Extract:

We chose to create a project based on two book-related datasets available publicly on kaggle. We thought these two datasets would create an interesting database when merged together because goodreads ratings are a well-trusted source of feedback on books and authors and it would be interesting to do an analysis comparing number of comments or rating of books/authors to a book's longevity on the NYT bestseller list. We also chose to include some other data we thought might be juicy, such as the Author's gender and book publication dates.

Datasets:

New York Times best sellers

https://www.kaggle.com/cmenca/new-york-times-hardcover-fiction-best-sellers

Description: (from Kaggle)

> "Gathered from the New York Times API for Hardcover Fiction best sellers from June 7, 2008 to July 22, 2018
>
> The API can be found here: https://developer.nytimes.com/
>
> Collected data includes the book title, author, the date of the best seller list, the published date of the list, the book description, the rank (this week and last week), the publisher, number of weeks on the list, and the price."

Columns selected: title, author, published date, bestsellers date

Goodreads books/author data

https://www.kaggle.com/brosen255/goodreads-books

Description: (from Kaggle): "Web Scraped 20K books found on Goodreads' most popular lists"

Since this dataset contained both books and author information, we used it to create two tables. One for Books and one for Authors

Columns in Books table:

book_title, author_name, genre_1, genre_2, book_avg_rating, num_ratings, num_reviews, pages, publish_date

Columns in Authors table:

Author_name, author_gender, author_average_rating, author_rating_count, birthplace

Transform:

Major Hurdles:

- (Goodreads) - many rows had missing publication date, but that was the only detected “null” in the dataset that dropna would get rid of.
- (Goodreads)- the data frame was littered with “/n”, and these were easily found and disposed of.
- (Goodreads) the non-null dates were in all different formats, so we used reg_ex to find those which contained four integer strings which began with 0,1, or 2. We then used the regex, group function to only keep this portion of the date. On rows where we could not find a string of this description, we set the value to Nan, so that we could drop that row when we ran dropna() again.
- (NYT) Dates : converted from unix timestamp then extracted year only
- (NYT + Goodreads)The book titles on the NYT list were in all caps, so we converted the titles on the goodreads list to all caps as well.
- Some book titles on the goodreads list were in non UTF-8 characters (arabic), so we dropped all columns which caused this problem in SQL.
- We standardised table names so they would be more easily read and used and connect our tables more obviously
- We decided to add authors that appear in the NYT list but not in the goodreads-author table if they were not already there. To account for the table values that NYT did not provide, we just set them as null.

Load:

- Named the database as ‘goodbook_db’
- Created 3 tables :
    - author (has information for both NYT and goodreads)
        - id
        - name
        - Gender - if doesn't exist, null
        - Ratings - if doesn't exist, null
        - ratings count - if doesn't exist, null
    - nyt_books
        - id
        - title
        - author
        - published year
        - bestseller year
        - rank
    - goodreads_books

- id
- title
- author
- published date

Things we could have done:

One of the major limitations of our project was the scope of the New york times dataset. There are many new york times best seller lists, all accessible through the free NYT API. If we had more time we would have used many of these lists from a larger date range instead of just “Hardcover Fiction Best Sellers from 2008 to 2018”.

In this same vein, if we really wanted more good reads data we would have scraped the site, not only the most popular lists.