataFrames from the return statement equal to the file names in Step 6. In this step, you are reassigning the variables created in Step 6 to the variables in the return statement.
9. In Steps 9-11, check that all three files are converted to a DataFrame. See the images below for confirmation:

- The `wiki_movies_df` DataFrame

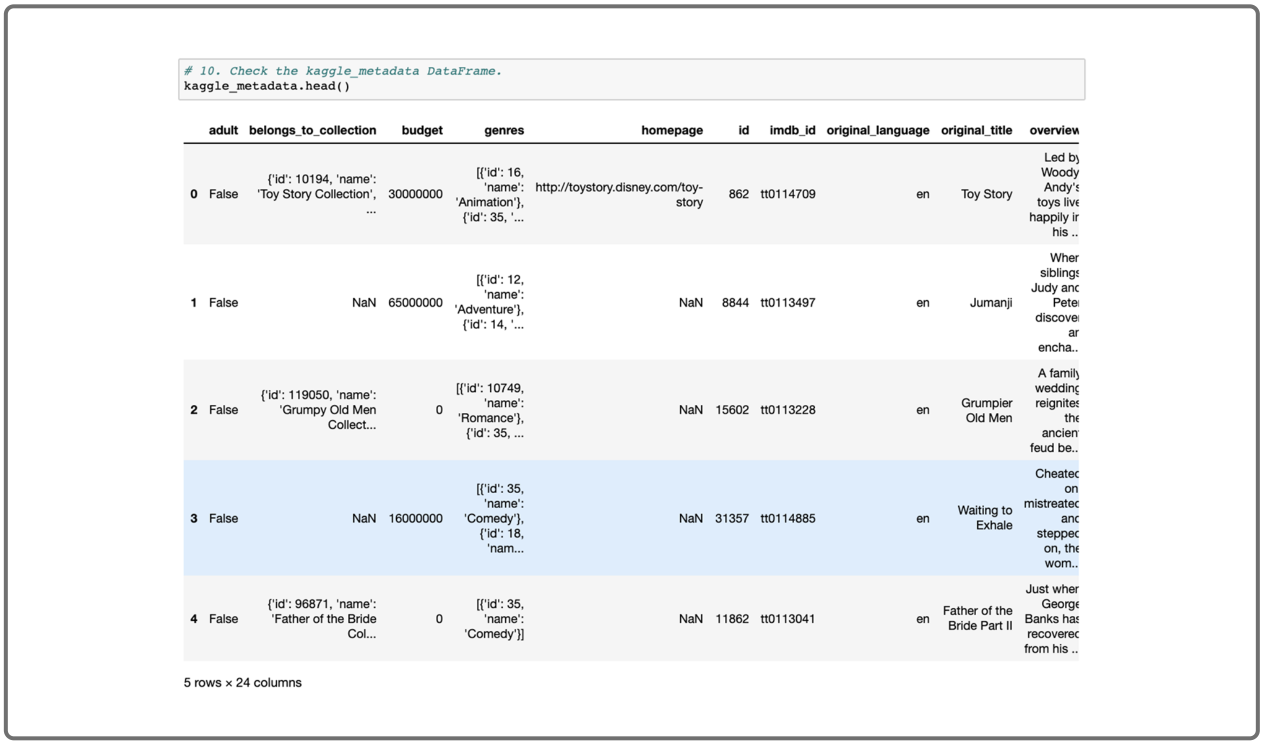


```
# 9. Check the wiki_movies_df DataFrame.
wiki_movies_df.head()
```

	url	year	imdb_link	title	Directed by	Produced by	Screenplay by
0	https://en.wikipedia.org/wiki/The_Adventures_o...	1990.0	https://www.imdb.com/title/tt0098987/	The Adventures of Ford Fairlane	Renny Harlin	[Steve Perry, Joel Silver]	[David Arnott, James Cappe, Daniel Waters]
1	https://en.wikipedia.org/wiki/After_Dark,_My_S...	1990.0	https://www.imdb.com/title/tt0098994/	After Dark, My Sweet	James Foley	[Ric Kidney, Robert Redlin]	[James Foley, Robert Redlin]
2	https://en.wikipedia.org/wiki/Air_America_(film)	1990.0	https://www.imdb.com/title/tt0099005/	Air America	Roger Spottiswoode	Daniel Melnick	[John Eskow, Richard Rush]
3	https://en.wikipedia.org/wiki/Alice_(1990_film)	1990.0	https://www.imdb.com/title/tt0099012/	Alice	Woody Allen	Robert Greenhut	NaN
4	https://en.wikipedia.org/wiki/Almost_an_Angel	1990.0	https://www.imdb.com/title/tt0099018/	Almost an Angel	John Cornell	John Cornell	NaN

5 rows × 193 columns

- The `kaggle_metadata` DataFrame

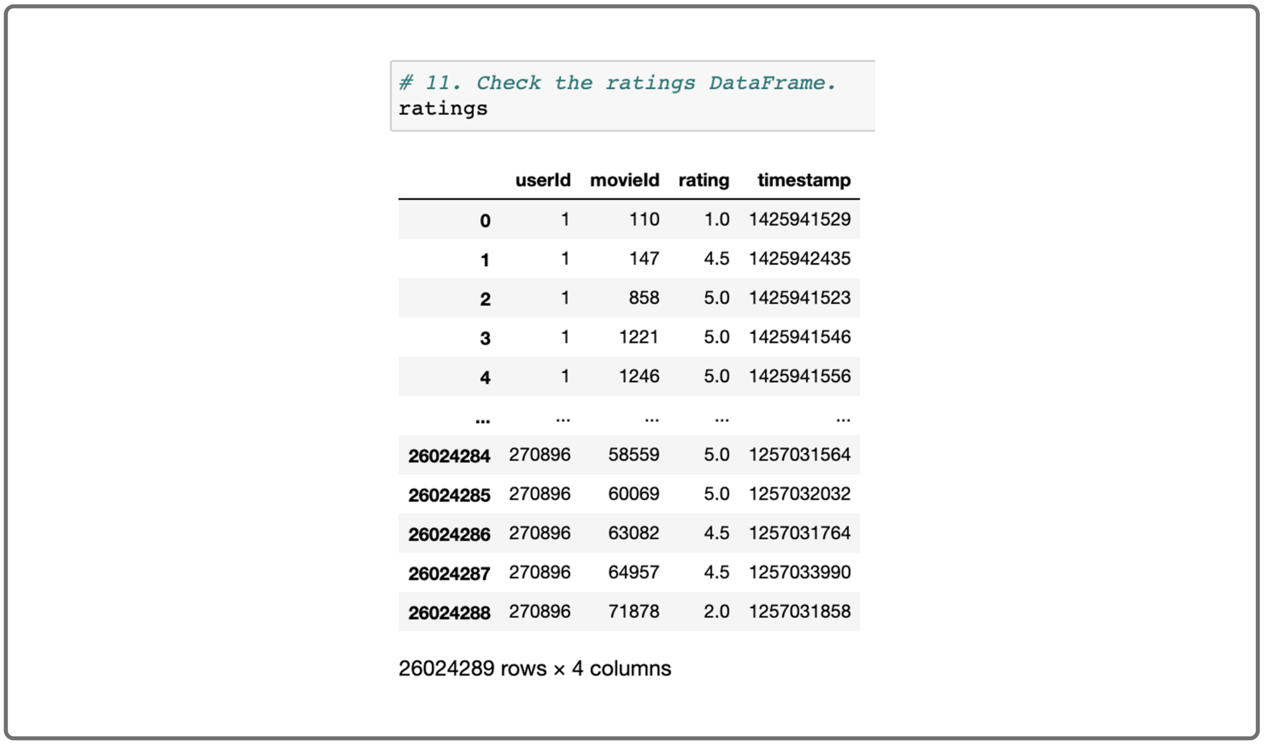


```python
# 10. Check the kaggle_metadata DataFrame.
kaggle_metadata.head()
```

	adult	belongs_to_collection	budget	genres	homepage	id	imdb_id	original_language	original_title	overview
0	False	{'id': 10194, 'name': 'Toy Story Collection', ...	30000000	[{'id': 16, 'name': 'Animation'}, {'id': 35, '...	http://toystory.disney.com/toy-story	862	tt0114709	en	Toy Story	Led by Woody Andy's toys live happily ir his ..
1	False	NaN	65000000	[{'id': 12, 'name': 'Adventure'}, {'id': 14, '...	NaN	8844	tt0113497	en	Jumanji	Wher siblings Judy anc Peter discover ar encha..
2	False	{'id': 119050, 'name': 'Grumpy Old Men Collect...	0	[{'id': 10749, 'name': 'Romance'}, {'id': 35, ...	NaN	15602	tt0113228	en	Grumpier Old Men	A family wedding reignites the ancient feud be..
3	False	NaN	16000000	[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'nam...	NaN	31357	tt0114885	en	Waiting to Exhale	Cheated on mistreated and stepped on, the wom..
4	False	{'id': 96871, 'name': 'Father of the Bride Col...	0	[{'id': 35, 'name': 'Comedy'}]	NaN	11862	tt0113041	en	Father of the Bride Part II	Just wher George Banks has recovered from his ..

5 rows × 24 columns

- The `ratings` DataFrame



```python
# 11. Check the ratings DataFrame.
ratings
```

	userId	movieId	rating	timestamp
0	1	110	1.0	1425941529
1	1	147	4.5	1425942435
2	1	858	5.0	1425941523
3	1	1221	5.0	1425941546
4	1	1246	5.0	1425941556
...	...	...	...	...
26024284	270896	58559	5.0	1257031564
26024285	270896	60069	5.0	1257032032
26024286	270896	63082	4.5	1257031764
26024287	270896	64957	4.5	1257033990
26024288	270896	71878	2.0	1257031858

26024289 rows × 4 columns

10. After you confirm that all three DataFrames are correct, save the `ETL_function_test.ipynb` file in your Movies-ETL GitHub folder.

Deliverable 1 Requirements

You will earn a perfect score for Deliverable 1 by completing all requirements below:

- An ETL function is written to read in the three data files. **(10 pt)**
- The function converts the Wikipedia JSON file to a Pandas DataFrame, and the DataFrame is displayed in the `ETL_function_test.ipynb` file. **(5 pt)**
- The function converts the Kaggle metadata file to a Pandas DataFrame, and the DataFrame is displayed in the `ETL_function_test.ipynb` file. **(5 pt)**
- The function converts the MovieLens ratings data file to a Pandas DataFrame, and the DataFrame is displayed in the `ETL_function_test.ipynb` file. **(5 pt)**

Deliverable 2: Extract and Transform the Wikipedia Data (30 points)

Deliverable 2 Instructions

Using your knowledge of Python, Pandas, the ETL process, and code refactoring, extract and transform the Wikipedia data so you can merge it with the Kaggle metadata. While extracting the IMDb IDs using a regular expression string and dropping duplicates, use a `try-except` block to catch errors.

REWIND

For this deliverable, you've already done the following in this module:

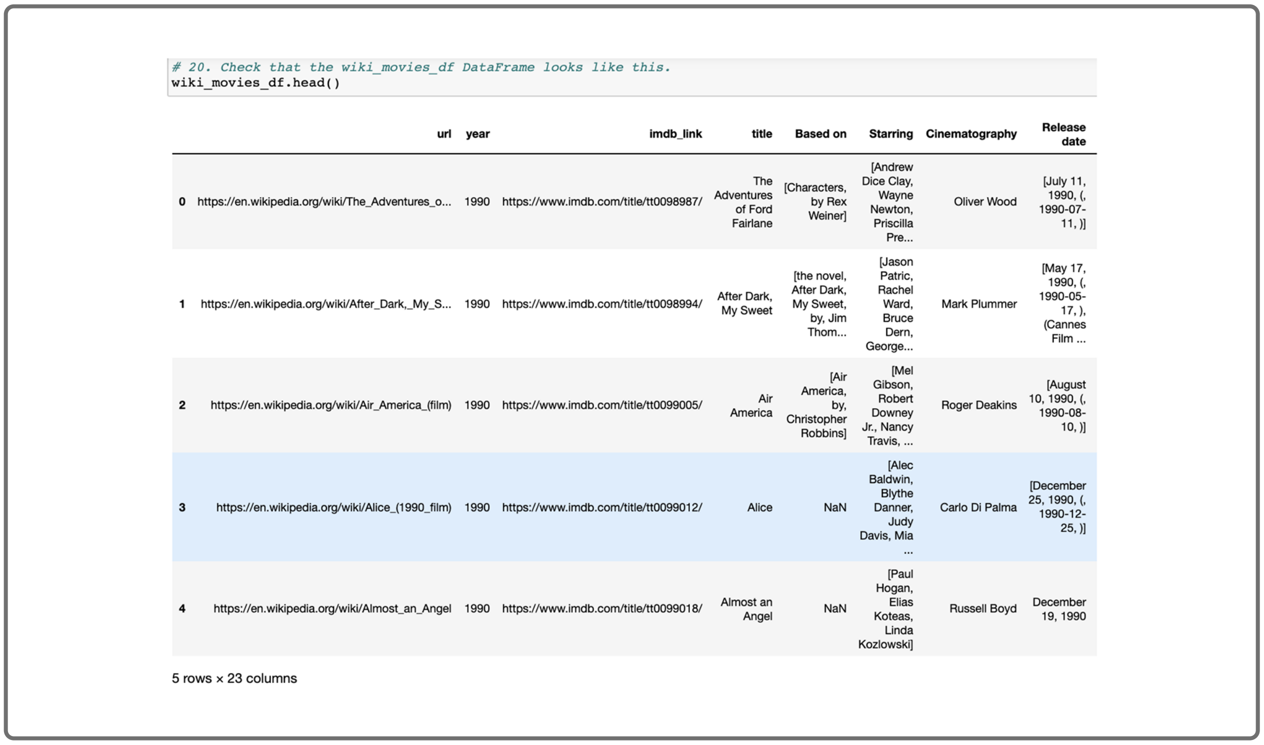
- **Lesson 8.3.3:** Clean and filter data with list comprehensions
- **Lesson 8.3.4:** Write functions
- **Lesson 8.3.6:** Create the clean movie function
- **Lesson 8.3.7:** Remove duplicates with regular expressions
- **Lesson 8.3.7:** Remove columns with null values
- **Lesson 8.3.8:** Drop null values and convert data to string values
- **Lesson 8.3.10:** Clean the box office data
- **Lesson 8.3.11:** Clean the budget data, the release date, and the running time

Download the `ETL_Deliverable2_starter_code.ipynb` file, add it to your Movies-ETL GitHub folder, and rename the file `ETL_clean_wiki_movies.ipynb`. Follow the instructions below to refactor the code from this module as indicated by the numbered comments in the starter code file.

1. In Step 1, add the code from this module for the clean movie function that takes in the argument "movie".

2. In Step 2, add the function you created in Deliverable 1 that reads in the three data files.
3. In Step 3, inside the function you created in Deliverable 1, remove the code that creates the `wiki_movies_df` DataFrame from the `wiki_movies_raw` file, then write a list comprehension that filters out TV shows from the `wiki_movies_raw` file.
4. In Step 4, write a list comprehension to iterate through the cleaned wiki movies list that you created in Step 3.
5. In Step 5, read in the cleaned movies list from Step 4 as a DataFrame.
6. In Step 6, write a `try-except` block that will catch errors while extracting the IMDb IDs with a regular expression string and dropping any `imdb_id` duplicates. If there is an error, capture and print the exception.
7. In Step 7, write a list comprehension to keep the columns that have non-null values from the DataFrame created in Step 5, then create a `wiki_movies_df` DataFrame from the list.
8. In Step 8, create a variable that will hold all the non-null values from the "Box office" column.
9. In Step 9, convert the box office data created in Step 8 to string values using the lambda and join functions.
10. In Step 10, write a regular expression to match the six elements of `form_one` of the box office data.
11. In Step 11, write a regular expression to match the three elements of `form_two` of the box office data.
12. In Step 12, add the `parse_dollars()` function.
13. In Step 13, add the code that cleans the box office column in the `wiki_movies_df` DataFrame using the `form_one` and `form_two` lists created in Steps 10 and 11, respectively.

14. In Step 14, add code that cleans the budget column in the `wiki_movies_df` DataFrame.
15. In Step 15, add code that cleans the release date column in the `wiki_movies_df` DataFrame.
16. In Step 16, add code that cleans the running time column in the `wiki_movies_df` DataFrame.
17. In Step 17, use the variables provided to create a path to the Wikipedia data, the Kaggle metadata, and the MovieLens rating data files.
18. In Step 18, set the three variables in Step 17 equal to the function created in Deliverable 1.
19. In Step 19, set the `wiki_movies_df` equal to the `wiki_file` variable.
20. In Step 20, check that your `wiki_movies_df` DataFrame looks like this image:

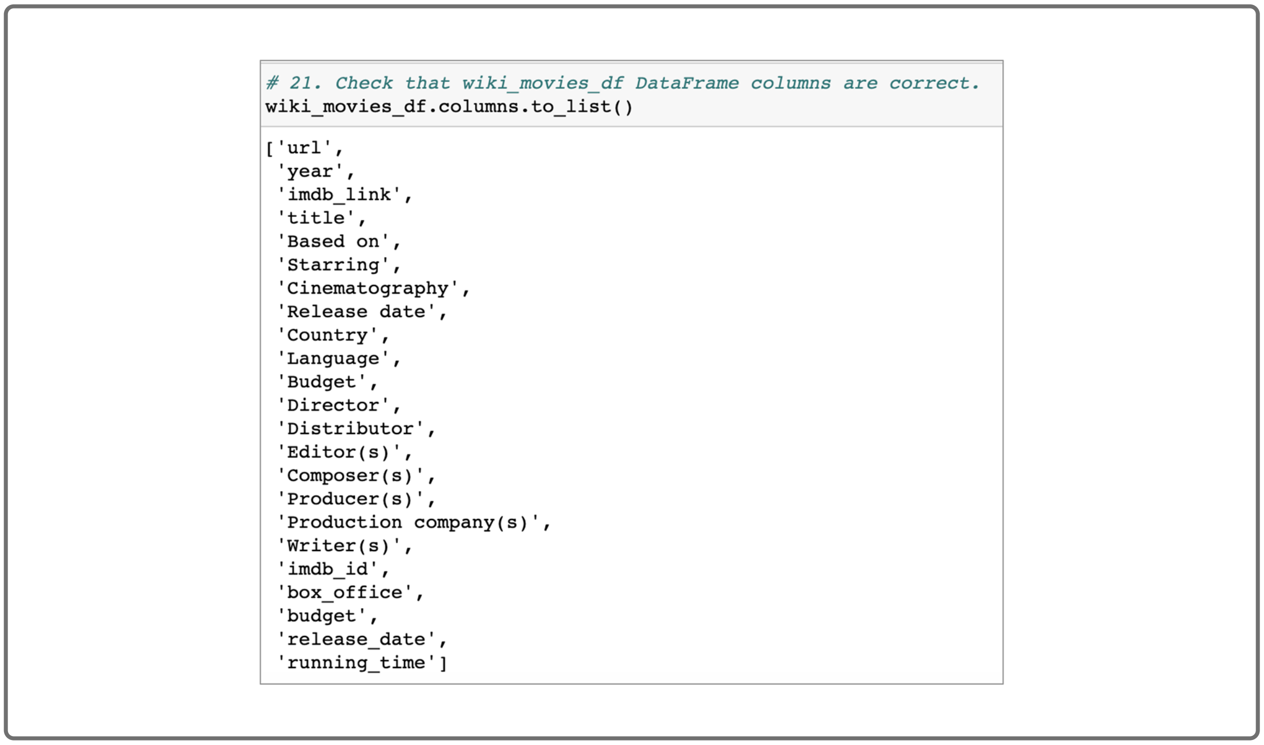


```
# 20. Check that the wiki_movies_df DataFrame looks like this.
wiki_movies_df.head()
```

	url	year	imdb_link	title	Based on	Starring	Cinematography	Release date
0	https://en.wikipedia.org/wiki/The_Adventures_o...	1990	https://www.imdb.com/title/tt0098987/	The Adventures of Ford Fairlane	[Characters, by Rex Weiner]	[Andrew Dice Clay, Wayne Newton, Priscilla Pre...	Oliver Wood	[July 11, 1990, (, 1990-07-11,)]
1	https://en.wikipedia.org/wiki/After_Dark,_My_S...	1990	https://www.imdb.com/title/tt0098994/	After Dark, My Sweet	[the novel, After Dark, My Sweet, by, Jim Thom...	[Jason Patric, Rachel Ward, Bruce Dern, George...	Mark Plummer	[May 17, 1990, (, 1990-05-17,), (Cannes Film ...
2	https://en.wikipedia.org/wiki/Air_America_(film)	1990	https://www.imdb.com/title/tt0099005/	Air America	[Air America, by, Christopher Robbins]	[Mel Gibson, Robert Downey Jr., Nancy Travis, ...	Roger Deakins	[August 10, 1990, (, 1990-08-10,)]
3	https://en.wikipedia.org/wiki/Alice_(1990_film)	1990	https://www.imdb.com/title/tt0099012/	Alice	NaN	[Alec Baldwin, Blythe Danner, Judy Davis, Mia ...	Carlo Di Palma	[December 25, 1990, (, 1990-12-25,)]
4	https://en.wikipedia.org/wiki/Almost_an_Angel	1990	https://www.imdb.com/title/tt0099018/	Almost an Angel	NaN	[Paul Hogan, Elias Koteas, Linda Kozlowski]	Russell Boyd	December 19, 1990

5 rows × 23 columns

21. In Step 21, add the columns from `wiki_movies_df` DataFrame to a list, and confirm that they are the same as this image:



```
# 21. Check that wiki_movies_df DataFrame columns are correct.
wiki_movies_df.columns.to_list()
```

```
['url',
 'year',
 'imdb_link',
 'title',
 'Based on',
 'Starring',
 'Cinematography',
 'Release date',
 'Country',
 'Language',
 'Budget',
 'Director',
 'Distributor',
 'Editor(s)',
 'Composer(s)',
 'Producer(s)',
 'Production company(s)',
 'Writer(s)',
 'imdb_id',
 'box_office',
 'budget',
 'release_date',
 'running_time']
```

22. After you confirm that the `wiki_movies_df` DataFrame is correct, save the `ETL_clean_wiki_movies.ipynb` file in your Movies-ETL GitHub folder.

Deliverable 2 Requirements

You will earn a perfect score for Deliverable 2 by completing all requirements below:

- The TV shows are filtered out, and the `wiki_movies_df` DataFrame is created. **(3 pt)**
- A `try-except` block is used to catch errors while extracting the IMDb IDs with a regular expression and dropping duplicate IDs. **(5 pt)**
- The extraction and transformation of the Wikipedia data in the ETL function does the following:

  - A list comprehension is used to keep columns with non-null values. **(3 pt)**
  - The non-null box office data is converted to string values using the lambda and join functions. **(3 pt)**
  - A regular expression is used to match the six elements of "form_one" of the box office data. **(2 pt)**
  - A regular expression is used to match the three elements of "form_two" of the box office data. **(2 pt)**
  - The following columns are cleaned in the Wikipedia DataFrame: **(8 pt)**
    - The box office column
    - The budget column
    - The release date column
    - The running time column
- The cleaned Wikipedia data is converted to a Pandas DataFrame, and the DataFrame is displayed in the `ETL_clean_wiki_movies.ipynb` file. **(4 pt)**

Deliverable 3: Extract and Transform the Kaggle Data (30 points)

Deliverable 3 Instructions

Using your knowledge of Python, Pandas, the ETL process, and code refactoring, extract and transform the Kaggle metadata and MovieLens rating data, then convert the transformed data into separate DataFrames. Then, you'll merge the Kaggle metadata DataFrame with the Wikipedia movies DataFrame to create the `movies_df` DataFrame. Finally, you'll merge the MovieLens rating data DataFrame with the `movies_df` DataFrame to create the `movies_with_ratings_df`.

REWIND

For this deliverable, you've already done the following in this module:

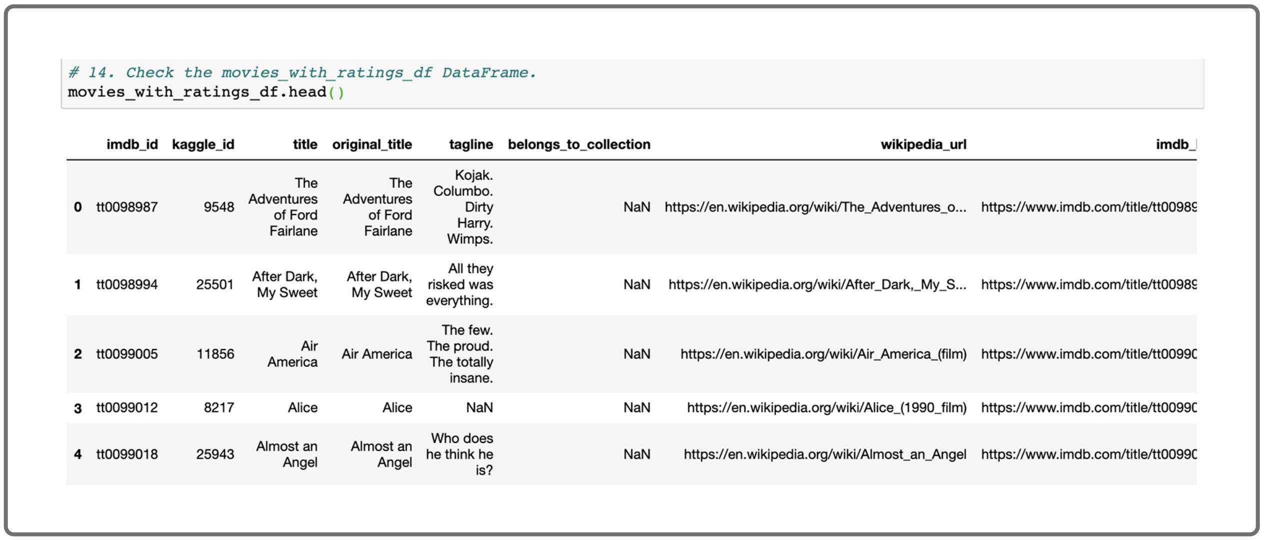
- **Lesson 8.3.11:** Clean the budget data, the release date, and the running time
- **Lesson 8.3.12:** Clean the Kaggle data
- **Lesson 8.4.1:** Merge Wikipedia and Kaggle DataFrames
- **Lesson 8.4.2:** Transform and merge the ratings data

Download the `ETL_Deliverable3_starter_code.ipynb` file, add it to your Movies-ETL GitHub folder, and rename the file `ETL_clean_kaggle_data.ipynb`. Follow the instructions below to refactor the code from this module as indicated by the numbered comments in the starter code file.

1. In Step 1, add the function you created in Deliverable 1 that reads in the three data files and creates the `kaggle_metadata` and `ratings` DataFrames.
2. Before Step 2, add all the code you wrote for Deliverable 2.

3. In Step 2, below the code that cleans the running time column in the `wiki_movies_df` DataFrame from Deliverable 2, add the code that cleans the Kaggle metadata.
4. In Step 3, merge the `wiki_movies_df` DataFrame and the `kaggle_metadata` DataFrames, then name the new DataFrame, `movies_df`.
5. In Step 4, drop unnecessary columns from the `movies_df` DataFrame.
6. In Step 5, add the `fill_missing_kaggle_data()` function that fills in the missing Kaggle data on the `movies_df` DataFrame.
7. In Step 6, call the `fill_missing_kaggle_data()` function with the `movies_df` DataFrame and the Kaggle and Wikipedia columns to be cleaned as the arguments.
8. In Step 7, filter the `movies_df` DataFrame to keep the necessary columns.
9. In Step 8, rename the columns in the `movies_df` DataFrame.
10. In Step 9, transform and merge the ratings DataFrame with the `movies_df` DataFrame, name the new DataFrame `movies_with_ratings_df`, then clean the `movies_with_ratings_df` DataFrame.
11. In Step 10, use the variables provided to create a path to the Wikipedia data, the Kaggle metadata, and the MovieLens rating data files.
12. In Step 11, set the three variables from Step 17 of Deliverable 2 equal to the function created in Deliverable 1.
13. In Step 12, set the DataFrames from the return statement after Step 9 equal to the file names in Step 11.
14. In Step 13, check that your `wiki_movies_df` DataFrame is the same as in Deliverable 2.

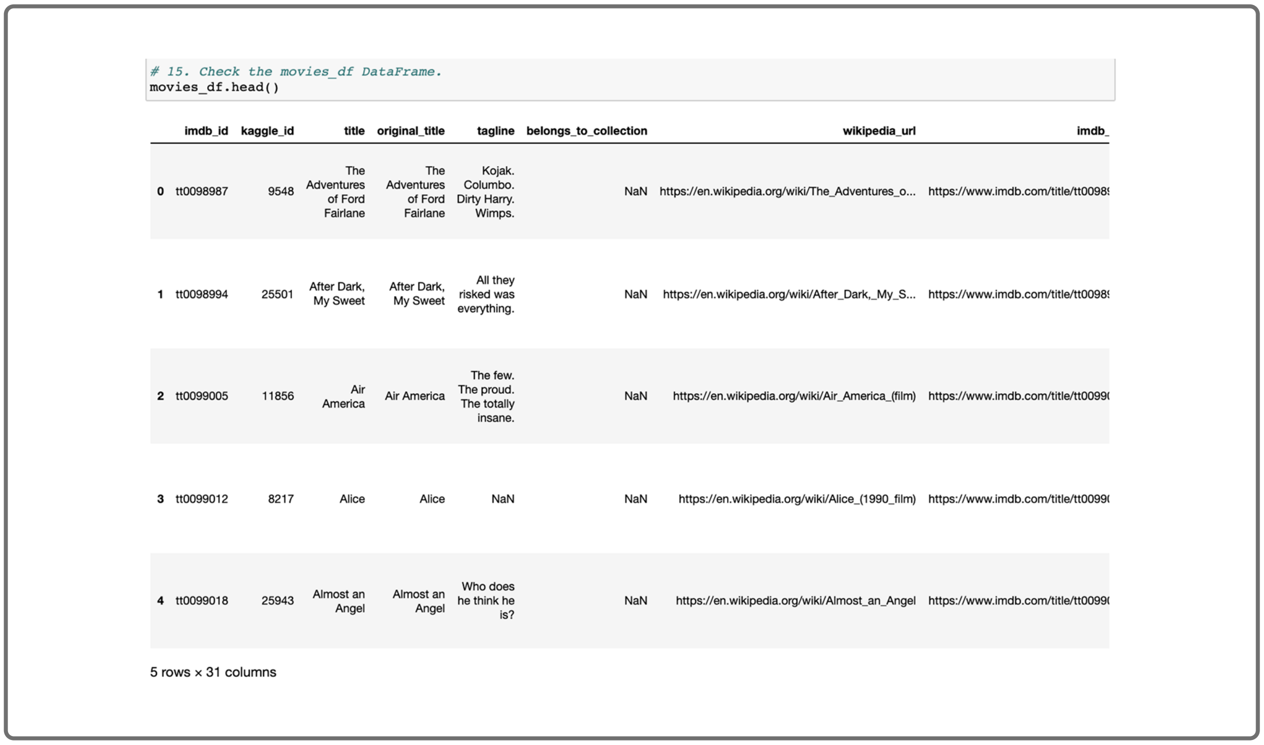
15. In Step 14, check that your `movies_with_ratings_df` DataFrame looks like this image:



```
# 14. Check the movies_with_ratings_df DataFrame.
movies_with_ratings_df.head()
```

	imdb_id	kaggle_id	title	original_title	tagline	belongs_to_collection	wikipedia_url	imdb_
0	tt0098987	9548	The Adventures of Ford Fairlane	The Adventures of Ford Fairlane	Kojak. Columbo. Dirty Harry. Wimps.	NaN	https://en.wikipedia.org/wiki/The_Adventures_o...	https://www.imdb.com/title/tt00989
1	tt0098994	25501	After Dark, My Sweet	After Dark, My Sweet	All they risked was everything.	NaN	https://en.wikipedia.org/wiki/After_Dark,_My_S...	https://www.imdb.com/title/tt00989
2	tt0099005	11856	Air America	Air America	The few. The proud. The totally insane.	NaN	https://en.wikipedia.org/wiki/Air_America_(film)	https://www.imdb.com/title/tt00990
3	tt0099012	8217	Alice	Alice	NaN	NaN	https://en.wikipedia.org/wiki/Alice_(1990_film)	https://www.imdb.com/title/tt00990
4	tt0099018	25943	Almost an Angel	Almost an Angel	Who does he think he is?	NaN	https://en.wikipedia.org/wiki/Almost_an_Angel	https://www.imdb.com/title/tt00990

16. In Step 15, check that your `movies_df` DataFrame looks like this image:



```
# 15. Check the movies_df DataFrame.
movies_df.head()
```

	imdb_id	kaggle_id	title	original_title	tagline	belongs_to_collection	wikipedia_url	imdb_
0	tt0098987	9548	The Adventures of Ford Fairlane	The Adventures of Ford Fairlane	Kojak. Columbo. Dirty Harry. Wimps.	NaN	https://en.wikipedia.org/wiki/The_Adventures_o...	https://www.imdb.com/title/tt0098
1	tt0098994	25501	After Dark, My Sweet	After Dark, My Sweet	All they risked was everything.	NaN	https://en.wikipedia.org/wiki/After_Dark,_My_S...	https://www.imdb.com/title/tt0098
2	tt0099005	11856	Air America	Air America	The few. The proud. The totally insane.	NaN	https://en.wikipedia.org/wiki/Air_America_(film)	https://www.imdb.com/title/tt0099
3	tt0099012	8217	Alice	Alice	NaN	NaN	https://en.wikipedia.org/wiki/Alice_(1990_film)	https://www.imdb.com/title/tt0099
4	tt0099018	25943	Almost an Angel	Almost an Angel	Who does he think he is?	NaN	https://en.wikipedia.org/wiki/Almost_an_Angel	https://www.imdb.com/title/tt0099

5 rows × 31 columns

17. After you confirm that all three DataFrames are correct, save the `ETL_clean_kaggle_data.ipynb` file in your Movies-ETL GitHub folder.

Deliverable 3 Requirements

You will earn a perfect score for Deliverable 3 by completing all requirements below:

- The extraction and transformation of the Kaggle metadata using the ETL function does the following:
  - The Kaggle metadata is cleaned. **(4 pt)**
  - The Wikipedia and Kaggle DataFrames are merged. **(3 pt)**
  - The following is performed on the merged Wikipedia and Kaggle DataFrames to create the `movies_df`: **(8 pt)**
    - Unnecessary columns are dropped.
    - A function is used to fill in the missing Kaggle data.
    - The `movies_df` DataFrame is filtered to keep specific columns.
    - The `movies_df` DataFrame columns are renamed.
- The extraction and transformation of the MovieLens ratings data using the ETL function does the following:
  - The ratings counts are cleaned. **(3 pt)**
  - The `movies_df` DataFrame is merged with the cleaned ratings DataFrame to create the `movies_with_ratings_df` DataFrame. **(4 pt)**
  - The empty values in the `movies_with_ratings_df` DataFrame are filled with “0”. **(3 pt)**

- The `movies_with_ratings_df` and the `movies_df` DataFrames are displayed in the `ETL_clean_kaggle_data.ipynb` file. **(5 pt)**

Deliverable 4: Create the Movie Database (15 points)

Deliverable 4 Instructions

Use your knowledge of Python, Pandas, the ETL process, code refactoring, and PostgreSQL to add the `movies_df` DataFrame and MovieLens rating CSV data to a SQL database.

REWIND

For this deliverable, you've already done the following in this module:

- **Lesson 8.5.1:** Create and connect to the database, then import data

Make a copy of the `ETL_clean_kaggle_data.ipynb` file in the Movies-ETL GitHub, and rename the file `ETL_create_database.ipynb`. Follow the instructions below to add the `movies_df` DataFrame and MovieLens rating CSV data to a SQL database.

1. In the first cell, uncomment the `# from config import db_password` so this code is working.
2. Remove the return statement, `return wiki_movies_df, movies_with_ratings_df, movies_df`.
3. After Step 9, `Transform and merge the ratings DataFrame`, add the code to create the connection to the PostgreSQL database, then add the `movies_df` DataFrame to a SQL database.

Hint: Use `'replace'` for the `if_exists` parameter so that the `movies_df` DataFrame data won't be added to the table again.

4. Before reading in the MovieLens rating CSV data, drop the `ratings` table in pgAdmin.
5. Add the code that prints out the elapsed time to import each row.
6. Refactor Step 11 of Deliverable 3 so that you pass in the variables for the files created in Step 10 of Deliverable 3 in the function created in Deliverable 1.
7. Run the program.
8. After the program has finished, run a query on the PostgreSQL database that retreives the number of rows for the `movies` and `ratings` tables.
9. After you confirm that the `movies` table has 6,052 rows and the `ratings` table has 26,024,289 rows, take a screenshot of each query and the output, then save them as `movies_query.png` and `ratings_query.png`, respectively.
10. Save the `ETL_create_database.ipynb` file in your Movies-ETL GitHub folder.
11. Save the `movies_query.png` and `ratings_query.png` files in the Resources folder.

Deliverable 4 Requirements

You will earn a perfect score for Deliverable 4 by completing all requirements below:

- The data from the `movies_df` DataFrame replaces the current data in the movies table in the SQL database, as determined by the `movies_query.png`. **(5 pt)**
- The data from the MovieLens rating CSV file is added to the `ratings` table in the SQL database, as determined by the `ratings_query.png`. **(5 pt)**
- The elapsed time to add the data to the database is displayed in the `ETL_create_database.ipynb` file. **(5 pt)**

Submission

Once you're ready to submit, make sure to check your work against the rubric to ensure you are meeting the requirements for this Challenge one final time. It's easy to overlook items when you're in the zone!

As a reminder, the deliverables for this Challenge are as follows:

- 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

IMPORTANT

> Don’t clear the output of your Jupyter Notebook files. Doing so will result in a lower score.

Upload the following to your Movies-ETL GitHub repository:

1. The `ETL_function_test.ipynb` file
2. The `ETL_clean_wiki_movies.ipynb` file
3. The `ETL_clean_kaggle_data.ipynb` file
4. The `ETL_create_database.ipynb` file
5. The Resources folder with the `wikipedia_movies.json`, `movies_metadata.csv`, `movies_query.png`, and `ratings_query.png` files.
6. A README.md that describes the purpose of the repository. Although there is no graded written analysis for this Challenge, it is encouraged and good practice to add a brief description of your project.

To submit your challenge assignment for grading in Bootcamp Spot, click Start Assignment, click the Website URL tab, then provide the URL of your Movies-ETL GitHub repository, and then click Submit. Comments are disabled for graded submissions in BootCampSpot. If you have questions about your feedback, please notify your instructional staff or the Student Success Manager. If you would like to resubmit your work for an improved grade, you can use the **Re-Submit Assignment** button to upload new links. You may resubmit up to 3 times for a total of 4 submissions.

IMPORTANT

> Once you receive feedback on your Challenge, make any suggested updates or adjustments to your work. Then, add this week’s Challenge to your professional portfolio.

NOTE

> You are allowed to miss up to two Challenge assignments and still earn your certificate. If you complete all Challenge assignments, your lowest two grades will be dropped. If you wish to skip this assignment, click Next, and move on to the next Module.

Module-8 Rubric

Criteria	Ratings					Pts
Deliverable 1: Write an ETL function to read three data files	**25 to >23.0 pts Demonstrating Proficiency** The ETL function does the following: ✓ The three data files are passed into the function. ✓ All three data sets are converted to DataFrames, and the DataFrames are correct and displayed.	**23 to >19.0 pts Approaching Proficiency** The ETL function does the following: ✓ The three data files are passed into the function. ✓ The Wikipedia JSON file is converted to DataFrame, and the DataFrame is displayed. ✓ The Kaggle metadata and MovieLens ratings data are converted to a DataFrames, but the wrong DataFrames are displayed.	**19 to >16.0 pts Developing Proficiency** The ETL function does the following: ✓ The three data files are passed into the function. ✓ The Wikipedia JSON file is converted to DataFrame, but it is not displayed. ✓ The Kaggle metadata and MovieLens ratings data are converted to DataFrames, but the DataFrames are incorrect.	**16 to >0.0 pts Emerging** The ETL function does the following: ✓ The three data files are passed into the function. ✓ The Wikipedia JSON file is ONLY converted to a DataFrame file. ✓ The Kaggle metadata and MovieLens ratings data are converted to DataFrames, but they are not displayed.	**0 pts Incomplete**	25 pts
Deliverable 2: Extract and Transform the Wikipedia Data	**30 to >27.0 pts Demonstrating Proficiency** ✓ TV shows are filtered out, and the wiki_movies DataFrame is created. ✓ A try-except block is used successfully. ✓ All of the tasks for the extraction & transformation of the Wikipedia data are completed. ✓ The cleaned Wikipedia data is converted to a DataFrame, and the DataFrame is displayed.	**27 to >25.0 pts Approaching Proficiency** ✓ TV shows are filtered out, and the wiki_movies DataFrame is created. ✓A try-except block is used successfully. During the extraction & transformation of the Wikipedia data, the following are done: ✓ Columns with null values are dropped. ✓ The non-null box office data is converted to string values. ✓ Regular expression codes for "form_one" and "form_two" of the box office data are correct. ✓ THREE of the	**25 to >20.0 pts Developing Proficiency** ✓ TV shows are filtered out, and the wiki_movies DataFrame is created. ✓ A try-except block is used successfully. During the extraction & transformation of the Wikipedia data, the following are done: ✓ Columns with null values are dropped. ✓ The non-null box office data is converted to string values. ✓ Regular expression codes for "form_one" and "form_two" of the box office data are correct. ✓ TWO of the	**20 to >0.0 pts Emerging** ✓ TV shows are filtered out, and the wiki_movies DataFrame is created. ✓ A try-except block is written but doesn't catch errors. During the extraction & transformation of the Wikipedia data, the following are done: ✓ Columns with null values are dropped. ✓ The non-null box office data is converted to string values. ✓ Regular expression codes for "form_one" and "form_two" of the box office data are correct. ✓ ONE of the FOUR columns	**0 pts Incomplete**	30 pts

Criteria	Ratings					Pts
Deliverable 3: Extract and Transform the Kaggle Data	**30 to >27.0 pts Demonstrating Proficiency** During the extraction & transformation of the Kaggle metadata, the following are done: ✓ The metadata is cleaned. ✓ The Wikipedia and Kaggle DataFrames are merged ✓ The "movies" DataFrame is created, and all FOUR tasks are performed. ✓ All THREE tasks are completed during the extraction & transformation of the MovieLens rating data. ✓ The Kaggle and ratings DataFrames are correct and displayed.	FOUR columns are cleaned. ✓ Wikipedia data is not cleaned but is converted to a DataFrame and displayed. **27 to >22.0 pts Approaching Proficiency** During the extraction & transformation of the Kaggle metadata, the following are done: ✓ The metadata is cleaned. ✓ The Wikipedia and Kaggle DataFrames are merged ✓ The "movies" DataFrame is created, but only THREE of the FOUR tasks are performed. ✓ All THREE tasks are completed during the extraction & transformation of the MovieLens rating data. ✓ The Kaggle and MovieLens rating DataFrames are displayed, but the "movies" DataFrame is incorrect.	FOUR columns are cleaned. ✓ Wikipedia data is not cleaned but is converted to a DataFrame and displayed. **22 to >16.0 pts Developing Proficiency** During the extraction & transformation of the Kaggle metadata, the following are done: ✓ The metadata is cleaned. ✓ The Wikipedia and Kaggle DataFrames are merged ✓ The "movies" DataFrame is created, but only TWO of the FOUR tasks are performed. During the extraction & transformation of the MovieLens rating data, the following are done: ✓ The ratings counts are cleaned. ✓ The two DataFrames are merged. ✓ There is an attempt to fill the empty values with "0". ✓ The Kaggle and MovieLens rating DataFrames are displayed but incorrect.	is cleaned. ✓ Wikipedia data is not cleaned but is converted to a DataFrame and displayed. **16 to >0.0 pts Emerging** During the extraction & transformation of the Kaggle metadata, the following are done: ✓ The metadata is cleaned. ✓ The Wikipedia and Kaggle DataFrames are merged, but there is an error. ✓ The "movies" DataFrame is created, but only ONE of the FOUR tasks is performed. During the extraction & transformation of the MovieLens rating data, the following are done: ✓ The ratings counts are cleaned. ✓ The two DataFrames are merged, but there is an error ✓ There is an attempt to fill the empty values with "0". ✓ The Kaggle and MovieLens rating DataFrames are	**0 pts Incomplete**	30 pts
Deliverable 4: Create the Movie Database	**15 to >14.0 pts Demonstrating Proficiency** ✓ The data in the movies table in the SQL database is replaced. ✓ The ratings table is dropped, and the rating CSV file is added to the SQL ratings table. ✓ The elapsed time to add the data to the database is displayed.	**14 to >11.0 pts Approaching Proficiency** ✓ The data in the movies table in the SQL database is replaced. ✓ The ratings table is dropped, but not all of the MovieLens rating CSV file is added to the ratings table. ✓ The elapsed time to add the data to the database is displayed.	**11 to >8.0 pts Developing Proficiency** ✓ The data in the movies table in the SQL database is not replaced. ✓ The ratings table is not dropped, but the MovieLens rating CSV file is added to the ratings table. ✓ The elapsed time to add the data to the database is not displayed	**8 to >0.0 pts Emerging** ✓ There is an error adding the movies table in the database. ✓ The ratings table is not dropped, and there is an error adding the MovieLens rating CSV data. ✓ The elapsed time to add the data to the database is partially displayed with an error.	**0 pts Incomplete**	15 pts
						Total Points: 100

displayed but
incorrect.