# BGC Partners - Big Data Coding Exercise

## Data Source

The dataset used in the problem statement was downloaded from IMDB **https://datasets.imdbws.com/.** The webpage had 7 datasets of which only 3 datasets were used according to the problem statement.

Each dataset was contained in a gzipped, tab-seperarted-values (TSV) formatted file in the UTF-8-character set. Below are the information for the datasets used:

**title.basics.tsv.gz** - Contains the following information for titles:

- tconst (string) - alphanumeric unique identifier of the title
- titleType (string) – the type/format of the title (e.g. movie, short, tvseries, tvepisode, video, etc)
- primaryTitle (string) – the more popular title / the title used by the filmmakers on promotional materials at the point of release
- originalTitle (string) - original title, in the original language
- isAdult (boolean) - 0: non-adult title; 1: adult title
- startYear (YYYY) – represents the release year of a title. In the case of TV Series, it is the series start year
- endYear (YYYY) – TV Series end year. '\N' for all other title types
- runtimeMinutes – primary runtime of the title, in minutes
- genres (string array) – includes up to three genres associated with the title

**title.ratings.tsv.gz** – Contains the IMDb rating and votes information for titles

- tconst (string) - alphanumeric unique identifier of the title
- averageRating – weighted average of all the individual user ratings
- numVotes - number of votes the title has received

**name.basics.tsv.gz** – Contains the following information for names:

- nconst (string) - alphanumeric unique identifier of the name/person
- primaryName (string)– name by which the person is most often credited
- birthYear – in YYYY format
- deathYear – in YYYY format if applicable, else '\N'
- primaryProfession (array of strings)– the top-3 professions of the person
- knownForTitles (array of tconsts) – titles the person is known for

## Problem Statement

A spark application was developed to solve the below problems.

**Q1.** Retrieve the top 20 movies with a minimum of 50 votes with the ranking determined by:
(numVotes/averageNumberOfVotes) * averageRating

**Q2.** For these 20 movies, list the persons who are most often credited and list the different titles of the 20 movies.

In order to solve above problems below are the things that were taken care of:

- An efficient and neat code which can solve the problem in less time.
- The code should be reproducible.

Steps performed for solution 1:

- title_basics dataset was filtered with movieType = 'Movie' so that records can be reduced for further computation since we are only interested in movies.
- Title_basics dataset was joined title_ratings dataset to gather the information regarding ratings and votes of each move. To increase the efficiency broadcast join was used but since the application ran on Databricks community edition and there were no workers it won't show the effect. However, it was implemented in the code.
- Average of number of votes was calculated in title_ratings dataset and was stored in avgNumVotes variable.
- Rank column was created which stored the rank of all the movies whose number of votes > 5 and top 20 movies were retrieved with below calculation :
- The solution took around 40 secs to show the results starting from the data import phase.

Rank = (numVotes/averageNumberOfVotes) * averageRating

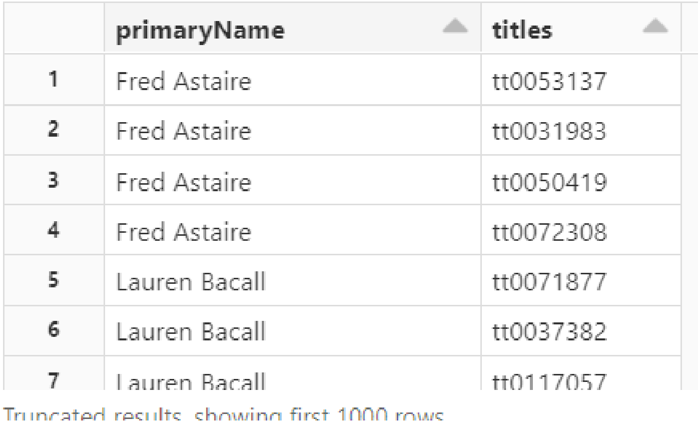
Below are top 20 movies retrieved:

| originalTitle |
|---|
| The Shawshank Redemption |
| The Dark Knight |
| Inception |
| Fight Club |
| Forrest Gump |
| Pulp Fiction |
| The Godfather |
| The Matrix |
| The Lord of the Rings: The Return of the King |
| The Lord of the Rings: The Fellowship of the Ring |
| Interstellar |
| The Lord of the Rings: The Two Towers |
| The Dark Knight Rises |
| Se7en |
| Django Unchained |
| Gladiator |
| The Silence of the Lambs |
| Schindler's List |

| |
|---|
| Batman Begins |
| Inglourious Basterds |

Steps for solution 2 :

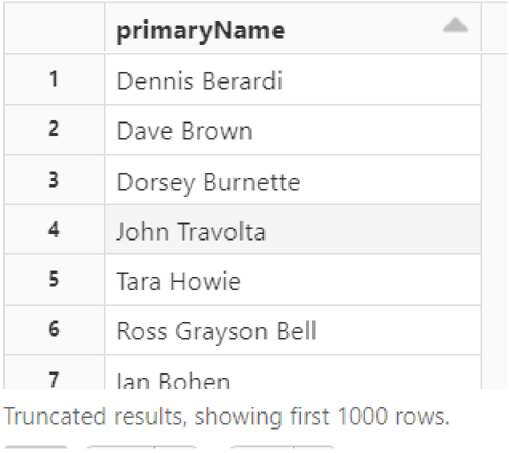
- Retrieve the titles for which the person is known for. Here in name_basics dataset, knownForTitles contains an array of titles, so we can't use it directly. So, all the titles were split and exploded into the new column titles for each person in persons dataframe.



| | primaryName | titles |
|---|---|---|
| 1 | Fred Astaire | tt0053137 |
| 2 | Fred Astaire | tt0031983 |
| 3 | Fred Astaire | tt0050419 |
| 4 | Fred Astaire | tt0072308 |
| 5 | Lauren Bacall | tt0071877 |
| 6 | Lauren Bacall | tt0037382 |
| 7 | Lauren Bacall | tt0117057 |

Truncated results, showing first 1000 rows.

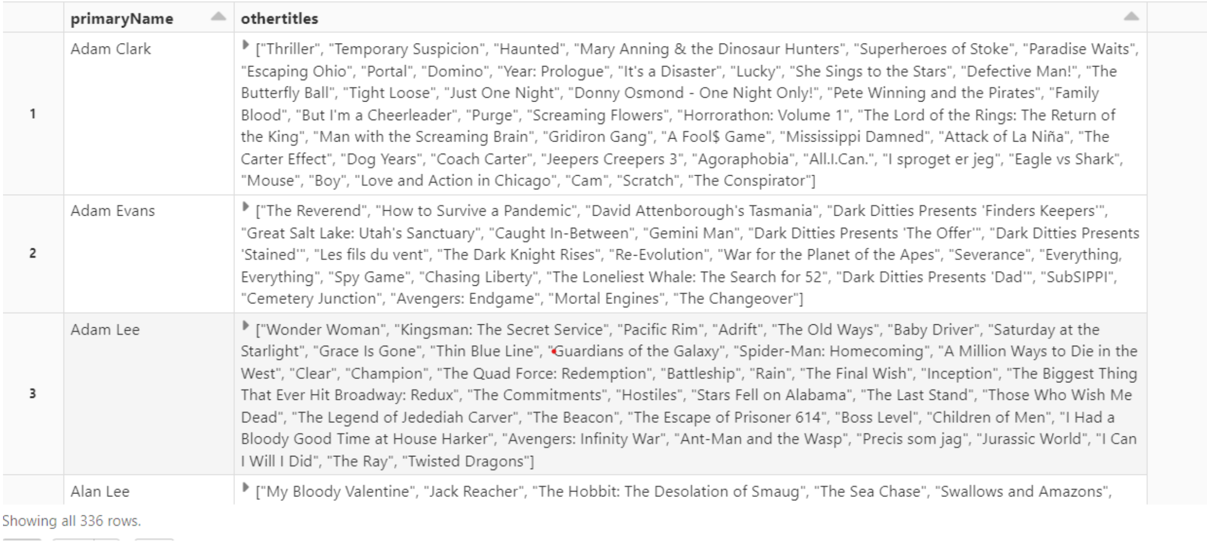
- Distinct primaryNames of persons were retrieved for top 20 movies by joining persons and averageRatingMovies_R(contains information of 20 movies) dataframes



| | primaryName |
|---|---|
| 1 | Dennis Berardi |
| 2 | Dave Brown |
| 3 | Dorsey Burnette |
| 4 | John Travolta |
| 5 | Tara Howie |
| 6 | Ross Grayson Bell |
| 7 | Ian Bohen |

Truncated results, showing first 1000 rows.

- Retrieve all the original titles of the movies for which the persons have worked in top 20 movies. This task was performed by joining persons and pers_workedontop20m(contains list of names of persons who worked in top 20 movies) dataframe  through which we will be able to fetch the information about the persons and then joining it with movies dataframe in order to fetch titles in which the persons in top 20 movies have worked.

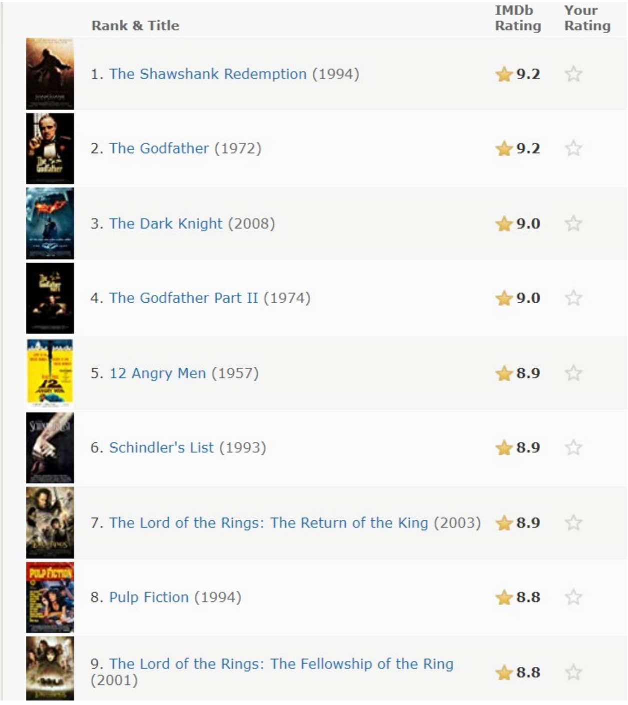


| | primaryName | othertitles |
|---|---|---|
| 1 | Adam Clark | ▸ ["Thriller", "Temporary Suspicion", "Haunted", "Mary Anning & the Dinosaur Hunters", "Superheroes of Stoke", "Paradise Waits", "Escaping Ohio", "Portal", "Domino", "Year: Prologue", "It's a Disaster", "Lucky", "She Sings to the Stars", "Defective Man!", "The Butterfly Ball", "Tight Loose", "Just One Night", "Donny Osmond - One Night Only!", "Pete Winning and the Pirates", "Family Blood", "But I'm a Cheerleader", "Purge", "Screaming Flowers", "Horrorathon: Volume 1", "The Lord of the Rings: The Return of the King", "Man with the Screaming Brain", "Gridiron Gang", "A Fool$ Game", "Mississippi Damned", "Attack of La Niña", "The Carter Effect", "Dog Years", "Coach Carter", "Jeepers Creepers 3", "Agoraphobia", "All.I.Can.", "I sproget er jeg", "Eagle vs Shark", "Mouse", "Boy", "Love and Action in Chicago", "Cam", "Scratch", "The Conspirator"] |
| 2 | Adam Evans | ▸ ["The Reverend", "How to Survive a Pandemic", "David Attenborough's Tasmania", "Dark Ditties Presents 'Finders Keepers'", "Great Salt Lake: Utah's Sanctuary", "Caught In-Between", "Gemini Man", "Dark Ditties Presents 'The Offer'", "Dark Ditties Presents 'Stained'", "Les fils du vent", "The Dark Knight Rises", "Re-Evolution", "War for the Planet of the Apes", "Severance", "Everything, Everything", "Spy Game", "Chasing Liberty", "The Loneliest Whale: The Search for 52", "Dark Ditties Presents 'Dad'", "SubSIPPI", "Cemetery Junction", "Avengers: Endgame", "Mortal Engines", "The Changeover"] |
| 3 | Adam Lee | ▸ ["Wonder Woman", "Kingsman: The Secret Service", "Pacific Rim", "Adrift", "The Old Ways", "Baby Driver", "Saturday at the Starlight", "Grace Is Gone", "Thin Blue Line", "Guardians of the Galaxy", "Spider-Man: Homecoming", "A Million Ways to Die in the West", "Clear", "Champion", "The Quad Force: Redemption", "Battleship", "Rain", "The Final Wish", "Inception", "The Biggest Thing That Ever Hit Broadway: Redux", "The Commitments", "Hostiles", "Stars Fell on Alabama", "The Last Stand", "Those Who Wish Me Dead", "The Legend of Jedediah Carver", "The Beacon", "The Escape of Prisoner 614", "Boss Level", "Children of Men", "I Had a Bloody Good Time at House Harker", "Avengers: Infinity War", "Ant-Man and the Wasp", "Precis som jag", "Jurassic World", "I Can I Will I Did", "The Ray", "Twisted Dragons"] |
| | Alan Lee | ▸ ["My Bloody Valentine", "Jack Reacher", "The Hobbit: The Desolation of Smaug", "The Sea Chase", "Swallows and Amazons", |

Showing all 336 rows.

## Test Evidence

For test evidence of the top 20 movies, it was being verified on IMDB website that the movies accumulated were one of highest rated movies produced with votes and rating. For instance in below picture, we can see that movies like The Shawshank Redemption, The Dark Knight, The Godfather part II are also in our top rank list.



| Rank & Title | IMDb Rating | Your Rating |
|---|---|---|
| 1. The Shawshank Redemption (1994) | ★9.2 | ☆ |
| 2. The Godfather (1972) | ★9.2 | ☆ |
| 3. The Dark Knight (2008) | ★9.0 | ☆ |
| 4. The Godfather Part II (1974) | ★9.0 | ☆ |
| 5. 12 Angry Men (1957) | ★8.9 | ☆ |
| 6. Schindler's List (1993) | ★8.9 | ☆ |
| 7. The Lord of the Rings: The Return of the King (2003) | ★8.9 | ☆ |
| 8. Pulp Fiction (1994) | ★8.8 | ☆ |
| 9. The Lord of the Rings: The Fellowship of the Ring (2001) | ★8.8 | ☆ |

Below are the test cases which were performed on the spark application.

| Test case Id | Test case description | Expected result | Actual Result | Outcome |
|---|---|---|---|---|
| 1 | To validate whether the results contain only titleType='movie' | dataframe Top20Movies should contain result which was having titleType = 'Movie' | Results accumulated with only titleType='movie' | Passed |
| 2 | To validate whether Dataframe contains Movie name with count = 20 | Count of dataframe Top20Movies should be 20 and it should contain movies with highest rating | Count dataframe Top20Movies is 20 and highest rating movies like The Shawshank Redemption, The Dark Knight etc., were present in the results | Passed |
| 3 | To validate after the split of titles in name_bsics dataframe, the primaryName has been populated with correct titles | name_bsics dataframe should be populated with primaryNames with respective titles | name_bsics dataframe was populated with respective titles and primaryName | Passed |
| 4 | To validate the titles of Primarynames which they know for are valid | moviesof_top_20_creditedP dataframe should contain valid list of titles | Result contains valid list of titles with respect to primaryNames | Passed |