FINAL PROJECT- ENGINEERING OF BIG DATA SYSTEMS

GitHub Source code: https://github.com/sirijami/BookCrossingProject

Dataset: **Book-crossing** dataset
http://www2.informatik.uni-freiburg.de/~cziegler/BX/

It includes 3 csv files:

BX-Users

- Contains the users. Note that user IDs (`*User-ID*`) have been anonymized and map to integers. Demographic data is provided (`*Location*`, `*Age*`) if available. Otherwise, these fields contain *NULL*-values.

BX-Books

- Books are identified by their respective ISBN. Invalid ISBNs have already been removed from the dataset. Moreover, some content-based information is given (`*Book-Title*`, `*Book-Author*`, `*Year-Of-Publication*`, `*Publisher*`), obtained from Amazon Web Services. Note that in case of several authors, only the first is provided. URLs linking to cover images are also given, appearing in three different flavours (`*Image-URL-S*`, `*Image-URL-M*`, `*Image-URL-L*`), i.e., small, medium, large. These URLs point to the Amazon web site.

BX-Book-Ratings

- Contains the book rating information. Ratings (`*Book-Rating*`) are either explicit, expressed on a scale from 1-10 (higher values denoting higher appreciation), or implicit, expressed by 0.

Glimpse of Dataset:

Book Ratings

```
"User-ID";"ISBN";"Book-Rating"
"276725";"034545104X";"0"
"276726";"0155061224";"5"
"276727";"0446520802";"0"
"276729";"052165615X";"3"
"276729";"0521795028";"6"
"276733";"2080674722";"0"
"276736";"3257224281";"8"
"276737";"0600570967";"6"
"276744";"038550120X";"7"
"276745";"342310538";"10"
"276746";"0425115801";"0"
"276746";"0449006522";"0"
"276746";"0553561618";"0"
"276746";"055356451X";"0"
"276746";"0786013990";"0"
"276746";"0786014512";"0"
"276747";"0060517794";"9"
"276747";"0451192001";"0"
"276747";"0609801279";"0"
"276747";"0671537458";"9"
"276747";"0679776818";"8"
"276747";"0943066433";"7"
"276747";"1570231028";"0"
"276747";"1885408226";"7"
"276748";"0747558167";"6"
"276748";"3442437407";"0"
"276751";"033390804X";"0"
"276751";"3596218098";"8"
"276754";"0684867621";"8"
"276755";"0451166892";"5"
"276760";"8440682697";"10"
"276762";"034544003X";"0"
"276762";"0380000059";"0"
"276762";"0380711524";"5"
"276762";"0451167317";"0"
"276762";"0451454952";"0"
```

Books:

BX-Books.csv

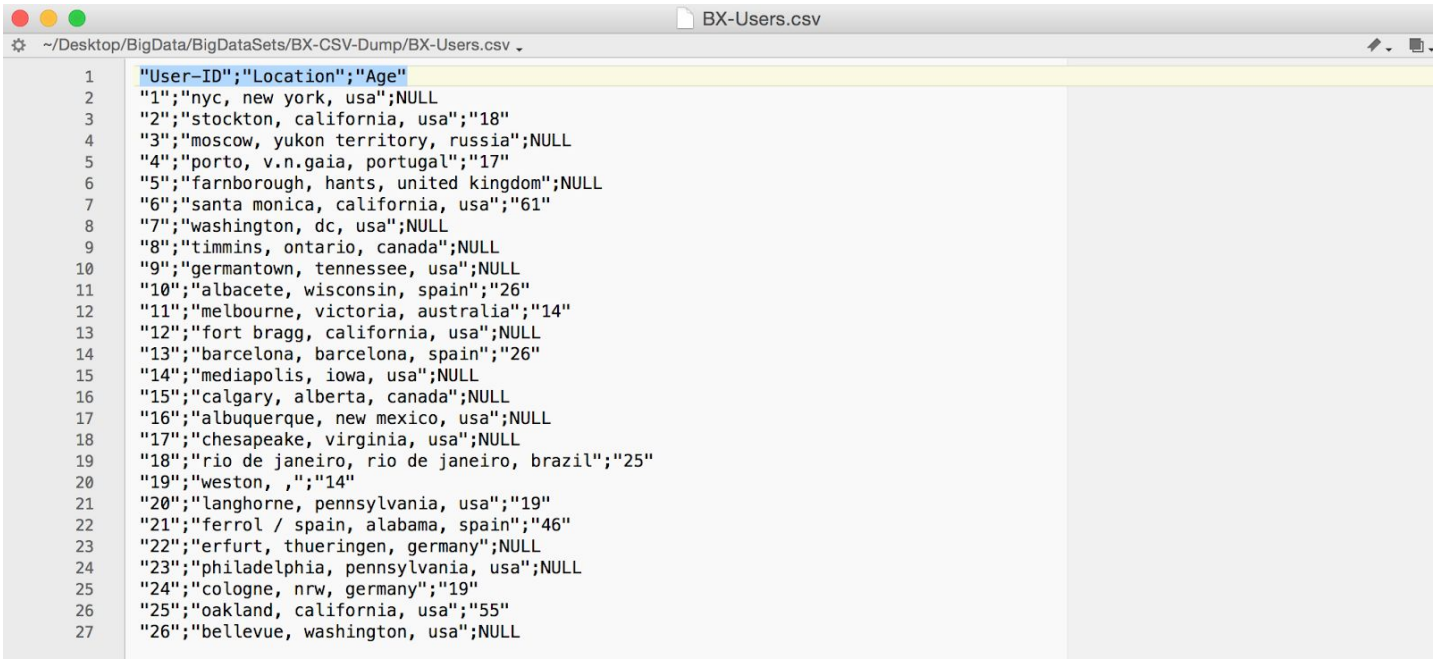
~/Desktop/BigData/BigDataSets/BX-CSV-Dump/BX-Books.csv

```
"ISBN";"Book-Title";"Book-Author";"Year-Of-Publication";"Publisher";"Image-URL-S";"Image-URL-M";"Image-URL-L"
"0195153448";"Classical Mythology";"Mark P. O. Morford";"2002";"Oxford University Press";"http://images.amazon.com/images/P/0195153448.01.THUMBZZZ.jpg";"http:/
"0002005018";"Clara Callan";"Richard Bruce Wright";"2001";"HarperFlamingo Canada";"http://images.amazon.com/images/P/0002005018.01.THUMBZZZ.jpg";"http://images
"0060973129";"Decision in Normandy";"Carlo D'Este";"1991";"HarperPerennial";"http://images.amazon.com/images/P/0060973129.01.THUMBZZZ.jpg";"http://images.amazc
"0374157065";"Flu: The Story of the Great Influenza Pandemic of 1918 and the Search for the Virus That Caused It";"Gina Bari Kolata";"1999";"Farrar Straus Girc
"0393045218";"The Mummies of Urumchi";"E. J. W. Barber";"1999";"W. W. Norton &amp; Company";"http://images.amazon.com/images/P/0393045218.01.THUMBZZZ.jpg";"htt
"0399135782";"The Kitchen God's Wife";"Amy Tan";"1991";"Putnam Pub Group";"http://images.amazon.com/images/P/0399135782.01.THUMBZZZ.jpg";"http://images.amazon.
"0425176428";"What If?: The World's Foremost Military Historians Imagine What Might Have Been";"Robert Cowley";"2000";"Berkley Publishing Group";"http://images
"0671870432";"PLEADING GUILTY";"Scott Turow";"1993";"Audioworks";"http://images.amazon.com/images/P/0671870432.01.THUMBZZZ.jpg";"http://images.amazon.com/image
"0679425608";"Under the Black Flag: The Romance and the Reality of Life Among the Pirates";"David Cordingly";"1996";"Random House";"http://images.amazon.com/im
"074322678X";"Where You'll Find Me: And Other Stories";"Ann Beattie";"2002";"Scribner";"http://images.amazon.com/images/P/074322678X.01.THUMBZZZ.jpg";"http://i
"0771074670";"Nights Below Station Street";"David Adams Richards";"1988";"Emblem Editions";"http://images.amazon.com/images/P/0771074670.01.THUMBZZZ.jpg";"http
"080652121X";"Hitler's Secret Bankers: The Myth of Swiss Neutrality During the Holocaust";"Adam Lebor";"2000";"Citadel Press";"http://images.amazon.com/images/
"0887841740";"The Middle Stories";"Sheila Heti";"2004";"House of Anansi Press";"http://images.amazon.com/images/P/0887841740.01.THUMBZZZ.jpg";"http://images.am
"1552041778";"Jane Doe";"R. J. Kaiser";"1999";"Mira Books";"http://images.amazon.com/images/P/1552041778.01.THUMBZZZ.jpg";"http://images.amazon.com/images/P/15
"1558746218";"A Second Chicken Soup for the Woman's Soul (Chicken Soup for the Soul Series)";"Jack Canfield";"1998";"Health Communications";"http://images.amaz
"1567407781";"The Witchfinder (Amos Walker Mystery Series)";"Loren D. Estleman";"1998";"Brilliance Audio - Trade";"http://images.amazon.com/images/P/1567407781
"1575663937";"More Cunning Than Man: A Social History of Rats and Man";"Robert Hendrickson";"1999";"Kensington Publishing Corp.";"http://images.amazon.com/imag
"1881320189";"Goodbye to the Buttermilk Sky";"Julia Oliver";"1994";"River City Pub";"http://images.amazon.com/images/P/1881320189.01.THUMBZZZ.jpg";"http://imag
"0440234743";"The Testament";"John Grisham";"1999";"Dell";"http://images.amazon.com/images/P/0440234743.01.THUMBZZZ.jpg";"http://images.amazon.com/images/P/044
"0452264464";"Beloved (Plume Contemporary Fiction)";"Toni Morrison";"1994";"Plume";"http://images.amazon.com/images/P/0452264464.01.THUMBZZZ.jpg";"http://image
"0609804618";"Our Dumb Century: The Onion Presents 100 Years of Headlines from America's Finest News Source";"The Onion";"1999";"Three Rivers Press";"http://im
"1841721522";"New Vegetarian: Bold and Beautiful Recipes for Every Occasion";"Celia Brooks Brown";"2001";"Ryland Peters &amp; Small Ltd";"http://images.amazon.
"1879384493";"If I'd Known Then What I Know Now: Why Not Learn from the Mistakes of Others? : You Can't Afford to Make Them All Yourself";"J. R. Parrish";"200
"0061076031";"Mary-Kate &amp; Ashley Switching Goals (Mary-Kate and Ashley Starring in)";"Mary-Kate &amp; Ashley Olsen";"2000";"HarperEntertainment";"http://im
"0439095026";"Tell Me This Isn't Happening";"Robynn Clairday";"1999";"Scholastic";"http://images.amazon.com/images/P/0439095026.01.THUMBZZZ.jpg";"http://images
"0689821166";"Flood : Mississippi 1927";"Kathleen Duey";"1998";"Aladdin";"http://images.amazon.com/images/P/0689821166.01.THUMBZZZ.jpg";"http://images.amazon.c
"0971880107";"Wild Animus";"Rich Shapero";"2004";"Too Far";"http://images.amazon.com/images/P/0971880107.01.THUMBZZZ.jpg";"http://images.amazon.com/images/P/09
"0345402871";"Airframe";"Michael Crichton";"1997";"Ballantine Books";"http://images.amazon.com/images/P/0345402871.01.THUMBZZZ.jpg";"http://images.amazon.com/i
"0345417623";"Timeline";"MICHAEL CRICHTON";"2000";"Ballantine Books";"http://images.amazon.com/images/P/0345417623.01.THUMBZZZ.jpg";"http://images.amazon.com/i
"0684823802";"OUT OF THE SILENT PLANET";"C.S. Lewis";"1996";"Scribner";"http://images.amazon.com/images/P/0684823802.01.THUMBZZZ.jpg";"http://images.amazon.com
"0375759778";"Prague : A Novel";"ARTHUR PHILLIPS";"2003";"Random House Trade Paperbacks";"http://images.amazon.com/images/P/0375759778.01.THUMBZZZ.jpg";"http:/
"0425163091";"Chocolate Jesus";"Stephan Jaramillo";"1998";"Berkley Publishing Group";"http://images.amazon.com/images/P/0425163091.01.THUMBZZZ.jpg";"http://ima
"3404921038";"Wie Barney es sieht.";"Mordecai Richler";"2002";"L√?¬ºbbe";"http://images.amazon.com/images/P/3404921038.01.THUMBZZZ.jpg";"http://images.amazon.c
"3442353866";"Der Fluch der Kaiserin. Ein Richter- Di- Roman.";"Eleanor Cooney";"2001";"Goldmann";"http://images.amazon.com/images/P/3442353866.01.THUMBZZZ.jpg
"3442410665";"Sturmzeit. Roman.";"Charlotte Link";"1991";"Goldmann";"http://images.amazon.com/images/P/3442410665.01.THUMBZZZ.jpg";"http://images.amazon.com/im
"3442446937";"Tage der Unschuld.";"Richard North Patterson";"2000";"Goldmann";"http://images.amazon.com/images/P/3442446937.01.THUMBZZZ.jpg";"http://images.ama
"0375406328";"Lying Awake";"Mark Salzman";"2000";"Alfred A. Knopf";"http://images.amazon.com/images/P/0375406328.01.THUMBZZZ.jpg";"http://images.amazon.com/ima
"0446310786";"To Kill a Mockingbird";"Harper Lee";"1988";"Little Brown &amp; Company";"http://images.amazon.com/images/P/0446310786.01.THUMBZZZ.jpg";"http://im
```

Users

BX-Users.csv

~/Desktop/BigData/BigDataSets/BX-CSV-Dump/BX-Users.csv



```
"User-ID";"Location";"Age"
"1";"nyc, new york, usa";NULL
"2";"stockton, california, usa";"18"
"3";"moscow, yukon territory, russia";NULL
"4";"porto, v.n.gaia, portugal";"17"
"5";"farnborough, hants, united kingdom";NULL
"6";"santa monica, california, usa";"61"
"7";"washington, dc, usa";NULL
"8";"timmins, ontario, canada";NULL
"9";"germantown, tennessee, usa";NULL
"10";"albacete, wisconsin, spain";"26"
"11";"melbourne, victoria, australia";"14"
"12";"fort bragg, california, usa";NULL
"13";"barcelona, barcelona, spain";"26"
"14";"mediapolis, iowa, usa";NULL
"15";"calgary, alberta, canada";NULL
"16";"albuquerque, new mexico, usa";NULL
"17";"chesapeake, virginia, usa";NULL
"18";"rio de janeiro, rio de janeiro, brazil";"25"
"19";"weston, ,";"14"
"20";"langhorne, pennsylvania, usa";"19"
"21";"ferrol / spain, alabama, spain";"46"
"22";"erfurt, thueringen, germany";NULL
"23";"philadelphia, pennsylvania, usa";NULL
"24";"cologne, nrw, germany";"19"
"25";"oakland, california, usa";"55"
"26";"bellevue, washington, usa";NULL
```

Assumption:

- In users file, it contains list of the user who purchased books.
- Rating file include users who have purchased and also given ratings

Relation between the files:

- Book Ratings with user via user-id
- Book Ratings with books via ISBN

Used: Apache hadoop Ecosystem

- Hadoop HDFS (To store the data)
- Hadoop MapReduce (To perform analysis)
- Apache Pig (High level data flow language, to perform analysis much faster)
- Apache Hive (Data Warehouse)
- Apache Mahout(Collaborative filtering-recommendation)

Analysis:

- Frequency of book published each year:(MapReduce)
- Find the list of users who have most active and sent maximum number of ratings.(MapReduce)
- Average Rating of each book (ISBN is unique book id)- (MapReduce)
- Find the book details with meaningful information like ‘Book id, Book title and average rating’(Pig)
- Display top-ten list of books with highest ratings.(Pig)
- Display list of all books published from year 2001 - 2008 and sort the books by year published.(Pig)
- Find all unique list of books with book id and book name.(Pig)
- Display number of books each author published.(case-sensitive, but this sometimes leads to wrong analysis)(pig)
- Used LOWER function to first convert and then group by name to get proper results.(Pig)
- Find the users who bought the books but not yet rated any book. So that we can send notifications to them to ask to rate.(Pig)
- Find the behaviour of users who have given ratings.(Pig)
- Give the list of user in sorted order in user-id.(Hive)
- Find the user with age above 20.(Hive)
- Find users who purchase books are mostly from which location.(Hive)
- Recommend user a book based on his past purchase history.(Mahout)
- User based recommendation using another similarity.(Mahout)
- User based recommendation based on Euclidean similarity(Mahout)

Getting started:

Load the data to hdfs:

```
Sirishas-MacBook-Pro:bin sirishaepari$ hadoop fs -mkdir /BigDataProject
2018-04-16 21:15:39,064 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where a
pplicable
Sirishas-MacBook-Pro:bin sirishaepari$ hadoop fs -mkdir /BigDataProject/Dataset
2018-04-16 21:15:54,708 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where a
pplicable
Sirishas-MacBook-Pro:bin sirishaepari$ hadoop fs -mkdir /BigDataProject/OutputData
2018-04-16 21:16:09,914 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where a
pplicable
Sirishas-MacBook-Pro:bin sirishaepari$ hadoop fs -ls /BigDataProject
2018-04-16 21:16:33,655 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where a
pplicable
Found 2 items
drwxr-xr-x   - sirishaepari supergroup          0 2018-04-16 21:15 /BigDataProject/Dataset
drwxr-xr-x   - sirishaepari supergroup          0 2018-04-16 21:16 /BigDataProject/OutputData
```

```
Sirishas-MacBook-Pro:bin sirishaepari$ hadoop fs -copyFromLocal /Users/sirishaepari/Desktop/BigData/BX-CSV-Dump /BigDataProject/Dataset
2018-04-16 21:17:53,377 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where a
pplicable
```

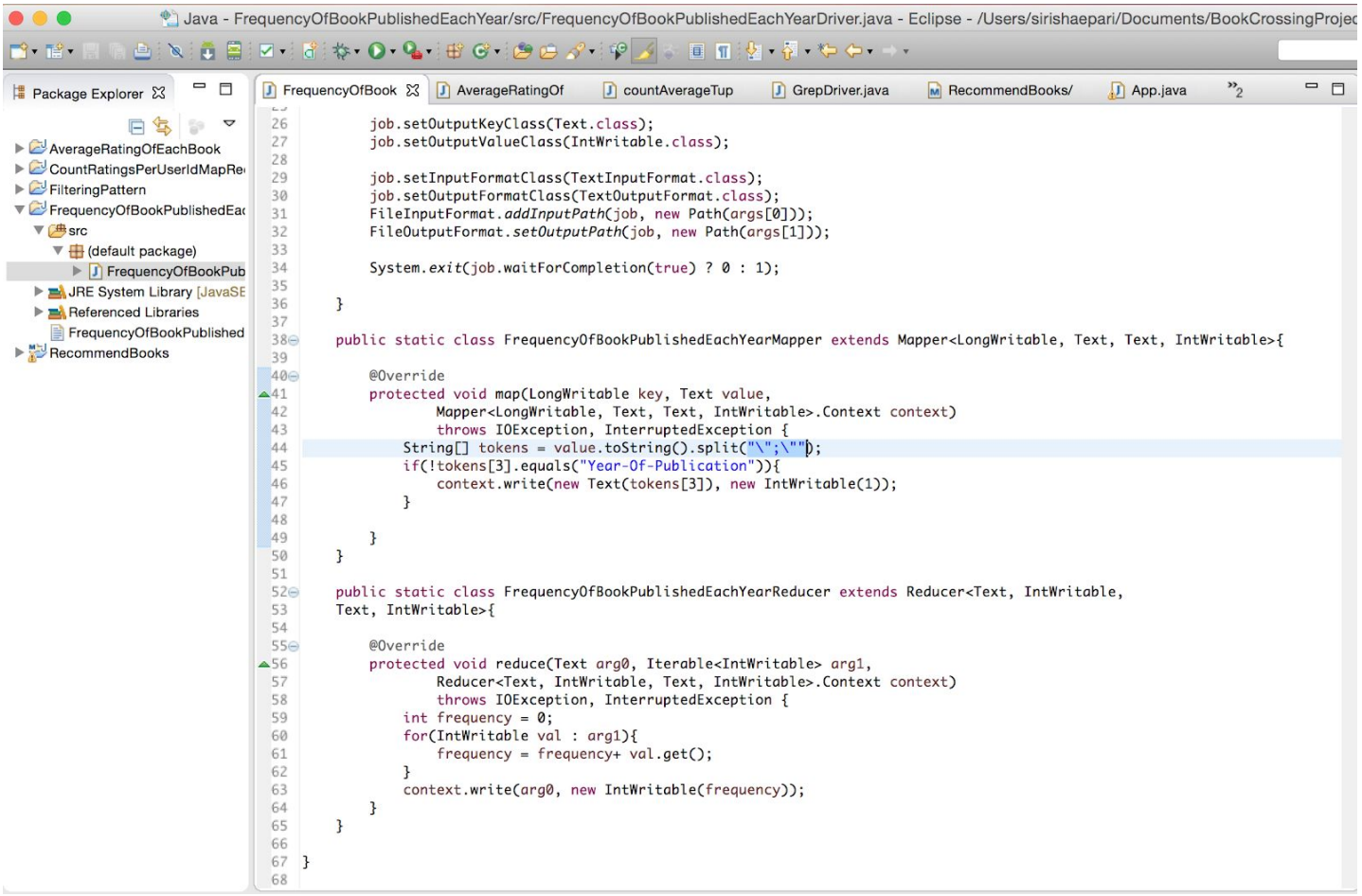
```
Sirishas-MacBook-Pro:bin sirishaepari$ hadoop fs -ls /BigDataProject/Dataset/BX-CSV-Dump
2018-04-16 21:18:33,871 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where a
pplicable
Found 3 items
-rw-r--r--   1 sirishaepari supergroup   30682276 2018-04-16 21:17 /BigDataProject/Dataset/BX-CSV-Dump/BX-Book-Ratings.csv
-rw-r--r--   1 sirishaepari supergroup   77787439 2018-04-16 21:17 /BigDataProject/Dataset/BX-CSV-Dump/BX-Books.csv
-rw-r--r--   1 sirishaepari supergroup   12284157 2018-04-16 21:17 /BigDataProject/Dataset/BX-CSV-Dump/BX-Users.csv
Sirishas-MacBook-Pro:bin sirishaepari$
```

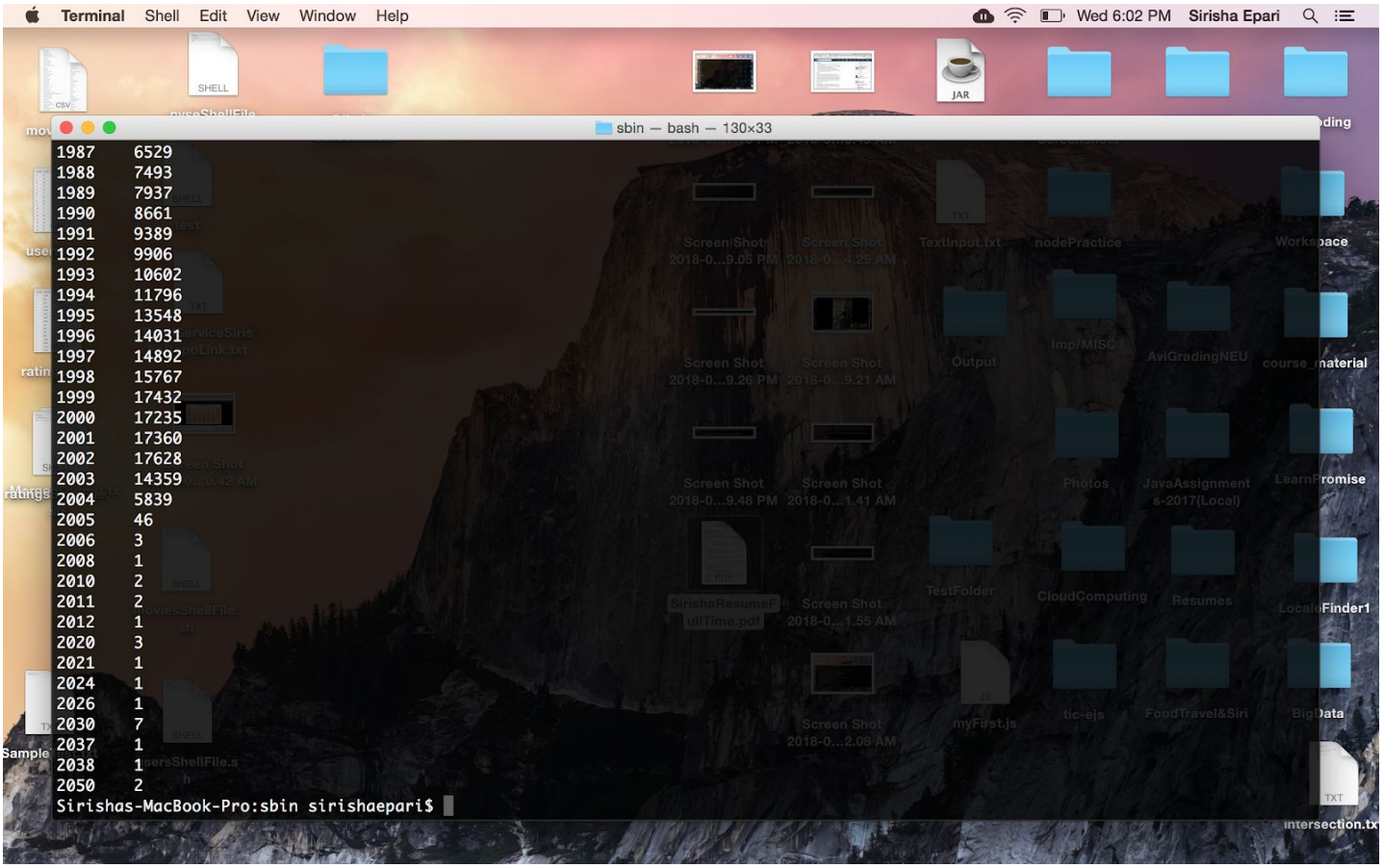
Analysis:

Frequency of book published each year

Sol: Use the ‘books’ file and set key as “Year of Published” and value as “1”.

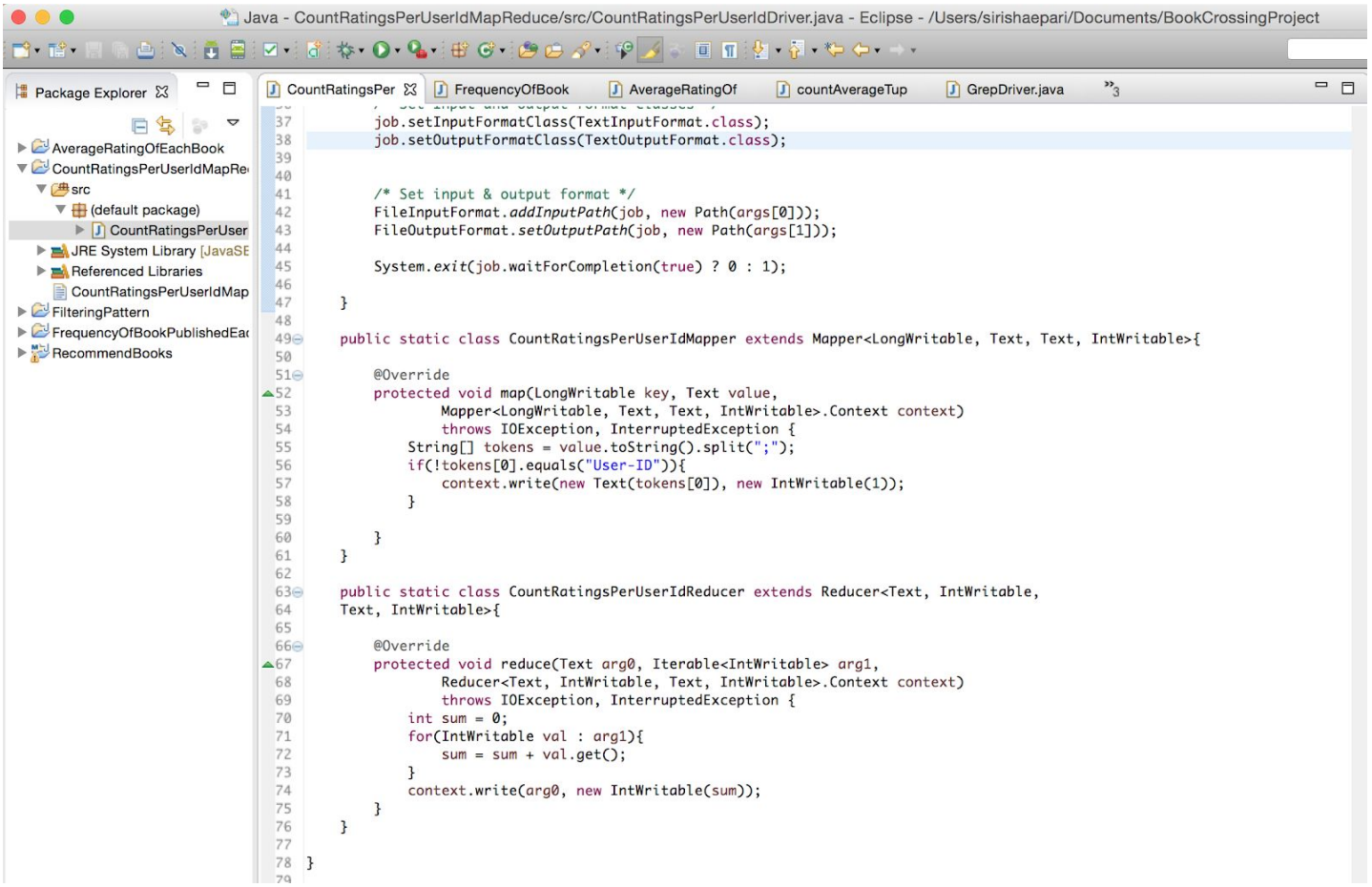
Challenge: Was to split data properly and retrieve correct fields. As the csv format was “col1”;”col2”;”col3”

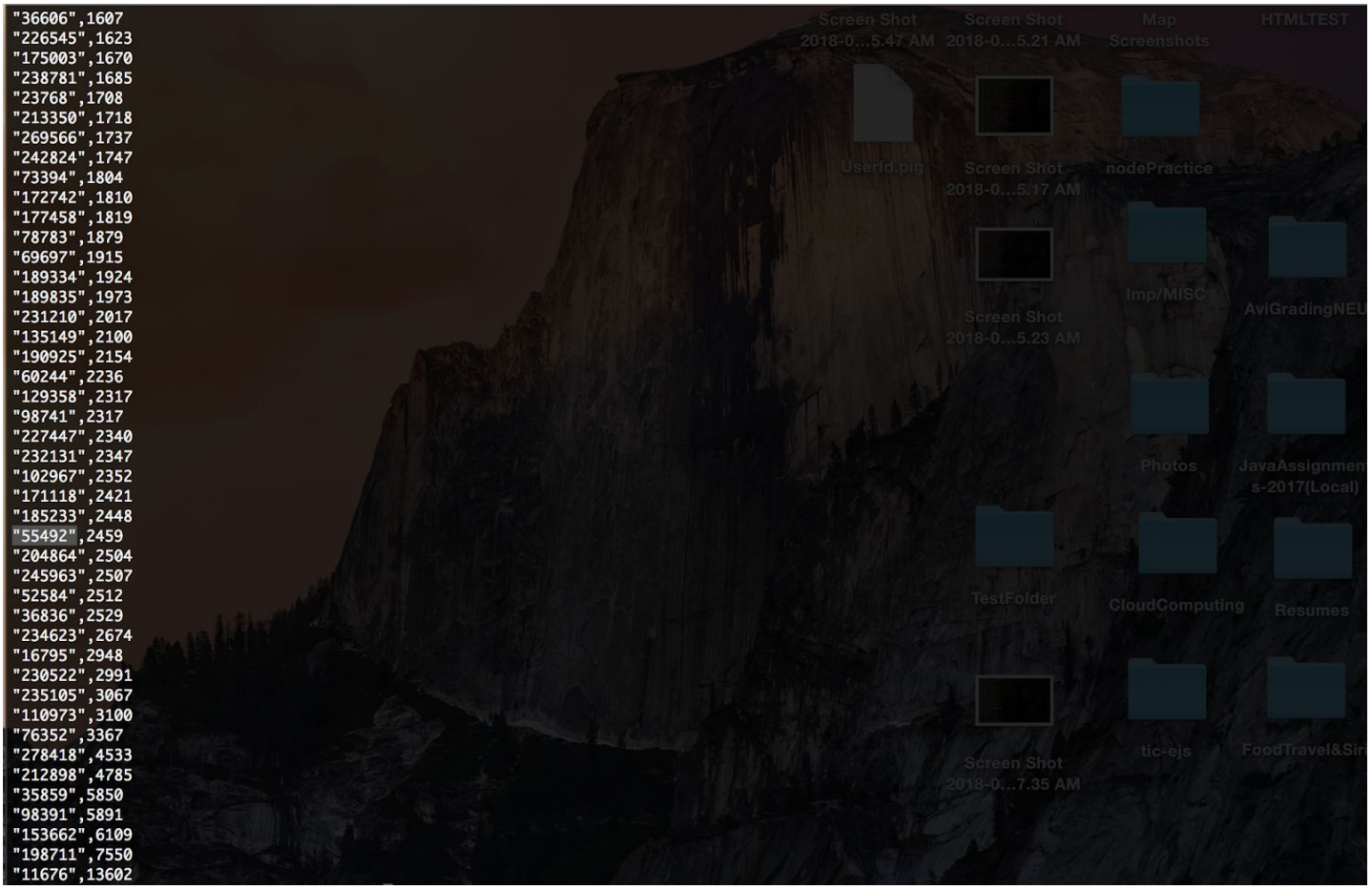




Find the list of users who are most active and sent maximum number of ratings.

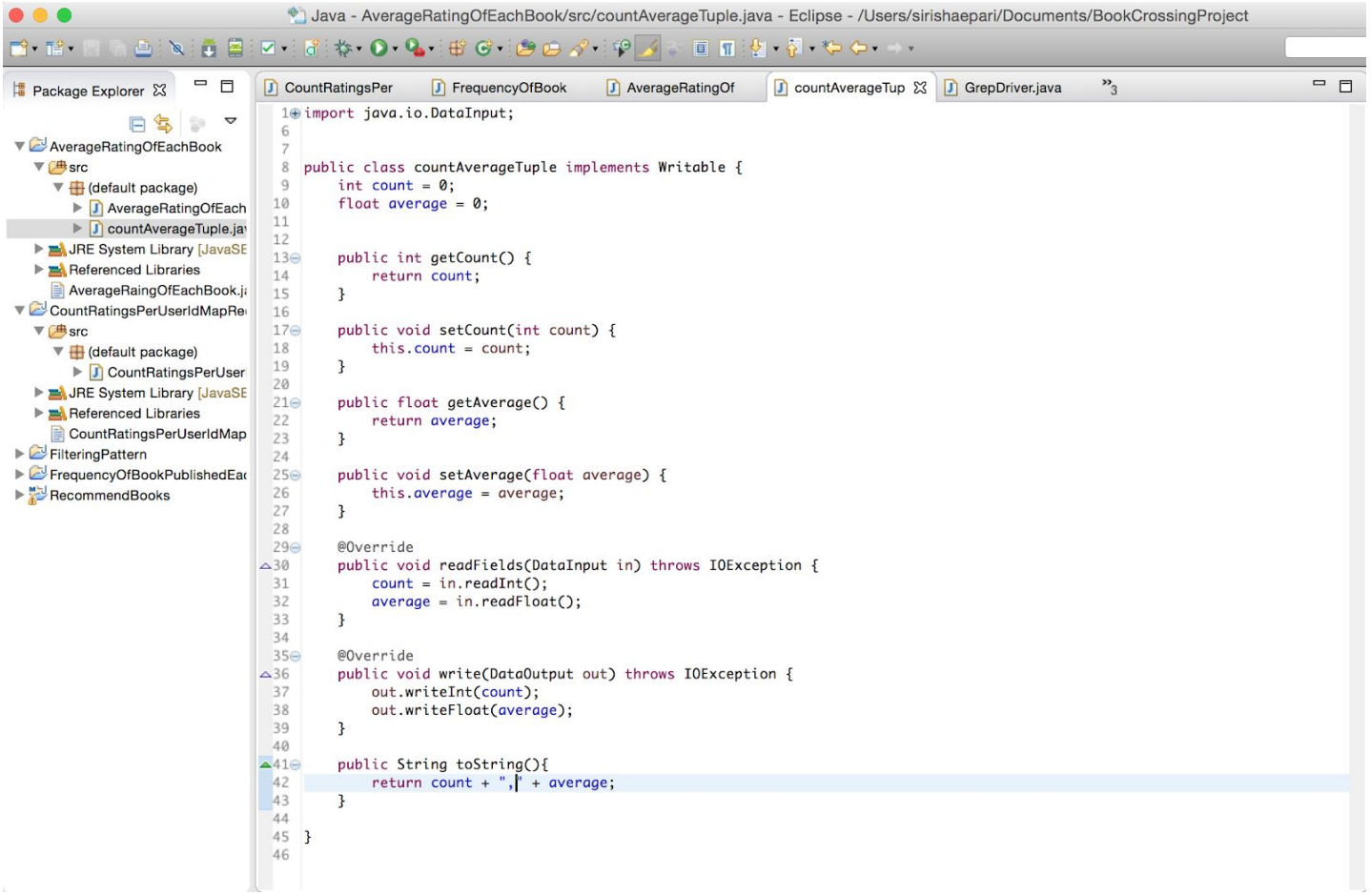
Sol: Count number of ratings given by each user and sort them by ratings.

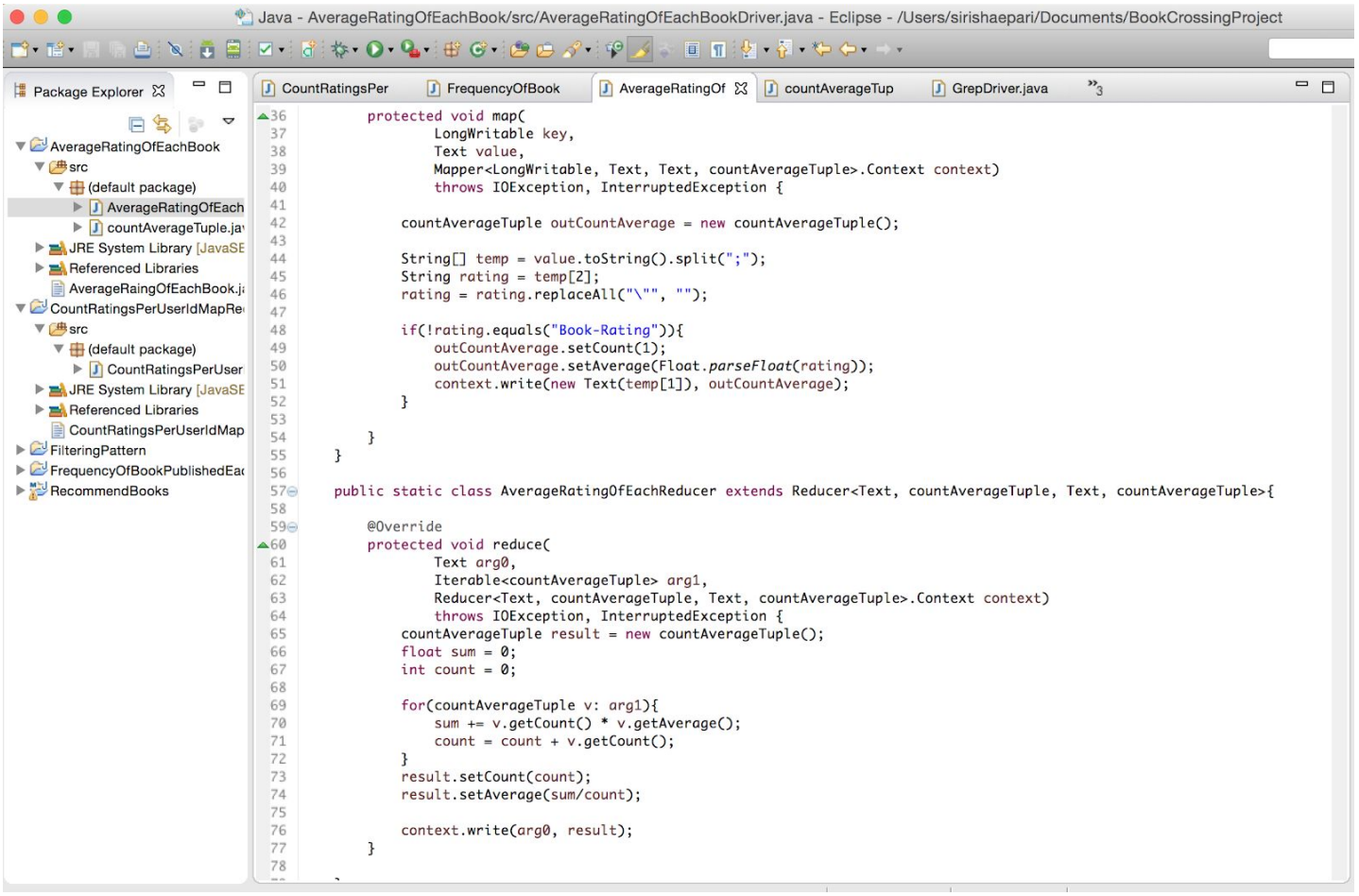


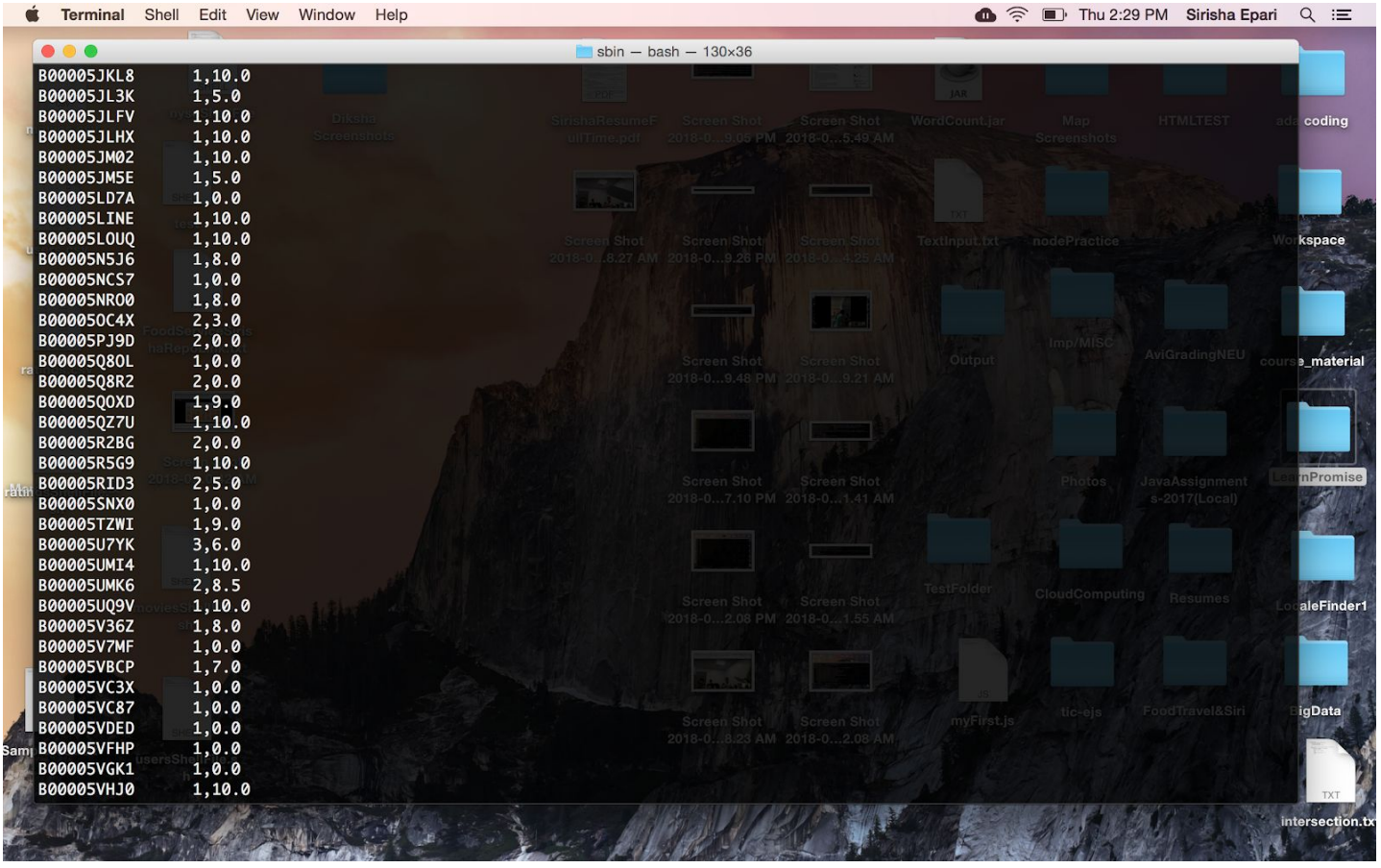


Average Rating of each book

Sol: Create a custom **countAverageTuple** writable class. I am implementing Writable class as it's passed as a value. Hence, i only need to override the read and write methods.
In the mapper class, send the key as book id and value as countAverageTuple.
In reducer, loop through the values and find average.







(hmm but what is first field? What does that mean?)

Find the book details with meaningful information like 'Book id, Book title and average rating'

Sol: Find the average rating of each book (above result) and join with the ‘Books’ file on book-id and determine the meaningful result. Limit the number of records to 10.

AverageRatingPerBook.pig

/usr/local/Cellar/hadoop/3.0.0/libexec/sbin/AverageRatingPerBook.pig

```
1  A = LOAD '/BigDataProject/Dataset/BX-CSV-Dump/BX-Books.csv' USING PigStorage(';') as (ISBN:chararray,BookTitle:chararray,BookAuthor:chararray,Yea
2  B = LOAD '/BigDataProject/OutputData/AverageRatingOfEachBook/part-r-00000' USING PigStorage(',') as (ISBN:chararray,count:int,average:float);
3  C = join A by $0, B by $0;
4  D = FOREACH C GENERATE $0,$1,$10;
5  E = limit D 10;
6  STORE E INTO '/BigDataProject/OutputData/AverageRatingPerBook';
7
```

```
("0000913154","The Way Things Work: An Illustrated Encyclopedia of Technology",8.0)
("0001010565","Mog's Christmas",0.0)
("0001046438","Liar",9.0)
("0001046713","Twopence to Cross the Mersey",0.0)
("000104687X","T.S. Eliot Reading \"The Wasteland\" and Other Poems",6.0)
("0001046934","The Prime of Miss Jean Brodie",0.0)
("0001047213","The Fighting Man",9.0)
("0001047647","First Among Equals",0.0)
("0001047663","Matter Of Honour",0.0)
("0001047868","Kidnapped",0.0)
```

Display top-ten list of books with highest ratings.

Sol: Sort the above result based on highest rating and display top ten records.

OrderByAverage.pig

/usr/local/Cellar/hadoop/3.0.0/libexec/sbin/OrderByAverage.pig

```
1  A = LOAD '/BigDataProject/Dataset/BX-CSV-Dump/BX-Books.csv' USING PigStorage(';') as (ISBN:chararray,BookTitle:chararray,BookAuthor
2  B = LOAD '/BigDataProject/OutputData/AverageRatingOfEachBook/part-r-00000' USING PigStorage(',') as (ISBN:chararray,count:int,avera
3  C = join A by $0, B by $0;
4  D = FOREACH C GENERATE $0,$1,$10;
5  E = ORDER D BY $2 DESC;
6  F = limit E 10;
7  STORE F INTO 'BigDataProject/OutputDat/sortedAverageRatingPerBook';
8
```

```
("1902418239","Avebury",10.0)
("1858683858","Ultimate Encyclopedia of Science Fiction the De",10.0)
("1858683688","The Natural History Museum Book of Dinosaurs",10.0)
("0759661804","Life Is an Adventure",10.0)
("0486424650","The Communist Manifesto and Other Revolutionary Writings: Marx, Marat, Paine, Mao Tse-Tung, Gandhi and Others",10.0)
("0619109688","Microsoft Office XP New Features Guide: Office 2000 To XP Changes",10.0)
("8525008044","Pau-Brasil (Obras completas de Oswald de Andrade)",10.0)
("968130165X","Como Hacerse Rico Sin Preocupacione",10.0)
("0307301990","Barbie &amp;amp,10.0)
("185863864X","George: a 20th Century Miniature",10.0)
```

Display list of all books published from year 2001 - 2008 and sort the books by year published.

Sol: Filter and then sort by books by year published.

rangeYearOfPublisher.pig

/usr/local/Cellar/hadoop/3.0.0/libexec/sbin/rangeYearOfPublisher.pig

```
1  A = LOAD '/BigDataProject/Dataset/BX-CSV-Dump/BX-Books.csv' USING PigStorage(';') as (ISBN:chararray,BookTitle:chararray,BookAuthor:c
2  B = FOREACH A GENERATE $0,$1,$2,(int)REPLACE($3,'\\"',''),$4 ;
3  C = FILTER B By $3 >= 2000 AND $3 <= 2008;
4  D = ORDER C BY $3;
5  DUMP D;
6
7
```

```
("1582343594","City of Masks : A Cree Black Novel","Daniel Hecht",2004,"Bloomsbury USA")
("0743257502","The Glory Cloak : A Novel of Louisa May Alcott and Clara Barton","Patricia O'Brien",2004,"Touchstone")
("1594140332","Designer Genes: Tales of the Biotech Revolution","Brian Stableford",2004,"Five Star (ME)")
("0765304775","Wind Spirit (Ella Clah)","Aimee Thurlo",2004,"Forge Books")
("0765310112","The Journal of Professor Abraham Van Helsing","Allen C. Kupfer",2004,"Forge Books")
("0849917905","Black","Ted Dekker",2004,"WestBow Press")
("0060570563","Babe in Toyland","Eugenie Seifer Olson",2004,"Avon Trade")
("082177557X","Come Summer","Sandra Steffen",2004,"Zebra Books")
("0345467159","The Alpine Pursuit (Daheim, Mary)","MARY DAHEIM",2004,"Ballantine Books")
("0802141358","The Mammoth Cheese","Sheri Holman",2004,"Grove Press")
("0312324766","The Men We Became : My Friendship with John F. Kennedy, Jr.","Robert T. Littell",2004,"St. Martin's Press")
("0758208022","The Royal Treatment","Maryjanice Davidson",2004,"BRAVA")
("0373691637","Legally Mine (Harlequin Temptation, No. 963)","Kate Hoffmann",2004,"Harlequin")
("1570722625","When the Last Magnolia Weeps","Mary Saums",2004,"Silver Dagger Mysteries")
("0451211707","Under a Lucky Star (Star Trilogy (Paperback))","Diane Farr",2004,"Signet Book")
("0736912932","A Quarter for a Kiss (Clark, Mindy Starns. Million Dollar Mysteries, 4.)","Mindy Starns Clark",2004,"Harvest House Publishers")
("0765301296","Nano","John Robert Marlow",2004,"Forge Books")
("0312865392","Inventing Memory","Anne Harris",2004,"Tor Books")
("0881501719","The Asey Mayo Trio: Three Cape Cod Mysteries","Phoebe Atwood Taylor",2005,"Countryman Press")
("0743492072","Crowner'S Quest : A Crowner John Mystery (Crowner John Mystery)","Bernard Knight",2005,"Simon &amp;amp)
("0743484894","The Winter's Tale (Folger Shakespeare Library)","William Shakespeare",2005,"Washington Square Press")
("0060541490","Sexy","Joyce Carol Oates",2005,"HarperTempest")
("0140144463","The Cornish Trilogy","Robertson Davies",2005,"Penguin Books Ltd")
("0140124373","Red Dwarf","Grant Naylor",2005,"Penguin Books Ltd")
("0141002999","Thunderball (James Bond 007)","Ian Fleming",2005,"Penguin Books Ltd")
("0060567341","Chinese Cinderella and the Secret Dragon Society","Adeline Yen Mah",2005,"HarperCollins")
("1552782662","The Shanghai Murders","David Rotenberg",2005,"McArthur &amp;amp)
("0060738189","Serpico","Peter Maas",2005,"Perennial")
("0743492056","The Sanctuary Seeker : A Crowner John Mystery","Bernard Knight",2005,"Simon &amp;amp)
("0140860363","On the Road","Jack Kerouac",2005,"Penguin Highbridge")
("0671878654","An Oblique Approach","Eric Flint",2005,"Baen")
("1552782603","The Lake Ching Murders","David Rotenberg",2005,"McArthur &amp;amp)
("0743492064","The Poisoned Chalice : A Crowner John Mystery (Crowner John Mystery)","Bernard Knight",2005,"Simon &amp;amp)
("0374103747","In My Brother's Shadow","Uwe Timm",2005,"Farrar, Straus and Giroux")
("0752848291","A Village Dilemma","Rebecca Shaw",2005,"Orion mass market paperback")
("0743490258","Best of Friends","Cathy Kelly",2005,"Downtown Press")
("0141002980","Moonraker (James Bond 007)","Ian Fleming",2005,"Penguin Books Ltd")
("0375829849","The Golden Goose","Dick King-Smith",2005,"Knopf Books for Young Readers")
("0385338082","Can You Keep a Secret?","SOPHIE KINSELLA",2005,"Delta")
("088150078X","The Annulet of Gilt: An Asey Mayo Cape Cod Mystery","Phoebe Atwood Taylor",2005,"Countryman Press")
("0140622454","A Woman of No Importance (Penguin Popular Classics)","Oscar Wilde",2005,"Penguin Books Ltd")
("1593100175","Beauty Is Soul Deep","Michelle Lee",2005,"Barbour Publishing")
```

Find all unique list of books with book id and book name.

Sol: Use DISTINCT to find list of unique list of books from ‘Books.csv’

UniqueBookList.pig

/usr/local/Cellar/hadoop/3.0.0/libexec/sbin/UniqueBookList.pig

```
1  /* Find unique list of books with thier ISBN number and Book name */
2  A = LOAD '/BigDataProject/Dataset/BX-CSV-Dump/BX-Books.csv' USING PigStorage(';') as (ISBN:chararray,BookTitle:chararray,BookAu
3  B = DISTINCT A;
4  C = FOREACH B GENERATE $0,$1;
5  DUMP C;
```

```
("9999669972","Being Bullied (Lets Talk About Series)")
("9999980538","Murphy's Law")
("9999984584","Anointed")
("9999991556","Impact of New Product Introductions on the Market Value of Firms (Research Program, Working Paper/Report No 89-105)")
("9999999999","World Directory of Awards and Prizes")
("B00001IVC7","Desperation")
("B00001U0CP","Unnatural Exposure")
("B000023VWQ","House Without A Key")
("B000023VWR","The Postman Always Rings Twice")
("B000023VWW","I, The Jury")
("B000023VWY","The Dragon Murder Case")
("B00004THMF","My Gal Sunday : Henry and Sunday Stories")
("B000051WXP","Emma")
("B000051X7T","The Oval Portrait")
("B000051XGM","Off on a Comet")
("B000051ZUO","Triangle")
("B000051ZUT","The Trellisane Confrontation")
("B0000523SS","The Covenant of the Crown")
("B0000523SU","Ghost Ship")
("B0000523SW","The Children of Hamlin (Star Trek: The Next Generation, Book 3) [DOWNLOAD: MICROSOFT READER]")
("B0000523SY","Strike Zone (Star Trek: The Next Generation, Book 5) [DOWNLOAD: MICROSOFT READER]")
("B000055Z5S","The Second Coming of Lucas Brokaw")
("B00005B4LM","Downward to the Earth")
("B00005BBW4","Sense and Sensibility")
("B00005BJZ7","Fierce Eden")
("B00005LD7A","Believing It All: What My Children Taught Me about Trout Fishing, Jelly Toast and Life")
("B00005N5J6","Dinosaur Summer")
("B00005NCS7","Moonlight Becomes You")
("B00005Q8OL","My Gal Sunday")
("B00005Q8R2","Hell's Kitchen")
("B00005TZWI","Sandkings")
("B00005U7YK","The New Gurus -- From Sun-Tzu And Jesus To Machiavelli And Winnie The Pooh")
("B00005UMI4","Desecration:  Left Behind #9")
("B00005UMK6","Best a Man Can Get")
("B00005V7MF","The Company: The Story of a Murderer")
("B0000633PU","The Story of Aladdin and the Wonderful Lamp")
("B00006391H","Southern Nights (Florida)")
("B0000639MD","Dying Inside")
("B0000640CT","For Love of Mother Not")
("B000068RVK","The Victoria's Secret Catalog Never Stops Coming: And Other Lessons I Learned From Breast Cancer")
("B000069F44","The Lady in the Lake")
("B00006CRTE","Devil Knows You're Dead, The: A Matthew Scudder Crime Novel")
```

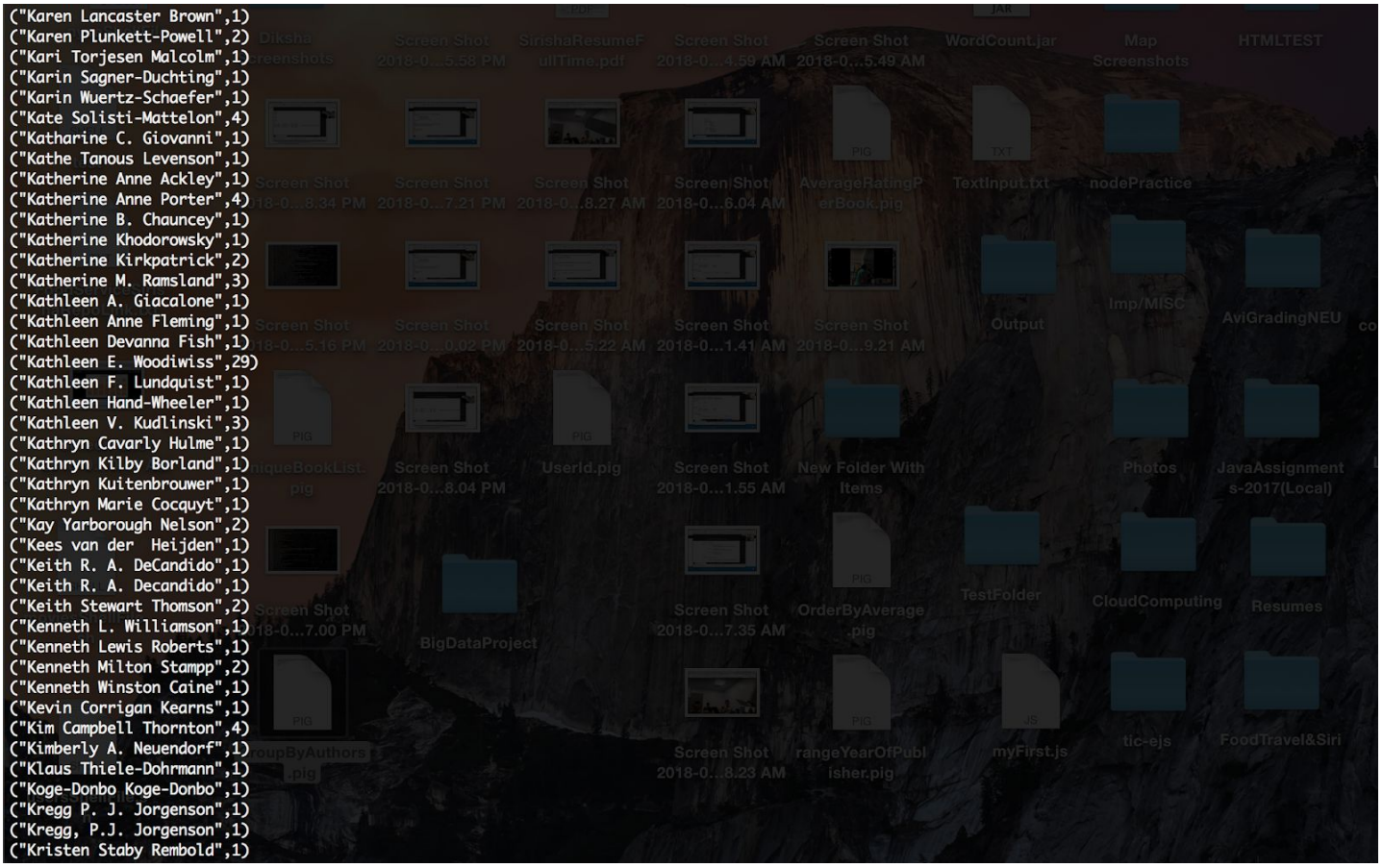
Display number of books each author published.

GroupByAuthors.pig

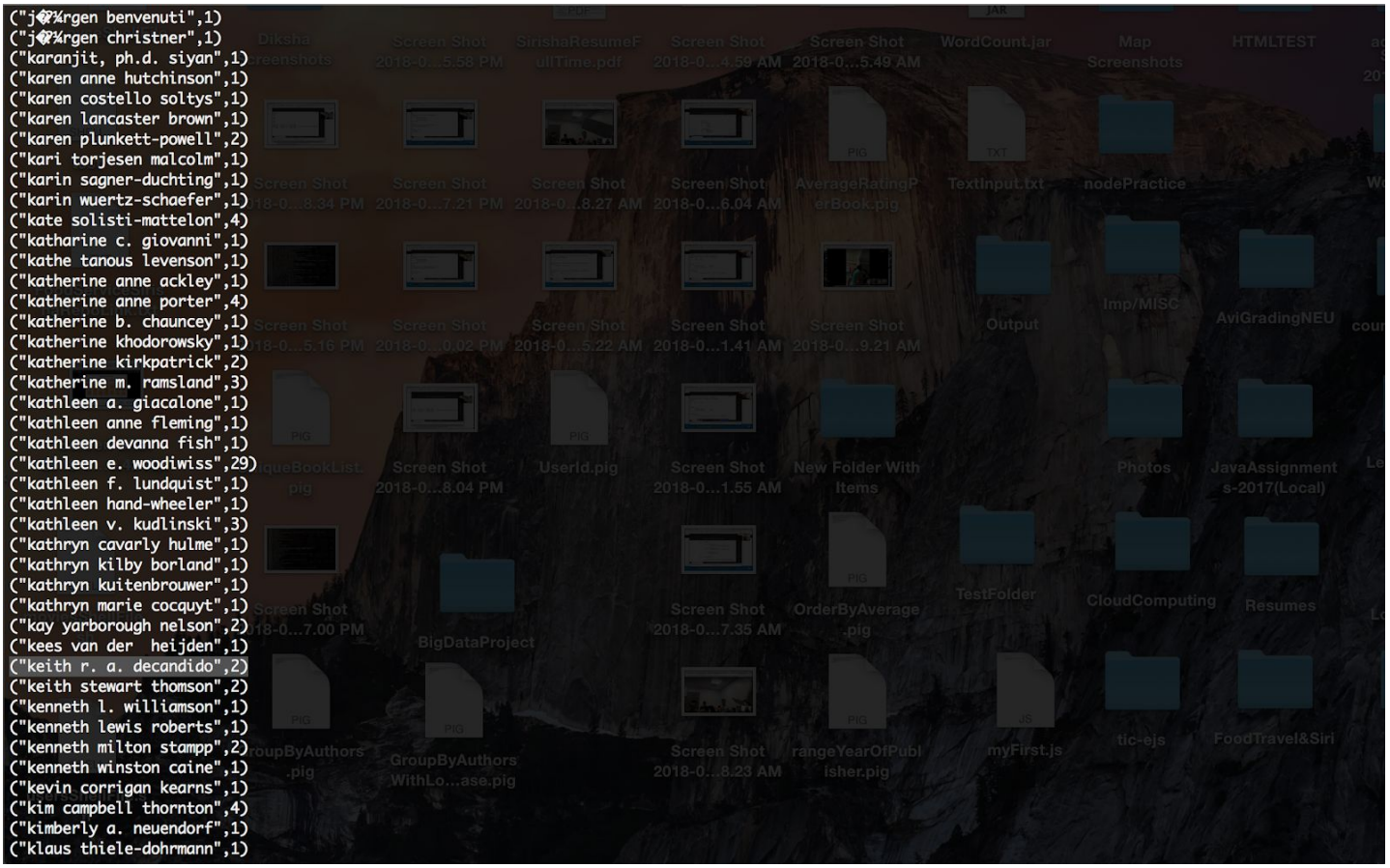
/usr/local/Cellar/hadoop/3.0.0/libexec/sbin/GroupByAuthors.pig

```
1
2  A = LOAD '/BigDataProject/Dataset/BX-CSV-Dump/BX-Books.csv' USING PigStorage(';') as (ISBN:chararray,BookTitle:chararray,BookAuthor:c
3  B = GROUP A BY $2;
4  C = FOREACH B GENERATE
5  group as author,
6  COUNT(A);
7  DUMP C;
8
```

/* When it group by it is case sensitive
Hence you can see few same author names but grouped differently
Ex: Keith R. A. Decandido */

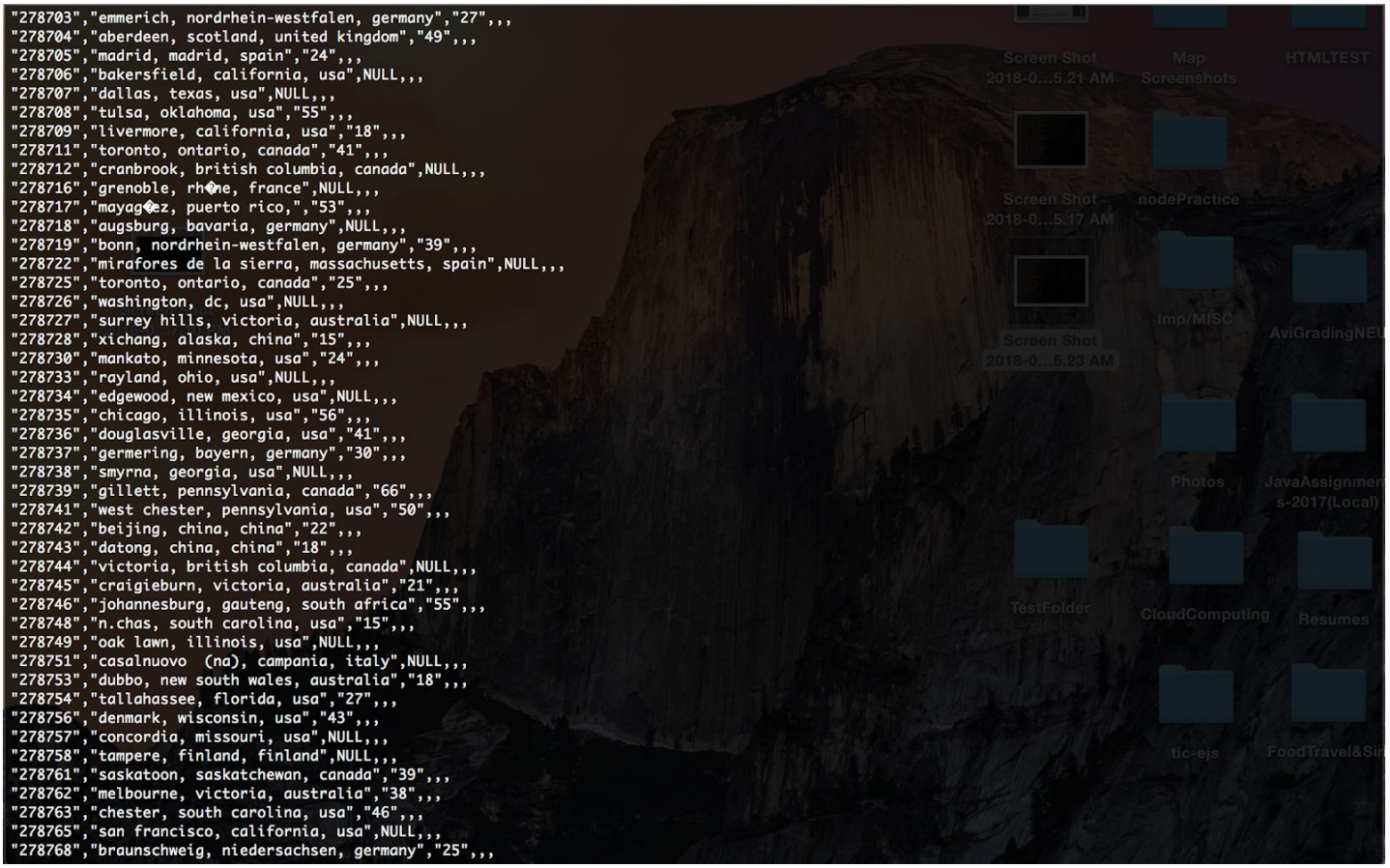


Used LOWER function to first convert and then group by name to get proper results.



Find the users who bought the books but not yet rated any book. So that we can send notifications to them to ask to rate.

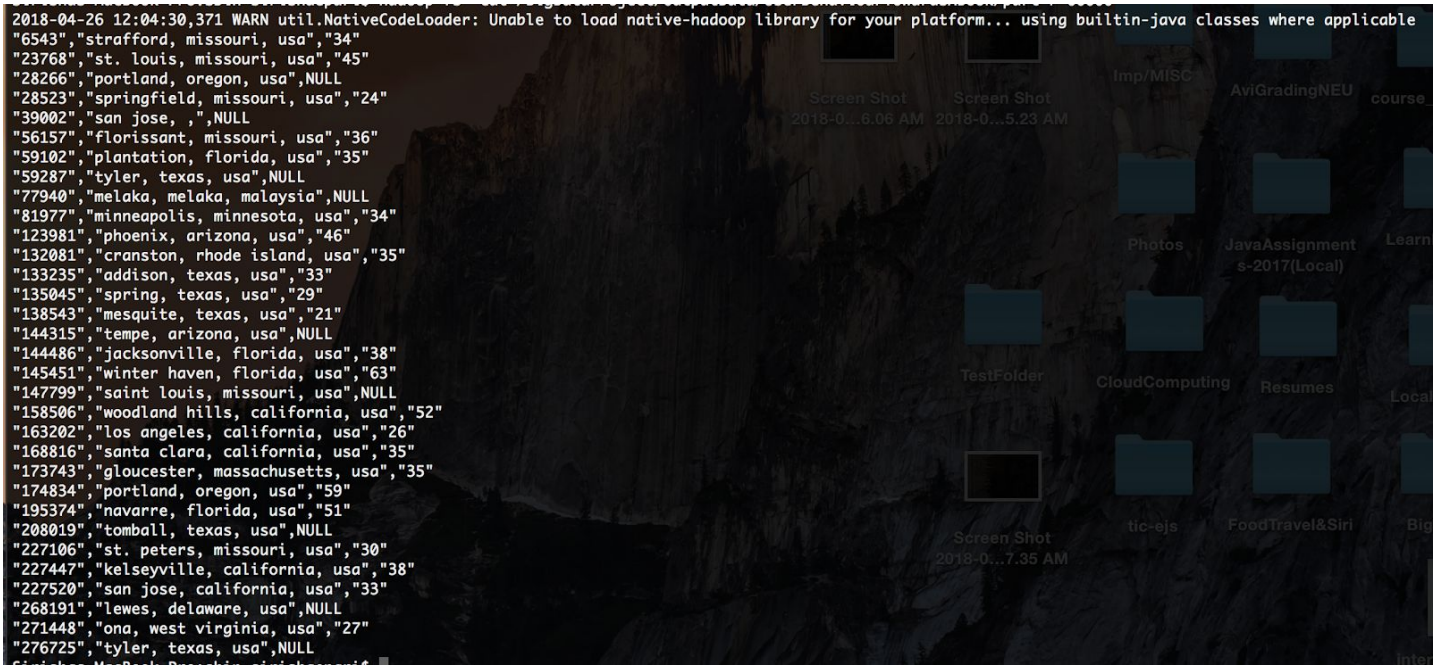
Sol: Find the distinct users in “Book-ratings” and distinct users in “users” file and do a outer left join to get the list of the users who have not yet given ratings.



Find the behaviour of users who have given ratings.

Sol: Join the ‘ratings file’ with ‘user file’ via user id and find the information.

This helps to find user behaviour like location, age etc



Hive :

Data cleaning: I have removed the first line of the csv file. So, that it will not be processed in hive while executing queries.

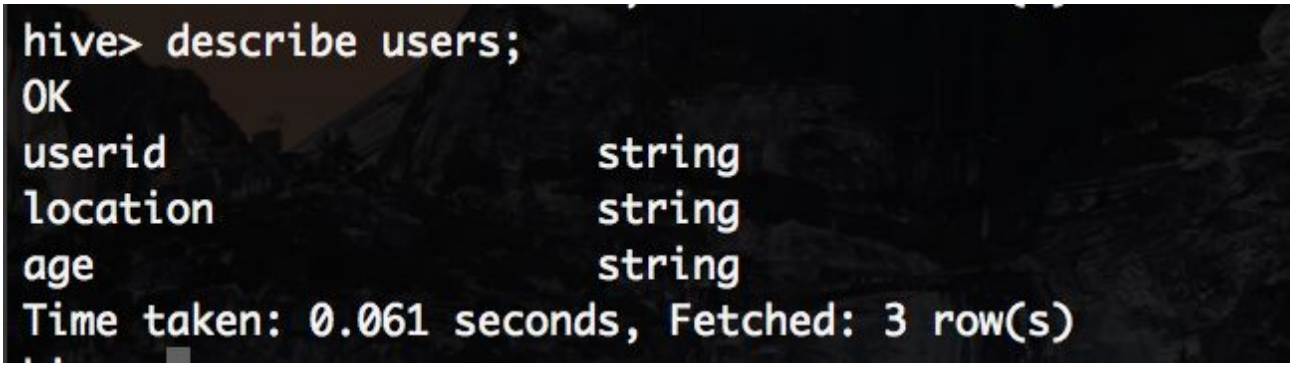
Created a directory named /BookCrossingDataset/Users in HDFS.

Used the hdfs put command to copy the file from local file system to hdfs.

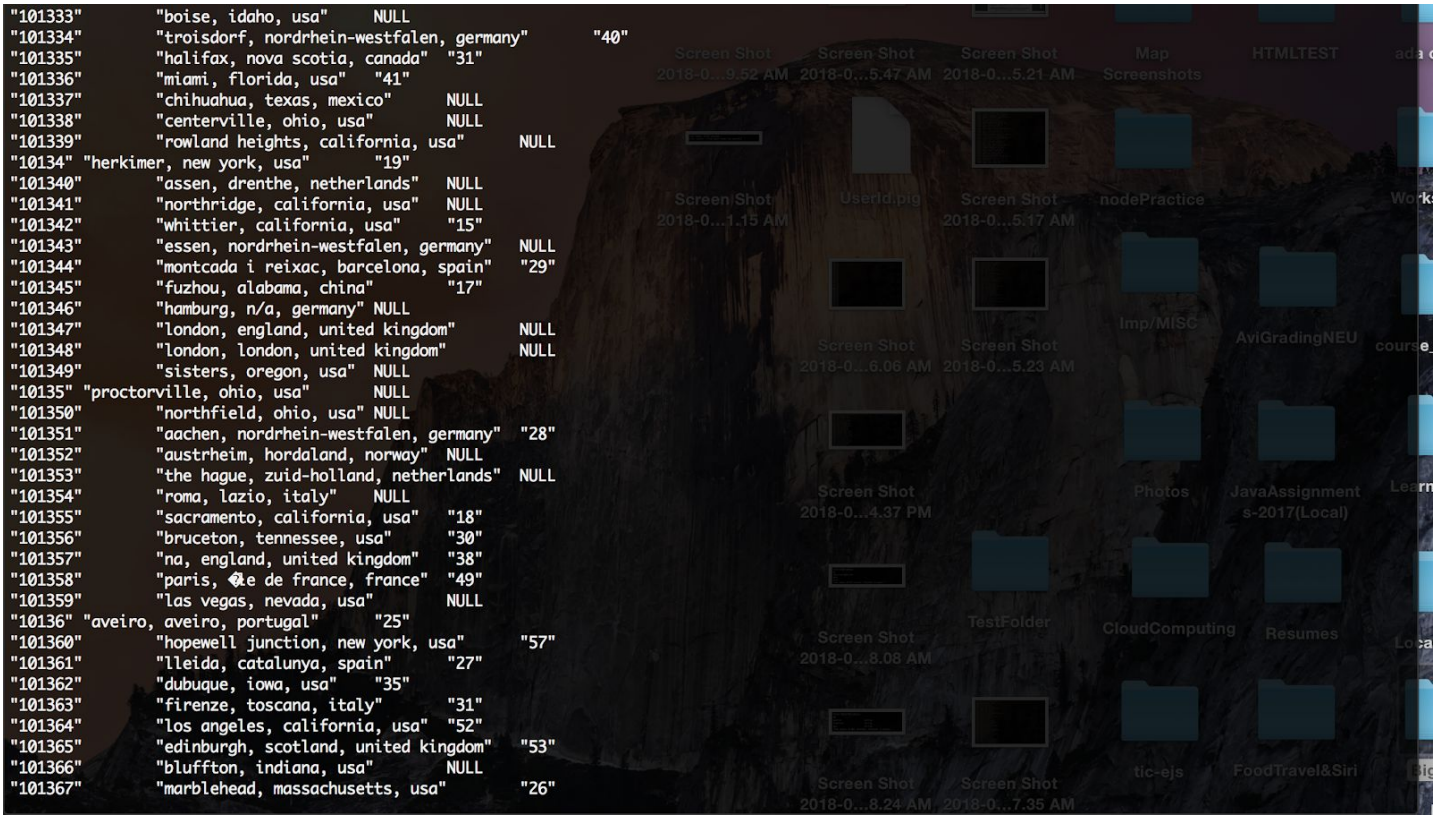
In hive created a table named **users**

```
CREATE TABLE users ( userid STRING, location STRING, age STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ‘;’
STORED AS TEXTFILE;

LOAD DATA INPATH ‘/../.../BX-Users1.csv’ OVERWRITE INTO TABLE users;
```



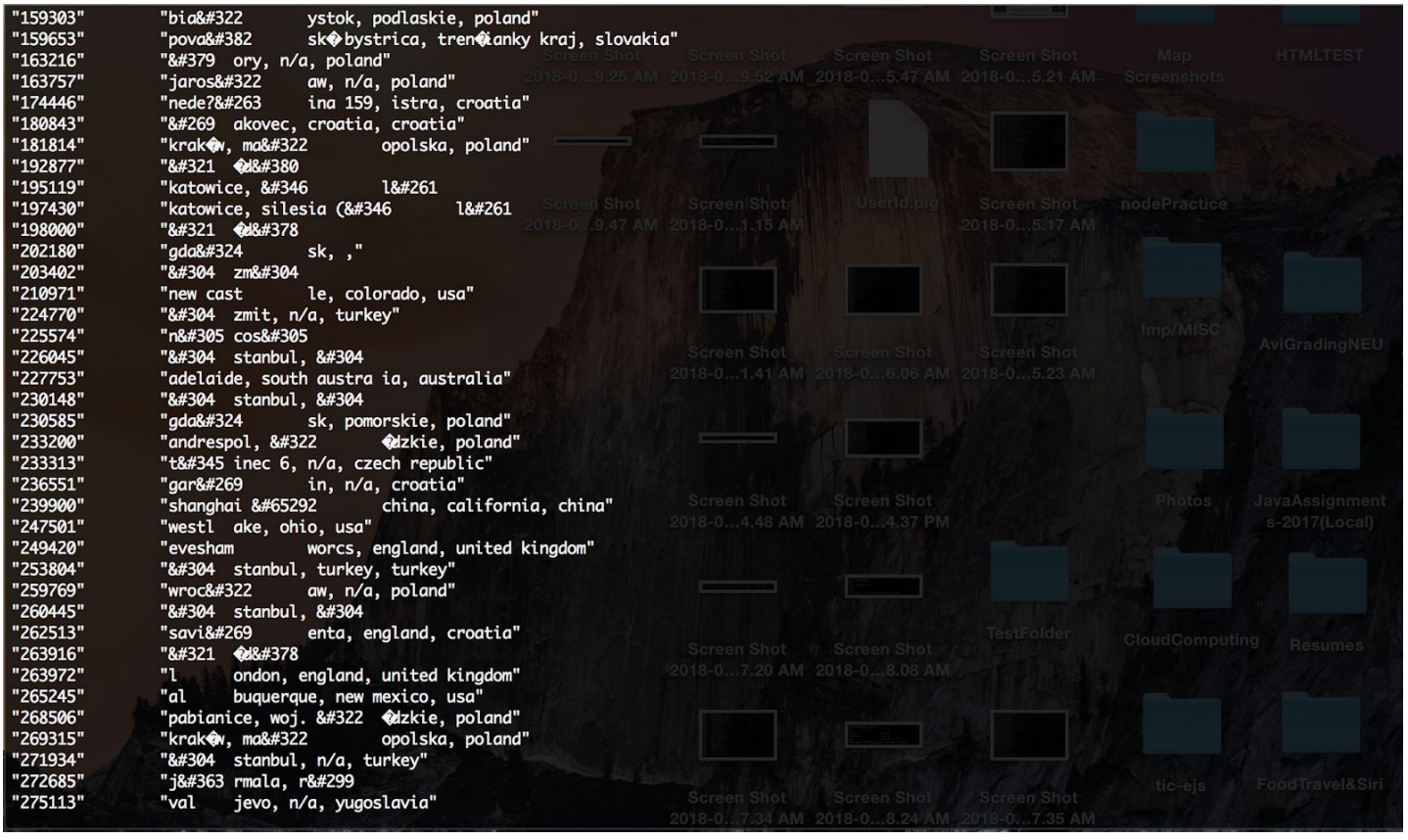
Give the list of user in sorted order by userid.



```
"101333"        "boise, idaho, usa"     NULL
"101334"        "troisdorf, nordrhein-westfalen, germany"       "40"
"101335"        "halifax, nova scotia, canada"  "31"
"101336"        "miami, florida, usa"   "41"
"101337"        "chihuahua, texas, mexico"      NULL
"101338"        "centerville, ohio, usa"        NULL
"101339"        "rowland heights, california, usa"      NULL
"10134" "herkimer, new york, usa"       "19"
"101340"        "assen, drenthe, netherlands"   NULL
"101341"        "northridge, california, usa"   NULL
"101342"        "whittier, california, usa"     "15"
"101343"        "essen, nordrhein-westfalen, germany"   NULL
"101344"        "montcada i reixac, barcelona, spain"   "29"
"101345"        "fuzhou, alabama, china"        "17"
"101346"        "hamburg, n/a, germany" NULL
"101347"        "london, england, united kingdom"       NULL
"101348"        "london, london, united kingdom"        NULL
"101349"        "sisters, oregon, usa"  NULL
"10135" "proctorville, ohio, usa"       NULL
"101350"        "northfield, ohio, usa" NULL
"101351"        "aachen, nordrhein-westfalen, germany"  "28"
"101352"        "austrheim, hordaland, norway"  NULL
"101353"        "the hague, zuid-holland, netherlands"  NULL
"101354"        "roma, lazio, italy"    NULL
"101355"        "sacramento, california, usa"   "18"
"101356"        "bruceton, tennessee, usa"      "30"
"101357"        "na, england, united kingdom"   "38"
"101358"        "paris, ?e de france, france"   "49"
"101359"        "las vegas, nevada, usa"        NULL
"10136" "aveiro, aveiro, portugal"      "25"
"101360"        "hopewell junction, new york, usa"      "57"
"101361"        "lleida, catalunya, spain"      "27"
"101362"        "dubuque, iowa, usa"    "35"
"101363"        "firenze, toscana, italy"       "31"
"101364"        "los angeles, california, usa"  "52"
"101365"        "edinburgh, scotland, united kingdom"   "53"
"101366"        "bluffton, indiana, usa"        NULL
"101367"        "marblehead, massachusetts, usa"        "26"
```

Find the user with age above 20

Sol: Since there are null values in the list. First filter the data where users age is not null and then filter by where age is greater than 20.

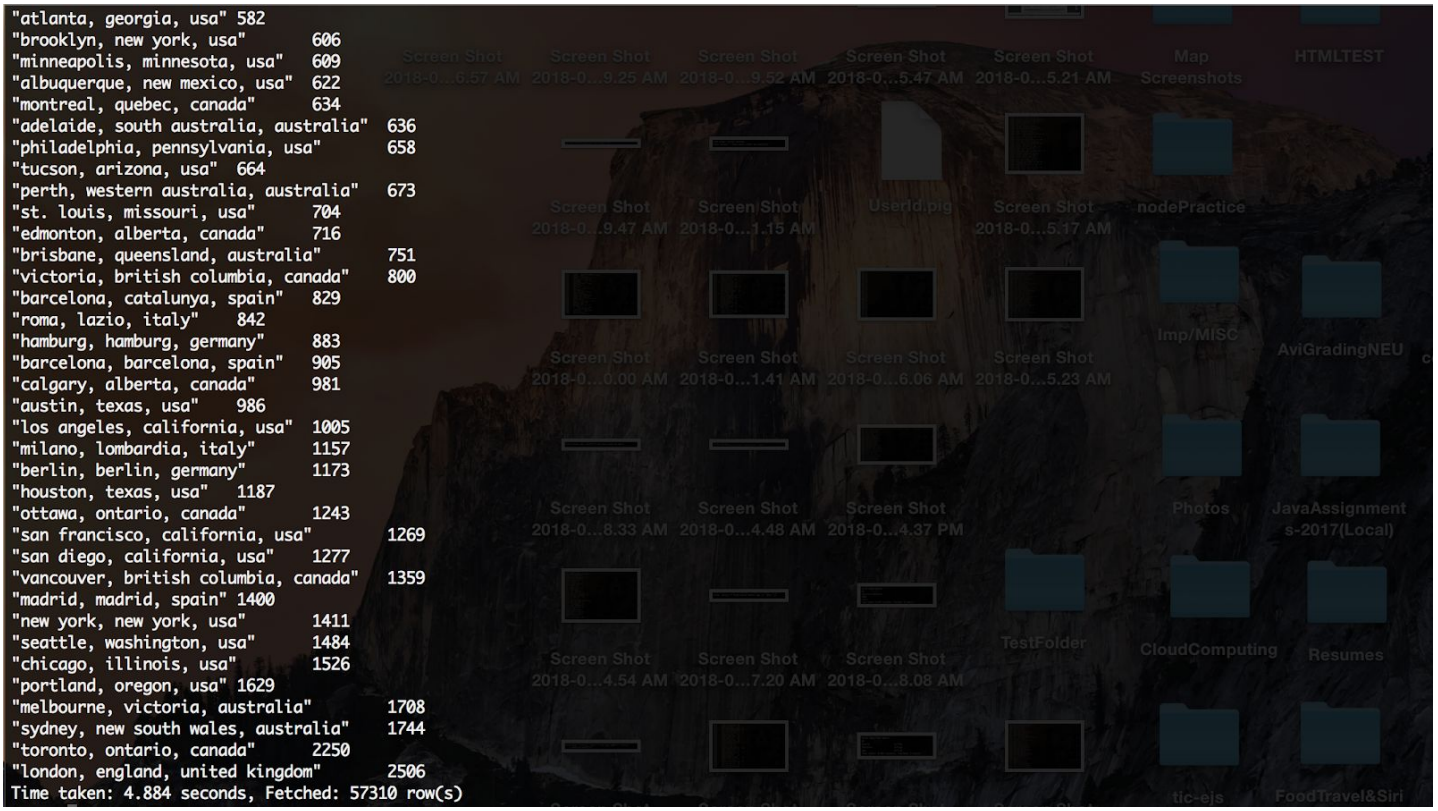


```
"159303"        "bia&#322       ystok, podlaskie, poland"
"159653"        "pova&#382      sk?bystrica, tren?anky kraj, slovakia"
"163216"        "&#379  ory, n/a, poland"
"163757"        "jaros&#322     aw, n/a, poland"
"174446"        "nede?&#263     ina 159, istra, croatia"
"180843"        "&#269  akovec, croatia, croatia"
"181814"        "krak?w, ma&#322        opolska, poland"
"192877"        "&#321  ?&#380
"195119"        "katowice, &#346        l&#261
"197430"        "katowice, silesia (&#346       l&#261
"198000"        "&#321  ?&#378
"202180"        "gda&#324       sk, ,"
"203402"        "&#304  zm&#304
"210971"        "new cast       le, colorado, usa"
"224770"        "&#304  zmit, n/a, turkey"
"225574"        "n&#305 cos&#305
"226045"        "&#304  stanbul, &#304
"227753"        "adelaide, south austra ia, australia"
"230148"        "&#304  stanbul, &#304
"230585"        "gda&#324       sk, pomorskie, poland"
"233200"        "andrespol, &#322       ?dzkie, poland"
"233313"        "t&#345 inec 6, n/a, czech republic"
"236551"        "gar&#269       in, n/a, croatia"
"239900"        "shanghai &#65292       china, california, china"
"247501"        "westl  ake, ohio, usa"
"249420"        "evesham        worcs, england, united kingdom"
"253804"        "&#304  stanbul, turkey, turkey"
"259769"        "wroc&#322      aw, n/a, poland"
"260445"        "&#304  stanbul, &#304
"262513"        "savi&#269      enta, england, croatia"
"263916"        "&#321  ?&#378
"263972"        "l      ondon, england, united kingdom"
"265245"        "al     buquerque, new mexico, usa"
"268506"        "pabianice, woj. &#322  ?dzkie, poland"
"269315"        "krak?w, ma&#322        opolska, poland"
"271934"        "&#304  stanbul, n/a, turkey"
"272685"        "j&#363 rmala, r&#299
"275113"        "val    jevo, n/a, yugoslavia"
```

Find users who purchase books are mostly from which location?

Sol: group by location and count, and then sort the results.

From this data set we understand most of the users are from **"london, england, united kingdom" 2506**



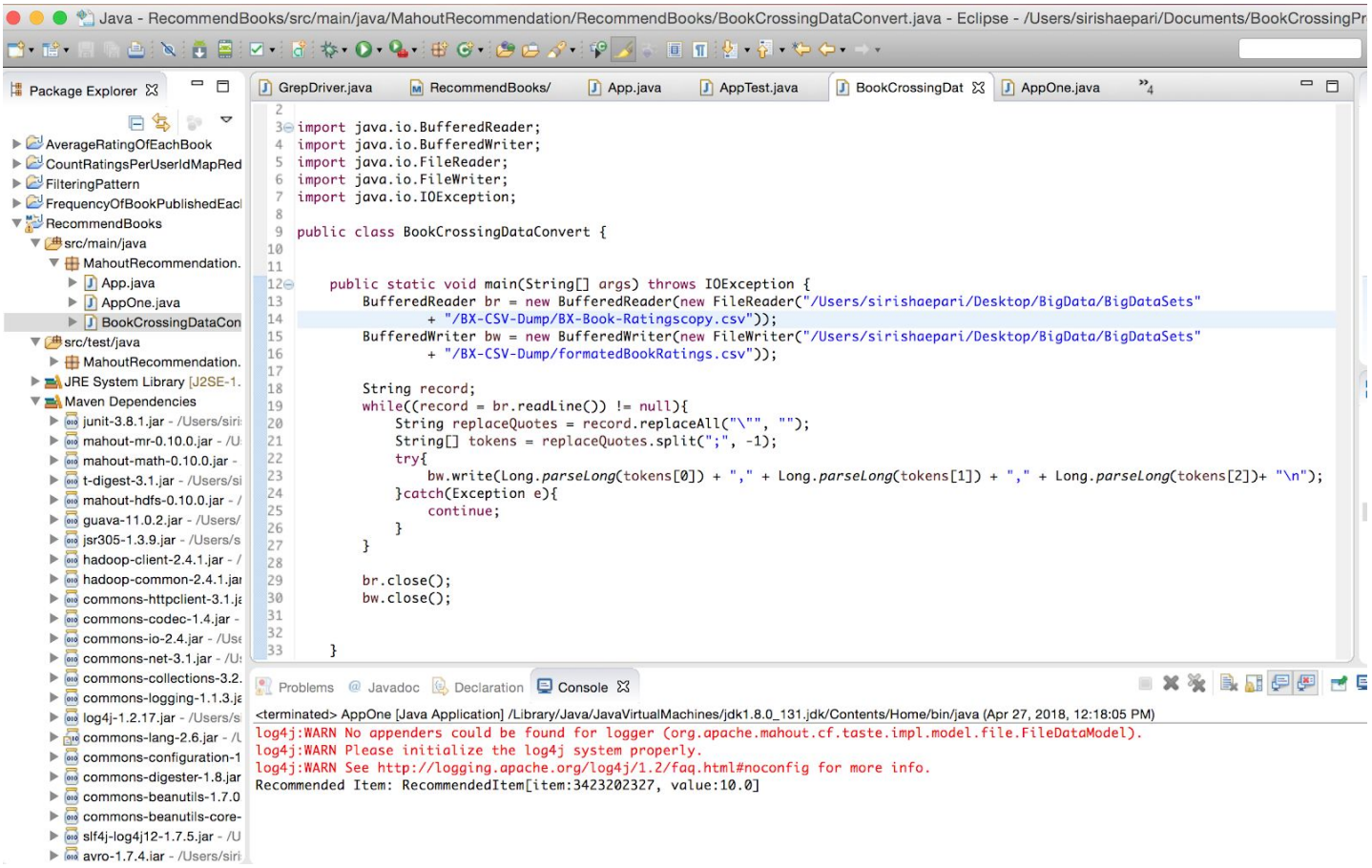
Recommend user a book based on his past purchase history.

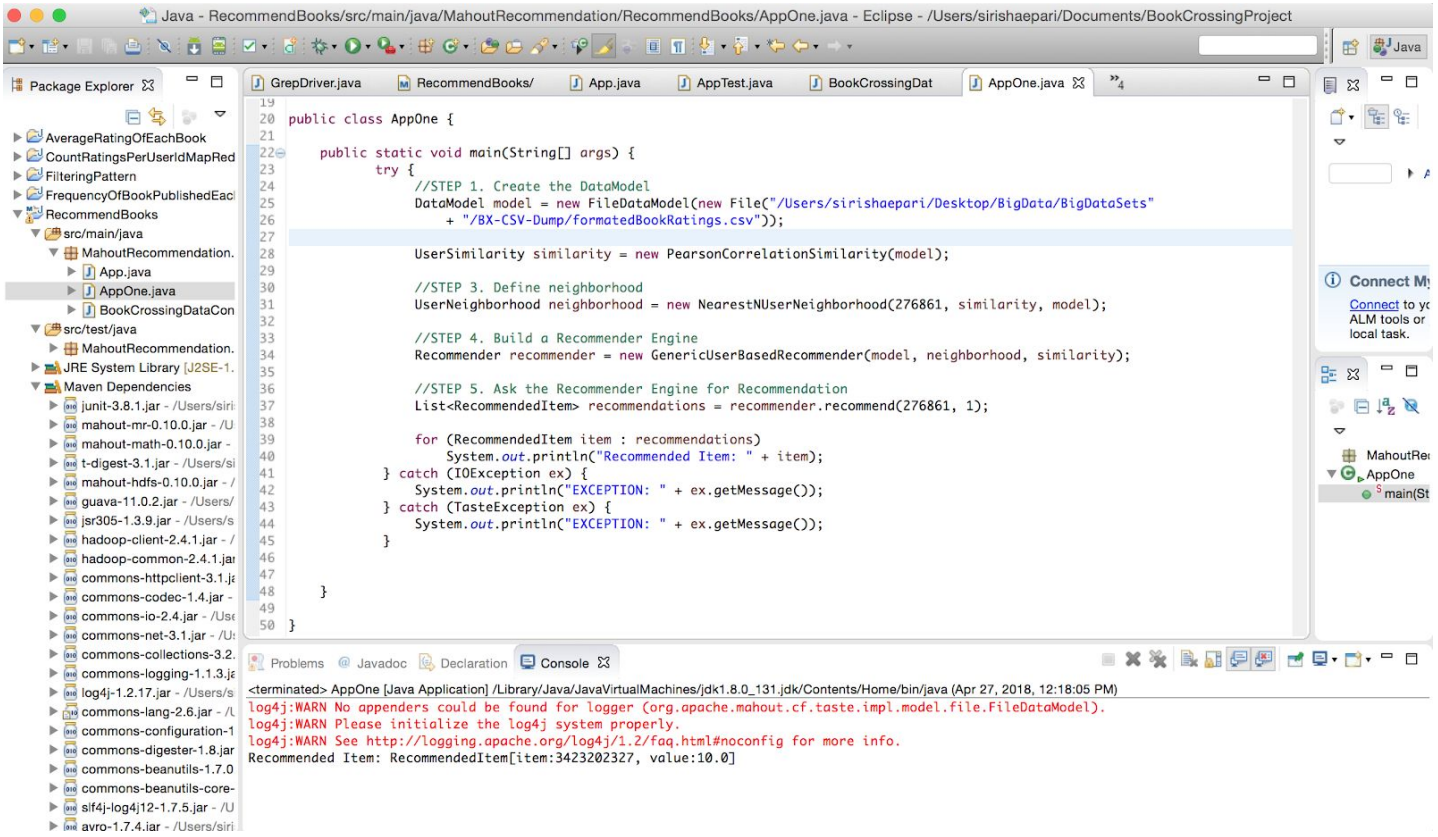
Sol: Firstly, I have to format the data.

- Remove all the quotes from the file
- Since, the book id had mix of integer & character, I converted that into Long.
- Stored in a csv file named **formatedBookRating.csv**
- In another class, created the data model, user similarity, user Neighborhood, recommender.
- It suggests user a book based on previous interests.
- Here is choose pearsonCorrelationSimilarity

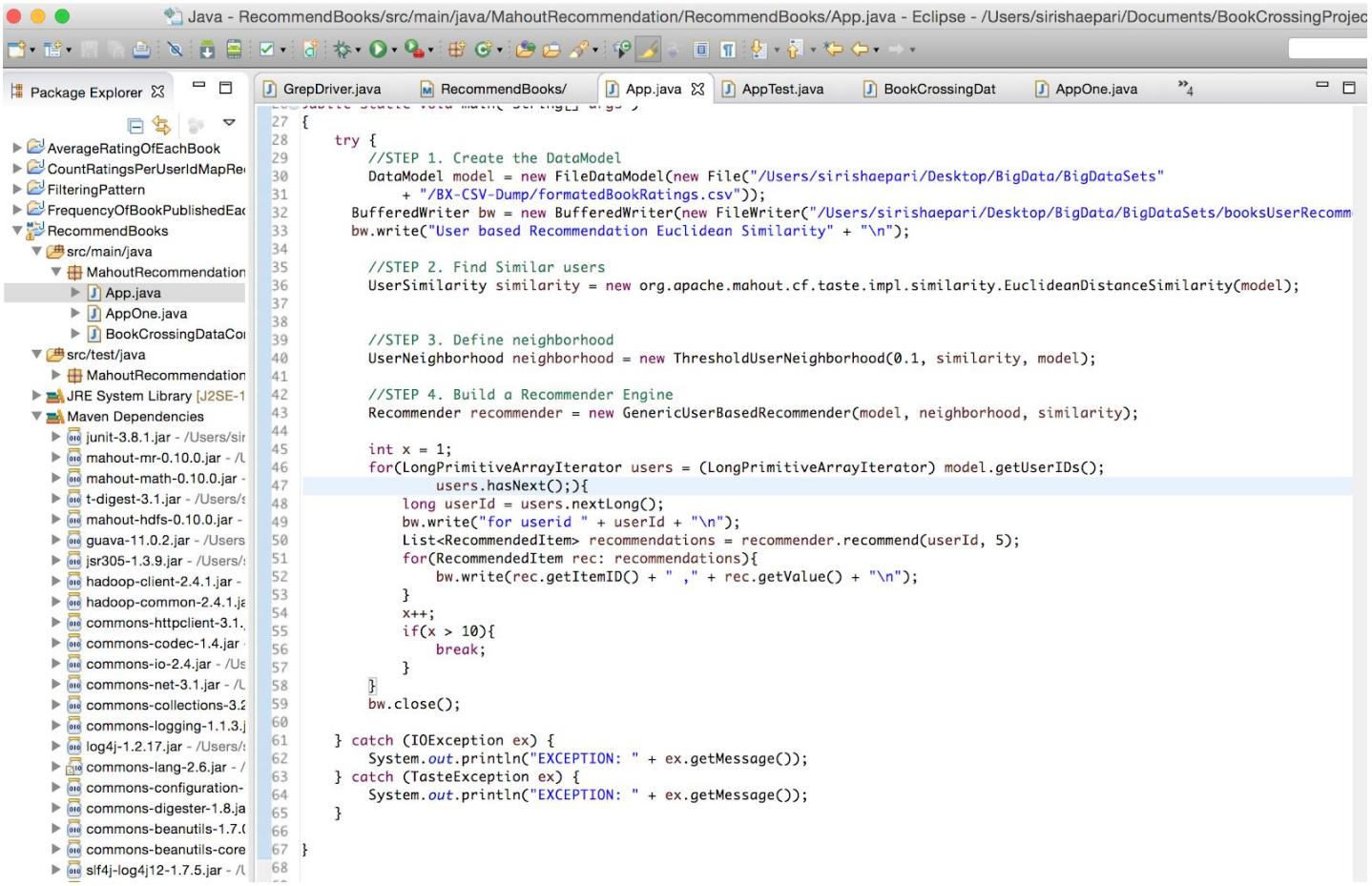
The format data looks like below:

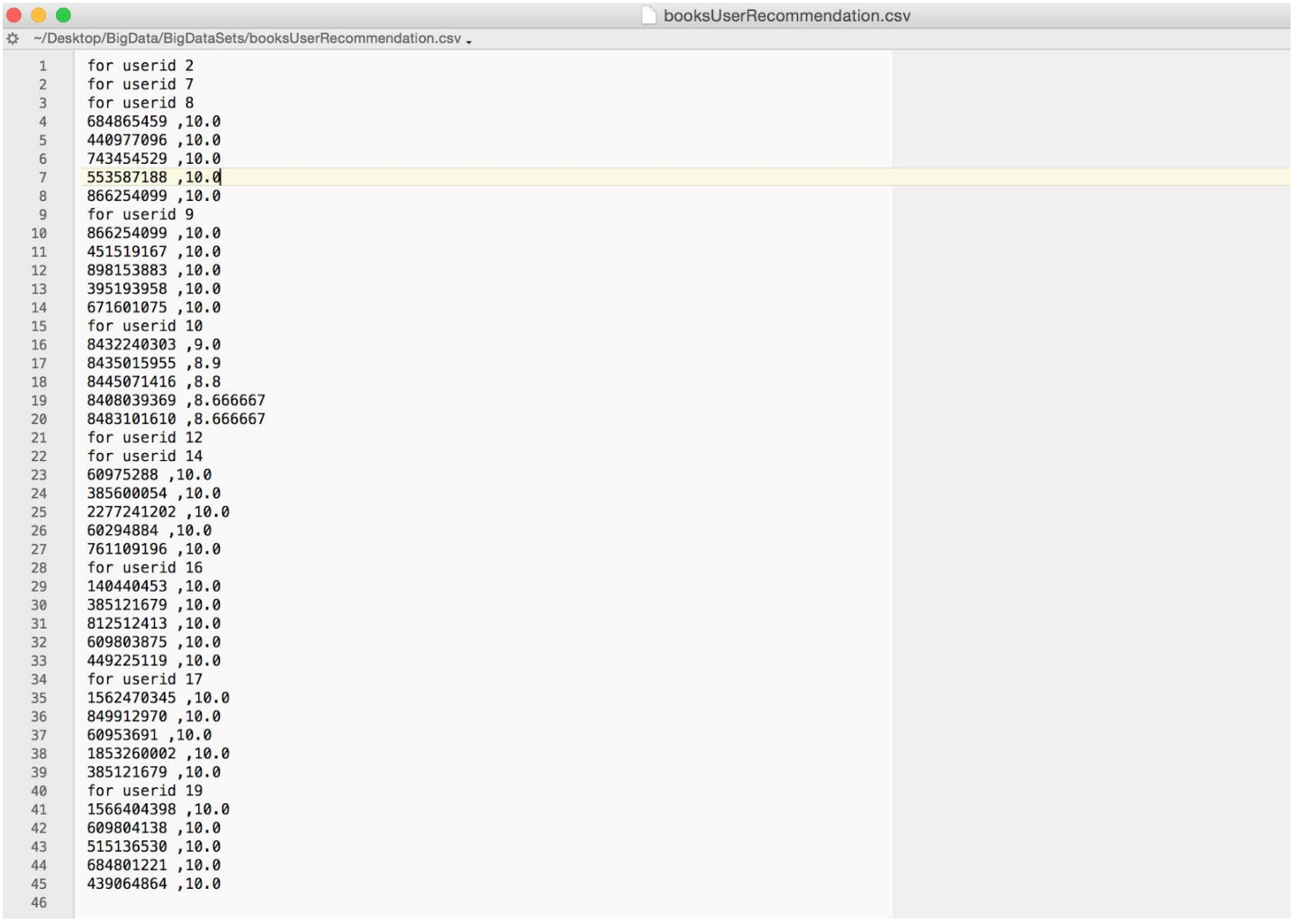
```
276726,155061224,5
276727,446520802,0
276729,521795028,6
276733,2080674722,0
276736,3257224281,8
276737,600570967,6
276745,342310538,10
276746,425115801,0
276746,449006522,0
276746,553561618,0
276746,786013990,0
276746,786014512,0
276747,60517794,9
276747,451192001,0
276747,609801279,0
276747,671537458,9
276747,679776818,8
276747,943066433,7
276747,1570231028,0
276747,1885408226,7
276748,747558167,6
276748,3442437407,0
276751,3596218098,8
276754,684867621,8
276755,451166892,5
276760,8440682697,10
276762,380000059,0
276762,380711524,5
276762,451167317,0
276762,451454952,0
276762,843920262,0
276762,3404122879,0
276762,3404182928,0
276762,3404611306,0
276762,342662429,0
276762,3426690179,0
276762,3442424216,0
276762,3442425573,0
276762,3453092007,8
276762,3453157745,0
276762,3453176944,0
276762,3453185137,0
276762,3453185323,0
276762,3453213025,3
276762,3453877241,0
276762,3492226604,0
276762,3517017442,0
```





User Recommendation using Euclidean Similarity: // Took help from internet


Java - RecommendBooks/src/main/java/MahoutRecommendation/RecommendBooks/App.java - Eclipse - /Users/sirishaepari/Documents/BookCrossingProjec
Package Explorer
GrepDriver.java
RecommendBooks/
App.java
AppTest.java
BookCrossingDat
AppOne.java
AverageRatingOfEachBook
CountRatingsPerUserIdMapRe
FilteringPattern
FrequencyOfBookPublishedEac
RecommendBooks
src/main/java
MahoutRecommendation
App.java
AppOne.java
BookCrossingDataCo
src/test/java
MahoutRecommendation
JRE System Library [J2SE-1
Maven Dependencies
junit-3.8.1.jar - /Users/sir
mahout-mr-0.10.0.jar - /U
mahout-math-0.10.0.jar -
t-digest-3.1.jar - /Users/s
mahout-hdfs-0.10.0.jar -
guava-11.0.2.jar - /Users
jsr305-1.3.9.jar - /Users/s
hadoop-client-2.4.1.jar -
hadoop-common-2.4.1.ja
commons-httpclient-3.1.
commons-codec-1.4.jar
commons-io-2.4.jar - /Us
commons-net-3.1.jar - /U
commons-collections-3.2
commons-logging-1.1.3.j
log4j-1.2.17.jar - /Users/s
commons-lang-2.6.jar - /
commons-configuration-
commons-digester-1.8.ja
commons-beanutils-1.7.(
commons-beanutils-core
slf4j-log4j12-1.7.5.jar - /U
{
try {
//STEP 1. Create the DataModel
DataModel model = new FileDataModel(new File("/Users/sirishaepari/Desktop/BigData/BigDataSets"
+ "/BX-CSV-Dump/formatedBookRatings.csv"));
BufferedWriter bw = new BufferedWriter(new FileWriter("/Users/sirishaepari/Desktop/BigData/BigDataSets/booksUserRecomm
bw.write("User based Recommendation Euclidean Similarity" + "\n");
//STEP 2. Find Similar users
UserSimilarity similarity = new org.apache.mahout.cf.taste.impl.similarity.EuclideanDistanceSimilarity(model);
//STEP 3. Define neighborhood
UserNeighborhood neighborhood = new ThresholdUserNeighborhood(0.1, similarity, model);
//STEP 4. Build a Recommender Engine
Recommender recommender = new GenericUserBasedRecommender(model, neighborhood, similarity);
int x = 1;
for(LongPrimitiveArrayIterator users = (LongPrimitiveArrayIterator) model.getUserIDs();
users.hasNext();){
long userId = users.nextLong();
bw.write("for userid " + userId + "\n");
List<RecommendedItem> recommendations = recommender.recommend(userId, 5);
for(RecommendedItem rec: recommendations){
bw.write(rec.getItemID() + " ," + rec.getValue() + "\n");
}
x++;
if(x > 10){
break;
}
}
bw.close();
} catch (IOException ex) {
System.out.println("EXCEPTION: " + ex.getMessage());
} catch (TasteException ex) {
System.out.println("EXCEPTION: " + ex.getMessage());
}
}


booksUserRecommendation.csv
~/Desktop/BigData/BigDataSets/booksUserRecommendation.csv
for userid 2
for userid 7
for userid 8
684865459 ,10.0
440977096 ,10.0
743454529 ,10.0
553587188 ,10.0
866254099 ,10.0
for userid 9
866254099 ,10.0
451519167 ,10.0
898153883 ,10.0
395193958 ,10.0
671601075 ,10.0
for userid 10
8432240303 ,9.0
8435015955 ,8.9
8445071416 ,8.8
8408039369 ,8.666667
8483101610 ,8.666667
for userid 12
for userid 14
60975288 ,10.0
385600054 ,10.0
2277241202 ,10.0
60294884 ,10.0
761109196 ,10.0
for userid 16
140440453 ,10.0
385121679 ,10.0
812512413 ,10.0
609803875 ,10.0
449225119 ,10.0
for userid 17
1562470345 ,10.0
849912970 ,10.0
60953691 ,10.0
1853260002 ,10.0
385121679 ,10.0
for userid 19
1566404398 ,10.0
609804138 ,10.0
515136530 ,10.0
684801221 ,10.0
439064864 ,10.0

Things I learnt from this project.

- How to transform the data in cleaner and good format to run analysis.
- How important data cleansing is.
- Look for Case sensitive, null, empty values, symbols like &amp etc are may give somewhat wrong analysis.
- How fast and simple pig (data flow language)
- Once you have data in HIVE, you can use any visualization tools like Tableau, Heu, Banana etc

https://wiki.apache.org/hadoop/MountableHDFS
https://github.com/d3/d3/wiki/Gallery