# PROJECT STAGE 4

Team – 29

Our entities were books. The two tables contain the following attributes/columns:

- title
- author
- rating
- format

'*title'* attribute was merged by choosing the one with larger length among the two schemas. This was done assuming larger length titles have more information.

'*author'* column was merged by choosing the one with smaller length. In case of multiple authors, one of the schemas have an 'and' string at the end of the author's name. The other schema did not have this problem. Hence, choosing the author's name of smaller length was logical.

*'rating'* attribute was merged by taking the average of the two. In case of missing value, we select it from the other schema.

*'format'* attribute was merged by taking the value from the first schema, TableA. This was due to the fact that this information was more reliable in the first schema than second.

## Statistics on Table E

For the purpose of analysis, we added a new column to the merged table by web page crawling to get the *year* in which the book was published.

Schema of Table E -

- title
- author
- rating
- format
- year

Number of tuples in Table E: 274

Sample tuples from Table E -

| Title | Author | Rating | Format | Year |
|---|---|---|---|---|
| To Kill a Mockingbird | Harper Lee | 4.48 | Paperback | 2006 |
| The Absolutely True Diary of a Part-Time Indian | Sherman Alexie | 4.36 | Hardcover | 2007 |
| The Very Hungry Caterpillar | Eric Carle | 4.45 | Board book | 1994 |
| Sapiens: A Brief History of Humankind | Yuval Noah Harari | 4.52 | Paperback | 2014 |

Problems encountered while merging:

The tables seemed to have some duplicate entries mainly because the same book title was given in many formats.

Eg:

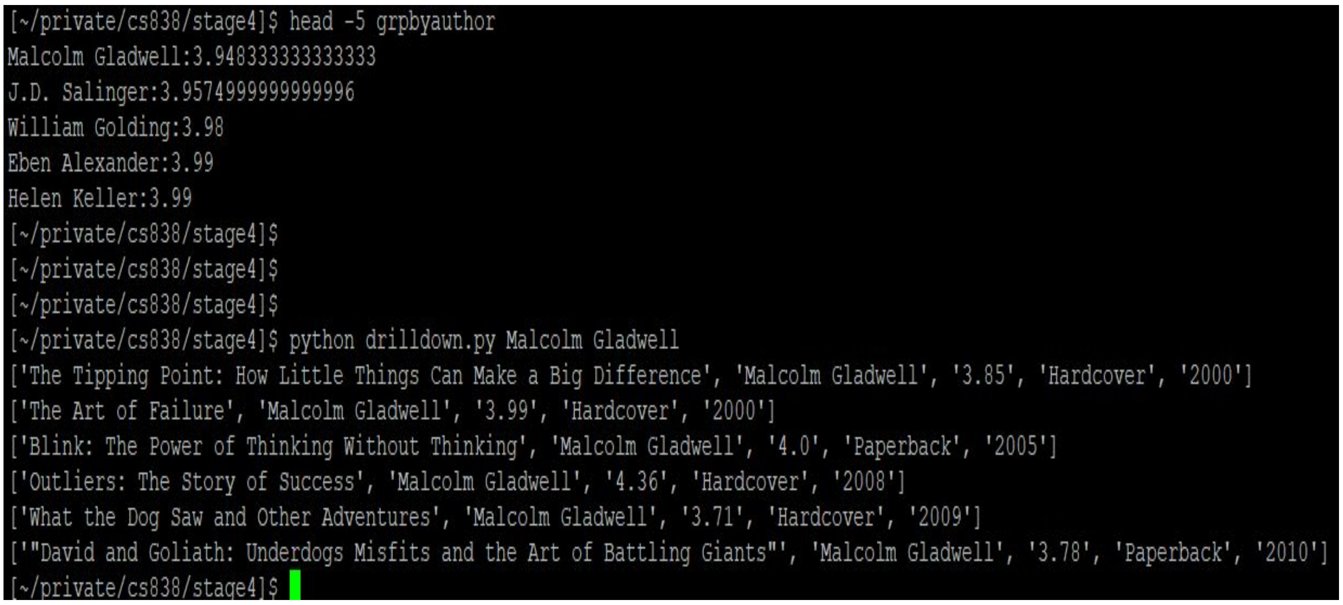
A Wrinkle in Time (Time Quintet, #1)

The Wrinkle in Time Quintet - Digest Size Boxed Set (Time Quintet, #1-5)

This case was handled during analysis with a special case in the python script.

## Data Analysis and Observations

We did OLAP analysis to see the progress of authors over years. A summary of the steps is shown in the screenshot below:

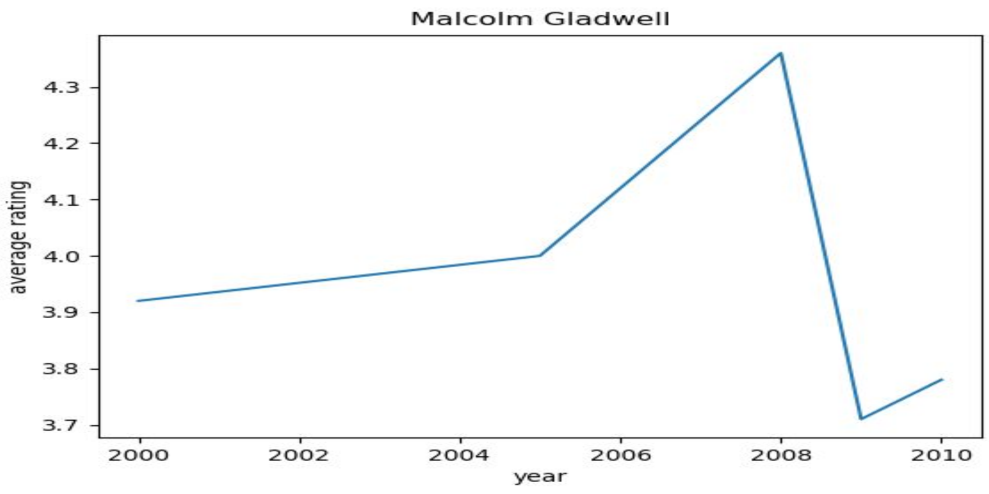


```
[~/private/cs838/stage4]$ head -5 grpbyauthor
Malcolm Gladwell:3.948333333333333
J.D. Salinger:3.9574999999999996
William Golding:3.98
Eben Alexander:3.99
Helen Keller:3.99
[~/private/cs838/stage4]$
[~/private/cs838/stage4]$
[~/private/cs838/stage4]$
[~/private/cs838/stage4]$ python drilldown.py Malcolm Gladwell
['The Tipping Point: How Little Things Can Make a Big Difference', 'Malcolm Gladwell', '3.85', 'Hardcover', '2000']
['The Art of Failure', 'Malcolm Gladwell', '3.99', 'Hardcover', '2000']
['Blink: The Power of Thinking Without Thinking', 'Malcolm Gladwell', '4.0', 'Paperback', '2005']
['Outliers: The Story of Success', 'Malcolm Gladwell', '4.36', 'Hardcover', '2008']
['What the Dog Saw and Other Adventures', 'Malcolm Gladwell', '3.71', 'Hardcover', '2009']
['"David and Goliath: Underdogs Misfits and the Art of Battling Giants"', 'Malcolm Gladwell', '3.78', 'Paperback', '2010']
[~/private/cs838/stage4]$
```

Steps:

1. We first did a groupby operation on Authors and got the average ratings for each author over all the books authored by him/her.
2. From the results of the above Roll-up operation, we took the author with the minimum rating - ***Malcolm Gladwell***
3. For this author, we tried to drill down to know the rating over years.
4. This showed that the author has written a highly rated excellent book - ***Outliers: The Story of Success.*** The average rating of the author was not good enough (nowhere to be listed in the top-rated authors in Goodread) because of the other below average books written by him.

5. The graph shows that the author started off good and showed improvements in writing but took a sudden fall because of the book - ***What the Dog Saw and Other Adventures.***



Problems during Data analysis:

1. The choice of attribute for OLAP style exploration was tricky.
2. It was difficult to choose which technique to use for data analysis.
3. Making appropriate inference was also difficult.

Future Work:

Extracting the book reviews from the web sources to do sentiment analysis. The accuracy of the classifier could be evaluated using the 'rating' attribute in our dataset.

Code of the Python script (that merges the tables):

```
import csv

import re



matches_path = "Labelled Set G.csv"

matches_new_path = "matches.csv"

path_A = "TableA.csv"

path_B = "TableB.csv"

mergedFilePath = "Table E.csv"
```

```
matches_new = open(matches_new_path, "w", encoding='utf-8')
matches_new.write("tableA_title, tableA_author, tableB_title, tableB_author\n")
TableE = open(mergedFilePath, "w", encoding='utf-8')
TableE.write("title, author, rating, format\n")

# Open TableA and store in the form of dict for later traversal
tableA = {}
with open(path_A, newline='', encoding='utf-8') as csvfile:
    rows = csv.reader(csvfile, delimiter=',')
    # Skips header
    next(rows)
    # Forms the dict
    for row in rows:
        tableA[row[0]] = [row[1], row[2], row[3], row[4]]

# Open TableB and store in the form of dict for later traversal
tableB = {}
with open(path_B, newline='', encoding='utf-8') as csvfile:
    rows = csv.reader(csvfile, delimiter=',')
    # Skips header
    next(rows)
    # Forms the dict
    for row in rows:
        tableB[row[0]] = [row[1], row[2], row[3], row[4]]

'''
```

```
For each row in the Matches.csv file, go to the corresponding entries in TableA and TableB.

Then merge these two based on certain rules for each column.

Schema of TableE is same as TableA. And schema of TableA = schema of TableB

'''

with open(matches_path, newline='', encoding='utf-8') as csvfile:

    tuplePairs = csv.reader(csvfile, delimiter=',')

    # Skips header

    next(tuplePairs)

    for row in tuplePairs:

        # If label attribute of the row is 1, then it is a correct match and merge this entry in the new TableE

        if (int(row[8]) == 1):


            # Write this matching entity in a new table. (For submission only)

            matches_new.write(row[4] + ',' + row[5] + ',' + row[6] + ',' + row[7] + '\n')


            dataA = tableA[row[2]]

            dataB = tableB[row[3]]


            # Selecting title based on length. Choosing the bigger length title.

            if (len(dataA[0]) >= len(dataB[0])):

                title = dataA[0]

            else:

                title = dataB[0]

            title = re.sub('[,]', '', title)


            # Selecting author based on length. Choosing the one with smaller length.
```

```
if (len(dataA[1]) <= len(dataB[1]) or len(dataB[1]) == 0):
   author = dataA[1]
else:
   author = dataB[1]

# Selecting the rating based on average of the two.
# Few values of rating are empty in TableB.
if (dataB[2] == ''):
   rating = float(dataA[2])
else:
    rating = (float(dataA[2]) + float(dataB[2]))/2

# Selecting the format based on TableA as it is more reliable source than TableB.
format = dataA[3]

# Writing to TableE
TableE.write(title + ',' + author + ',' + '{0:.2f}'.format(rating) + ',' + format + '\n')
```