BI Assignment 3

0427561 - Bernhard Müller

February 5, 2017

1 MovieLens dataset analysis with Hive

Each subsection presents the query and resulting rows. Additionally the discussion regarding mapreduce tasks for each query is added. Finally we give a short discussion on the possible scale up on a larger cluster [1]

1.1 Query 1 - How many movies are there in total in the dataset?

```
SELECT COUNT(*)
FROM movies;
```

Result: 27278

As the movies data set is very small (2MB) only one mapper is used to select the data. The reducer counts all rows an finishes.

1.2 Query 2 - How many movies in the dataset belong to the "Film-Noir" genre?

```
SELECT COUNT(*)
FROM movies
WHERE array_contains(split(genres,'\\|'), 'Film-Noir');
```

Result: 330

This query is similar to the first one with an extra filter for the data rows. This filter can be implemented in the mapper and therefore again, only one mapper is used to select the data. The reducer counts all rows an finishes.

1.3 Query 3 - Which are the 10 most frequently assigned tags (by users, i.e., from the tags table)?

```
SELECT tag_ranked
FROM
  (SELECT tag AS tag_ranked,
          count(tag) AS tag_count
  FROM tags
  GROUP BY tag
  ORDER BY tag_count DESC limit 10) r;
```

Result:

[1] This assignment was executed on a Cloudera Virtualbox VM with a single node.

Tag
sci-fi
based on a book
atmospheric
comedy
action
surreal
BD-R
twist ending
funny
dystopia

This query has to be split into two tasks. The first mapper solves the inner SELECT statement, except the LIMIT statement. The limiting to 10 rows and selection of the single output column can be done in the second job.

1.4 Query 4 - Which 10 movies were the most controversial in 2015 (i.e., had the highest variance in ratings between 2015/01/01 and 2015/12/31)?

```
SELECT title
FROM
  (SELECT title,
          variance(rating) AS rating_var
   FROM movies
   LEFT JOIN ratings
       ON (movies.movieId = ratings.movieId)
   WHERE year(from_unixtime(ratings.timestamp)) = 2015
   GROUP BY  title
   ORDER BY  rating_var DESC LIMIT 10) t;
```

Result:

Title
Rent-a-Kid (1995)
Harder They Come, The (1973)
Docks of New York, The (1928)
A Cinderella Story: Once Upon a Song (2011)
Born in Flames (1983)
Elle: A Modern Cinderella Tale (2011)
Conspirators of Pleasure (Spiklenci slasti) (1996)
The War at Home (1979)
Jesse Stone: Sea Change (2007)
Cry_Wolf (a.k.a. Cry Wolf) (2005)

This query will be split into 3 tasks. At first the tables movies and ratings will be joined and filtered for ratings from 2015. This data set gets aggregated by title and variance of rating calculated. The last task is again sorting the data descending, limiting the output to 10 rows and selecting the title column exclusively.

1.5 Query 5 - Which movies (titles) are the 10 most frequently tagged and how often have they been tagged?

```
SELECT title,
          count (tag) AS tag_count
FROM movies
LEFT JOIN tags
    ON (movies.movieId = tags.movieId)
GROUP BY  title
ORDER BY  tag_count DESC LIMIT 10;
```

Result:

Title	Tag count
Pulp Fiction (1994)	1994
Fight Club (1999)	1779
Inception (2010)	1552
Matrix, The (1999)	1430
Shawshank Redemption, The (1994)	1339
Eternal Sunshine of the Spotless Mind (2004)	1240
Donnie Darko (2001)	1177
Memento (2000)	1168
Silence of the Lambs, The (1991)	1100
Avatar (2009)	995

The tables have to be joined in a first job. Afterwards the data can be grouped and the count field aggregated. After ordering the last job is to limit the output to 10 rows.

By discovering the GUI we have found the EXPLAIN option. According to this output the JOIN operation is performed by a local task. We assume this is because of the small size of the tags and the movies table - only few megabyte. This gave a little bit more insight on the CLI output.

1.6 Query 6 - Which 15 movies (titles) have been most frequently tagged with the label "mars"?

```
SELECT title
FROM
  (SELECT title,
          count(tag) AS tag_count
   FROM movies
   LEFT JOIN tags
       ON (movies.movieId = tags.movieId)
   WHERE tag = 'mars'
   GROUP BY title
   ORDER BY tag_count DESC LIMIT 15) t;
```

Result:

Title
Mars Attacks! (1996)
War of the Worlds, The (1953)
Total Recall (2012)
Total Recall (1990)
Capricorn One (1978)
Martian Child (2007)
It Came from Outer Space (1953)
Day the Earth Stood Still, The (1951)
Mission to Mars (2000)
6th Day, The (2000)
RocketMan (a.k.a. Rocket Man) (1997)
Destination Moon (1950)
Red Planet (2000)
Impostor (2002)
Doom (2005)

This query is very similar to query 5, except with an additional WHERE clause. This filter can be added to the reduction job of the JOIN operation, so no additional job is needed.

1.7 Query 7 - Which are the 10 best-rated movies (on average; list titles) with more than 1000 ratings?

```
SELECT title
FROM
```

```
(SELECT title,
        t.rating_count
  FROM
    (SELECT title,
          avg(rating) AS rating_av,
          count(rating) AS rating_count
     FROM movies
     LEFT JOIN ratings
         ON (movies.movieId = ratings.movieId)
     GROUP BY title
     ORDER BY rating_av DESC) t
     ORDER BY t.rating_count desc) f LIMIT 10;
```

Result:

Title
Pulp Fiction (1994)
Forrest Gump (1994)
Shawshank Redemption, The (1994)
Silence of the Lambs, The (1991)
Jurassic Park (1993)
Star Wars: Episode IV - A New Hope (1977)
Braveheart (1995)
Terminator 2: Judgment Day (1991)
Matrix, The (1999)
Schindler's List (1993)

The ratings table is rather large (around 500MB), therefore the JOIN operation can be split over multiple mappers and reducers.

1.8 Query 8 - Which are the highest-rated "Film-Noir" movies with more than 10 ratings (average rating; movies with genre "Film-Noir", max. 10)?

```
SELECT title
FROM
  (SELECT title,
          avg(rating) AS rating_av,
          count(rating) AS rating_count
   FROM movies
   LEFT JOIN ratings
       ON (movies.movieId = ratings.movieId)
   WHERE array_contains(split(genres,'\\|'), 'Film-Noir')
   GROUP BY title
   ORDER BY rating_av DESC) t
WHERE rating_count > 10 LIMIT 10;
```

Result:

Title
Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)
Third Man, The (1949)
Double Indemnity (1944)
Big Sleep, The (1946)
Chinatown (1974)
Notorious (1946)
M (1931)
Maltese Falcon, The (1941)
Touch of Evil (1958)
Strangers on a Train (1951)

The first task will be again the JOIN operation and filter for the genre 'Film-Noir'. The second

task is the aggregation of average and count. Additionally the data will be filtered on the calculated count.Finally the output column title is selected and rows are limited.

1.9 Query 9 - What are the 15 most relevant genome tags for the movie "Toy Story (1995)" (movieId=1)?

```
SELECT tag
FROM
   (SELECT tag,
           g.relevance
   FROM
      (SELECT tagID,
              relevance
      FROM genome_scores
      WHERE movieId = 1) g
      LEFT JOIN genome_tags
          ON (genome_tags.tagId = g.tagId)
      ORDER BY  g.relevance DESC) f LIMIT 15;
```

Result:

Genome Tag
toys
computer animation
pixar animation
kids and family
animation
kids
pixar
children
cartoon
imdb top 250
animated
childhood
great movie
disney animated feature
friendship

By filtering the genome_scores table first in a subquery and join it afterwards, we could reduce the overall execution time by 25%. Hive reduces this query to a single job. But uses to mappers because of the large table genome_scores.

1.10 Query 10 - Which are the 10 most relevant movies for Vienna (i.e., with the highest genome tag relevance rating for the tag "vienna")?

```
SELECT title
FROM
   (SELECT movieId,
           g.tag,
           relevance
   FROM
      (SELECT tagId,
              tag
      FROM genome_tags
      WHERE tag = 'vienna') g
      LEFT JOIN genome_scores
          ON (genome_scores.tagId = g.tagId)
      ORDER BY  relevance DESC) t
   LEFT JOIN movies
      ON (t.movieId = movies.movieId) LIMIT 15;
```

Result:

Title
Third Man, The (1949)
Johnny Guitar (1954)
Before Sunrise (1995)
Before Sunset (2004)
Before Midnight (2013)
Night Porter, The (Portiere di notte, Il) (1974)
Illusionist, The (2006)
Amadeus (1984)
Foreign Affair, A (1948)
Love in the Afternoon (1957)
Odessa File, The (1974)
Bad Timing: A Sensual Obsession (1980)
Best Offer, The (Migliore offerta, La) (2013)
Odd Man Out (1947)
Stranger, The (1946)

For this final query a join task with filtering for the tag 'vienna' will be generated first. The resulting rows are joint with the movies table. Finally in the third task the output column is selected an limited to 15 rows.

1.11 Scale Up

The effects of scale up should not be very significant because of the relatively small size of the data sets. Queries containing the genome-scores or ratings table can finish in the half time. But all queries in this assignment have jobs with only a single mapper or reducer, which would not benefit from a larger cluster.

Even for HDFS with a default chunk size (Cloudera) of only 128 MB is over sized for this tasks.

But Hive is intended for much larger datasets starting at the level of multiple gigabytes. Therefore the small numbers of mappers in this examples are comprehensible. A to fast increase of mappers for small dataset would result in too much mappers for large data sets producing a large overhead and poor performance of the whole system.