

Hbase:

Note: for this question, because we need to parse email contents, some assumptions are made:

- The date corresponds to the value after Date:, sender corresponds to the value after From:, sendee corresponds to value after To: (some emails do not have this field, even though they might have 'X-To' field, we will assume the sendee as empty in this case)
- We assume only email body and sendee could have multiple lines and all other fields are only one-line fields.

1. Hbase model

Row key by timestamp, column family by employee name, each column family has columns email body, sender, sendee (send to).

2. Ingest and query script

Please see the attached file hw4_hbase.py, if you want to run this script, be sure to first delete or rename the table.

3-5. Results

Please see the attached output files hbase_output1.txt, hbase_output2.txt, hbase_output3.txt.

Hive:

1. Create ratings table

```
CREATE TABLE ratings (userid INT, movieid INT, rating FLOAT, timestamp INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

2. Load ratings data from local path

```
LOAD DATA LOCAL INPATH '/home/public/recommendationEngine' OVERWRITE INTO TABLE ratings;
```

3. Random select a user and identify his/her favorite movie

```
SELECT * FROM ratings WHERE userid=1 ORDER BY rating DESC LIMIT 1;
```

1	296	5.0	1147880044
---	-----	-----	------------

We want to make recommendations to userid 1 based on his/her favorite movieid 296

4. Create movie pairs frequency table

```
CREATE TABLE pairs(movie1 INT, movie2 INT, frequency INT);
```

5. Insert data into pairs table by querying from ratings table

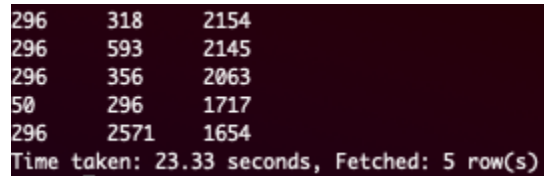
```
INSERT OVERWRITE TABLE pairs SELECT t.movie1 AS movie1, t.movie2 AS movie2, COUNT(*) AS frequency FROM (SELECT r1.movieid AS movie1, r2.movieid AS movie2 FROM ratings r1 INNER JOIN ratings r2 ON
```

```
r1.userid = r2.userid WHERE r1.movieid<r2.movieid AND r1.rating>=3 and  
r2.rating>=3) t GROUP BY t.movie1, t.movie2 ORDER BY frequency desc;
```

(The use of subquery is kind of redundant in this case, but when I try to redo it, neither thunder or wolf is running correctly, so I'll keep the results as it is)

4. Get recommendations

```
SELECT * FROM pairs WHERE movie1 = 296 OR movie2=296 ORDER BY  
frequency DESC LIMIT 5;
```



A terminal window with a dark background and light-colored text. It displays the results of a SQL query. The first five rows show movie IDs and their frequencies. The last line shows the execution time and the number of rows fetched.

296	318	2154
296	593	2145
296	356	2063
50	296	1717
296	2571	1654

Time taken: 23.33 seconds, Fetched: 5 row(s)

So we recommend movies 318, 593, 356, 50 and 2571 to user 1.