Case Study Description

Please download the movie lens data (about 250 MB) from this link. This dataset describes a 5-star rating and free-text tagging activity from www.movielens.org, a movie recommendation service. It contains 26024289 ratings and 753170 tag applications across 45843 movies. These data were created by 270896 users between January 09, 1995 and August 04, 2017. This dataset was generated on August 04, 2017.

You will find the following files in this dataset :-
1. genome-scores.csv
2. genome-tags.csv
3. links.csv
4. movies.csv
5. ratings.csv
6. readme.csv
7. tags.csv

Please load the following two files from the above data set into an appropriate location on hdfs :-

1. movies.csv
2. ratings.csv

Let us now implement the following use cases on the above data set :-

1. As a first excercise, please report the number of HDFS blocks created by both the file (movies.csv and ratings.csv) on HDFS.
2. Join the two tables (Hint :- Use Reduce Side Join) and find out the following :-
   - What are the movie titles that the user has rated ?
   - How many times a movie has been rated by the user ?
3. In question 2 above, what is the average rating given for a movie ?

SOLUTION#1

To see the number of blocks we can use Hadoop fsck command. We can also use web based Hadoop utility at localhost:50070. Here we use the fsck command

```
[acadgild.mmisra ~]$ hadoop fs -ls /files
18/07/20 11:23:02 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Found 5 items
drwxr-xr-x   - acadgild supergroup          0 2018-07-19 16:38 /files/assignment_1
drwxr-xr-x   - acadgild supergroup          0 2018-07-19 16:41 /files/assignment_2
-rw-r--r--   1 acadgild supergroup    2283410 2018-07-19 21:35 /files/movies.csv
-rw-r--r--   1 acadgild supergroup  709550327 2018-07-19 16:06 /files/ratings.csv
-rw-r--r--   1 acadgild supergroup        336 2018-07-19 21:50 /files/test.txt
You have new mail in /var/spool/mail/acadgild
```

```
[acadgild.mmisra ~]$ hadoop fsck /files/movies.csv
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

18/07/20 11:23:17 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Connecting to namenode via http://localhost:50070
FSCK started by acadgild (auth:SIMPLE) from /127.0.0.1 for path /files/movies.csv at
Fri Jul 20 11:23:18 IST 2018
.Status: HEALTHY
 Total size:    2283410 B
 Total dirs:    0
 Total files:   1
 Total symlinks:                0
 Total blocks (validated):      1 (avg. block size 2283410 B)
 Minimally replicated blocks:   1 (100.0 %)
 Over-replicated blocks:        0 (0.0 %)
 Under-replicated blocks:       0 (0.0 %)
 Mis-replicated blocks:         0 (0.0 %)
 Default replication factor:    1
 Average block replication:     1.0
 Corrupt blocks:                0
 Missing replicas:              0 (0.0 %)
 Number of data-nodes:          1
 Number of racks:               1
FSCK ended at Fri Jul 20 11:23:18 IST 2018 in 4 milliseconds


The filesystem under path '/files/movies.csv' is HEALTHY
[acadgild.mmisra ~]$ hadoop fsck /files/ratings.csv
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

18/07/20 11:23:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Connecting to namenode via http://localhost:50070
FSCK started by acadgild (auth:SIMPLE) from /127.0.0.1 for path /files/ratings.csv at
Fri Jul 20 11:23:35 IST 2018
.Status: HEALTHY
 Total size:    709550327 B
 Total dirs:    0
 Total files:   1
 Total symlinks:                0
 Total blocks (validated):      6 (avg. block size 118258387 B)
 Minimally replicated blocks:   6 (100.0 %)
 Over-replicated blocks:        0 (0.0 %)
 Under-replicated blocks:       0 (0.0 %)
 Mis-replicated blocks:         0 (0.0 %)
 Default replication factor:    1
 Average block replication:     1.0
 Corrupt blocks:                0
 Missing replicas:              0 (0.0 %)
 Number of data-nodes:          1
 Number of racks:               1
FSCK ended at Fri Jul 20 11:23:35 IST 2018 in 0 milliseconds


The filesystem under path '/files/ratings.csv' is HEALTHY
[acadgild.mmisra ~]$
```

Answer: 1 block for movies.csv and 6 blocks for ratings.csv

Solution2&3
Below is the pig script for answers 2 and 3 and its explanation

#Load the movies.csv into relation m1 using delimiter ‘,’. The column names are movieid,title and genres. The data type of columns are not specified and left as default which is bytearray.

```
m1 = load '/files/movies.csv' using PigStorage(',') as (movieid,title,genres);
```

#since the file also has the column names as the first line we filter it out and #generate a new relation movies by checking if the row contains ‘movieid as a string

```
m2 = filter m1 by NOT EqualsIgnoreCase(movieid,'movieId');
movies = foreach m2 generate movieid,title;
```

#Load the movies.csv into relation m1 using delimiter ‘,’. The column names are userid,movieid,rating and timestamp. The data type of only rating is specified as that is the only one which needs to be used later

```
r1 = load '/files/ratings.csv' using PigStorage(',') as
(userid,movieid,rating:double,timestamp);
```

#since the file also has the column names as the first line we filter it out and #generate a new relation r2 by checking if the row contains ‘userid as a string

```
r2 = filter r1 by NOT EqualsIgnoreCase(userid,'userId');
```

#we create a new relation ‘relations’ with only those fields which are necessary later for processing, other columns are removed. This is to avoid clutter as well as improving performance by filtering unnecessary columns before processing

```
ratings = foreach r2 generate userid,movieid,rating;
```

#we join both the relations using movieid as the key

```
j= join movies by movieid,ratings by movieid;
```

#we remove the movieid column as that is not required later. We need only userid,title and rating for further processing

```
j1 = foreach j generate $1 as title, $2 as userid, $4 as rating;
```

we group the relation by user ID to find out all the names of the movies which a user has rated.

```
j2 = group j1 by userid;
```

we generate list of movie titles that a user has rated

```
/* generate movie titles that user has rated*/
ans1 =   foreach j2 generate group, j1.title;
```

```
# we store the answer in a folder called assignment_1 on HDFS
store ans1 into '/files/assignment_1' using PigStorage(',');

#we group the relation j1 based on the title to find out how many times a movie has been rated and what is the average rating
j3 = group j1 by title;

#for each movie title we generate count and average of the rating
/* for each title, count how many times it has been rated, and the average
 * rating */
 ans2 = foreach j3 generate group, COUNT(j1.rating), AVG(j1.rating);

# we store the answer in a folder called assignment_2 on HDFS

store ans2 into '/files/assignment_2' using PigStorage(',');
```

 # here is the script

```
[acadgild.mmisra dd]$ cat movie.pig
m1 = load '/files/movies.csv' using PigStorage(',') as (movieid,title,genres);
m2 = filter m1 by NOT EqualsIgnoreCase(movieid,'movieId');
movies = foreach m2 generate movieid,title;

r1 = load '/files/ratings.csv' using PigStorage(',') as
(userid,movieid,rating:double,timestamp:long);
r2 = filter r1 by NOT EqualsIgnoreCase(userid,'userId');
ratings = foreach r2 generate userid,movieid,rating;

j= join movies by movieid,ratings by movieid;

j1 = foreach j generate $1 as title, $2 as userid, $4 as rating;

j2 = group j1 by userid;

/* generate movie titles that user has rated*/
ans1 =  foreach j2 generate group, j1.title;
store ans1 into '/files/assignment_1' using PigStorage(',');

j3 = group j1 by title;
/* for each title, count how many times it has been rated, and the average rating */
ans2 = foreach j3 generate group, COUNT(j1.rating), AVG(j1.rating);

store ans2 into '/files/assignment_2' using PigStorage(',');
```

 # running the script

```
[acadgild.mmisra dd]$ pig -x mapreduce movies.pig
[acadgild.mmisra dd]$
```

 # script output

Once the map reduce job finished successfully we look at the output files

```
[acadgild.mmisra dd]$
[acadgild.mmisra dd]$ hadoop fs -ls /files
18/07/20 11:43:37 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Found 5 items
drwxr-xr-x   - acadgild supergroup          0 2018-07-19 16:38 /files/assignment_1
drwxr-xr-x   - acadgild supergroup          0 2018-07-19 16:41 /files/assignment_2
-rw-r--r--   1 acadgild supergroup    2283410 2018-07-19 21:35 /files/movies.csv
-rw-r--r--   1 acadgild supergroup  709550327 2018-07-19 16:06 /files/ratings.csv
-rw-r--r--   1 acadgild supergroup        336 2018-07-19 21:50 /files/test.txt
```

```
[acadgild.mmisra dd]$ hadoop fs -ls /files/assignment_1
18/07/20 11:43:47 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Found 3 items
-rw-r--r--   1 acadgild supergroup          0 2018-07-19 16:38
/files/assignment_1/_SUCCESS
-rw-r--r--   1 acadgild supergroup  325561317 2018-07-19 16:38
/files/assignment_1/part-r-00000
-rw-r--r--   1 acadgild supergroup  323979113 2018-07-19 16:38
/files/assignment_1/part-r-00001
[acadgild.mmisra dd]$ hadoop fs -ls /files/assignment_2
18/07/20 11:43:51 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Found 3 items
-rw-r--r--   1 acadgild supergroup          0 2018-07-19 16:41
/files/assignment_2/_SUCCESS
-rw-r--r--   1 acadgild supergroup     873636 2018-07-19 16:41
/files/assignment_2/part-r-00000
-rw-r--r--   1 acadgild supergroup     871357 2018-07-19 16:41
/files/assignment_2/part-r-00001
```

#The output for question 2A , What are the movie titles that the user has rated we dump the small portion of file `/files/assignment_1/ part-r-00000`

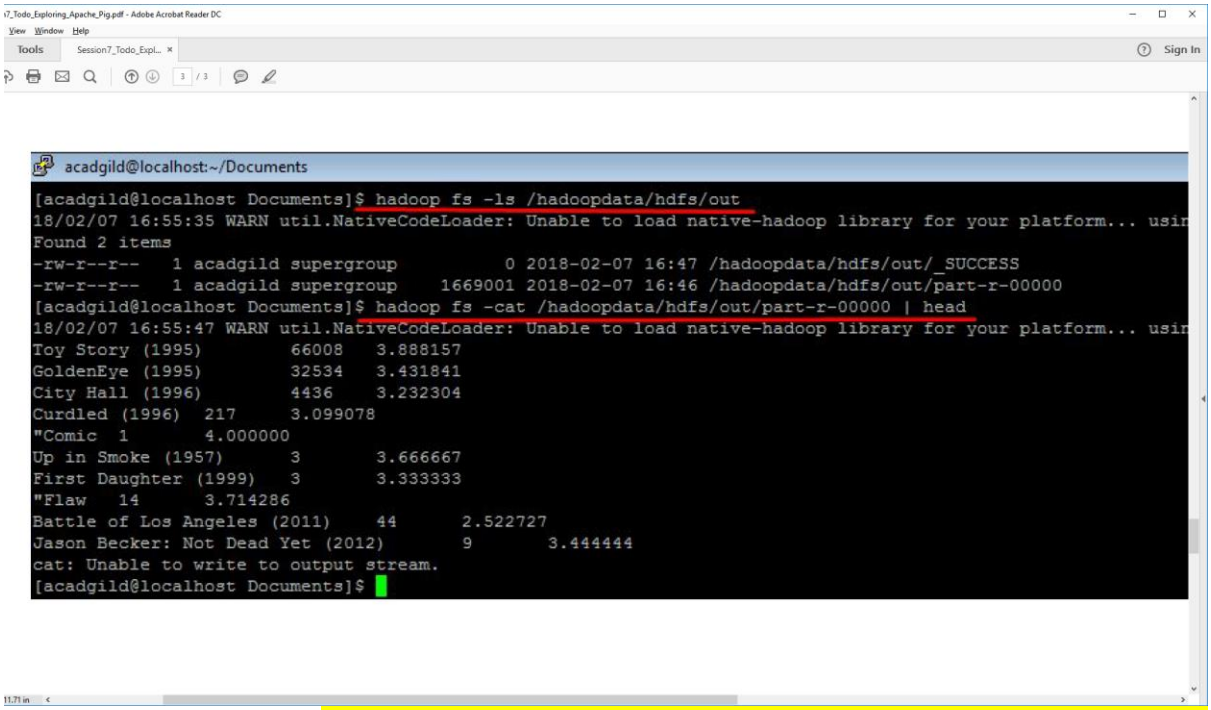
```
[acadgild.mmisra dd]$ hadoop fs -cat /files/assignment_1/part-r-00000 | tail -f
18/07/20 11:45:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
270878,{(Heathers (1989)),(Young Frankenstein (1974)),("Patriot),(Searching for Bobby
Fischer (1993)),(Rory O'Shea Was Here (Inside I'm Dancing) (2004)),(Sleepy Hollow
(1999)),(Dogma (1999)),(Office Space (1999)),(Election (1999)),("Lord of the Rings: The
Return of the King),(Pirates of the Caribbean: The Curse of the Black Pearl
(2003)),(Splash (1984)),(Chasing Amy (1997)),(Gattaca (1997)),(Pleasantville
(1998)),(Little Women (1994)),(Before Sunrise (1995))}
270881,{(First Knight (1995)),("Specialist),(I.Q. (1994)),(Three Colors: Red (Trois
couleurs: Rouge) (1994)),(Three Colors: Blue (Trois couleurs: Bleu) (1993)),(Three
Colors: White (Trzy kolory: Bialy) (1994)),(Stargate (1994)),(From Dusk Till Dawn
(1996)),("Postman),(Mighty Aphrodite (1995)),(Gladiator (2000)),(Do the Right Thing
(1989)),(Ran (1985)),("Shining),(Butch Cassidy and the Sundance Kid (1969)),(Apocalypse
Now (1979)),(Young Frankenstein (1974)),(Dead Poets Society
(1989)),("Graduate),(Cinema Paradiso (Nuovo cinema Paradiso) (1989)),(Monty Python's
Life of Brian (1979)),(Reservoir Dogs (1992)),(Two Much (1995)),(Boys (1996)),(1984
(Nineteen Eighty-Four) (1984)),(From Here to Eternity (1953)),("Breakfast Club),(Austin
Powers: International Man of Mystery (1997)),(Scream (1996)),(Seven Years in Tibet
(1997)),("Blair Witch Project),("Man and a Woman),(Airplane! (1980)),(Big
(1988)),("Color Purple),(Who Framed Roger Rabbit? (1988)),(Sweet Charity (1969)),("Lost
Honor of Katharina Blum),(Salaam Bombay! (1988)),(Green Card (1990)),(Day for Night (La
Nuit Américaine) (1973)),(Europa Europa (Hitlerjunge Salomon) (1990)),(Straw Dogs
(1971)),(Fahrenheit 451 (1966))}
270883,{(Die Hard (1988)),(Apollo 13 (1995)),(Taxi Driver (1976)),(Forget Paris
(1995)),(Léon: The Professional (a.k.a. The Professional) (Léon) (1994)),(Nobody's Fool
(1994)),(Outbreak (1995)),(Die Hard: With a Vengeance (1995)),(Four Weddings and a
Funeral (1994)),("Client),(Clear and Present Danger (1994)),(Forrest Gump
(1994)),(Dances with Wolves (1990)),(Phenomenon (1996)),(Batman (1989)),("Silence of
the Lambs),(Pretty Woman (1990)),(Sleepless in Seattle (1993)),(Speed
(1994)),("Firm),("Fugitive),(Maverick (1994)),(True Lies (1994)),(Dave (1993)),(In the
Line of Fire (1993)),(Mrs. Doubtfire (1993))}
```

#The output for question 2B and 3, How many times a movie has been rated by the user and what is the average rating given for a movie. We dump a part of file /files/assignment_2/ part-r-00000

```
[acadgild.mmisra dd]$
[acadgild.mmisra dd]$
[acadgild.mmisra dd]$
[acadgild.mmisra dd]$ hadoop fs -cat /files/assignment_2/part-r-00000 | tail -f
18/07/20 11:46:13 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Gurren Lagann: The Lights in the Sky are Stars (Gekijô ban Tengen toppa guren ragan:
Ragan hen) (2009),20,3.6
Investigation of a Citizen Above Suspicion (Indagine su un cittadino al di sopra di ogni
sospetto) (1970),126,3.8253968253968256
Dragon Ball Z: Broly Second Coming (Doragon bôru Z 10: Kiken na futari! Sûpâ senshi wa
nemurenai) (1994),33,3.0757575757575757
Riuscirà l'avvocato Franco Benenato a sconfiggere il suo acerrimo nemico il pretore
Ciccio De Ingras? (1971),1,3.0
Victory in the Ukraine and the Expulsion of the Germans from the Boundaries of the
Ukrainian Soviet Earth (1945),2,1.75
To Each His Own Cinema (Chacun son cinéma ou Ce petit coup au coeur quand la lumière
s'éteint et que le film commence) (2007),47,3.3297872340425534
Dragon Ball Z: The History of Trunks (Doragon bôru Z: Zetsubô e no hankô!! Nokosareta
chô senshi - Gohan to Torankusu) (1993),64,3.609375
More About the Children of Noisy Village (a.k.a. More About the Children of Bullerby
Village) (Mer om oss barn i Bullerbyn) (1987),3,3.5
Revolutionary Girl Utena: Adolescence of Utena (a.k.a. Revolutionary Girl Utena the
Movie) (Shoujo kakumei Utena: Adolescence mokushiroku) (1999),31,3.629032258064516
Dragon Ball Z the Movie: The World's Strongest (a.k.a. Dragon Ball Z: The Strongest Guy
in The World) (Doragon bôru Z: Kono yo de ichiban tsuyoi yatsu)
(1990),43,3.0348837209302326
[acadgild.mmisra dd]$
```

 # Checking correctness of the output
We can compare the results of the map reduce program (given as part of assignment) with what we got after pig script run

Below is the MR output given in the assignment. We can check that count and average rating for the same movie matches with our output




```
[acadgild.mmisra dd]$ hadoop fs -cat /files/assignment_2/part-r-00000 | grep "Toy Story (1995)"
18/07/20 12:00:23 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Toy Story (1995),66008,3.8881574960610834
You have new mail in /var/spool/mail/acadgild
[acadgild.mmisra dd]$ hadoop fs -cat /files/assignment_2/part-r-00000 | grep "GoldenEye (1995)"
18/07/20 12:00:46 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
[acadgild.mmisra dd]$ hadoop fs -cat /files/assignment_2/part-r-00001 | grep "GoldenEye (1995)"
18/07/20 12:00:51 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
GoldenEye (1995),32534,3.431840536054589
[acadgild.mmisra dd]$ hadoop fs -cat /files/assignment_2/part-r-00001 | grep "City Hall (1996)"
18/07/20 12:01:18 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
[acadgild.mmisra dd]$ hadoop fs -cat /files/assignment_2/part-r-00000 | grep "City Hall (1996)"
18/07/20 12:01:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
City Hall (1996),4436,3.232303877366997
You have new mail in /var/spool/mail/acadgild
[acadgild.mmisra dd]$
```

 # We see that both count and average rating match of the specific movies