TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Dr. Thomas Neumann

Exercises for *Foundations in Data Engineering*, WiSe 23/24

Alexander Beischl, Maximilian Reif (i3fde@in.tum.de)
http://db.in.tum.de/teaching/ws2324/foundationsde

Sheet Nr. 09

Exercise 1 Order the following workloads by how well they are suited for cluster computing. Assume a cluster of machines which are connected via ethernet and that the dataset is distributed onto the machines when the workload starts. A workload that is well suited for cluster computing decreases in execution time proportional to the number of machines in the cluster.

- Search in text documents
- The toy example from the lecture (Basic Building Blocks, slide 26-35)
- Sorting records
- Breadth-first search
- Join of two relations (both relations to big for one machine)
- Shuffling a dataset

Exercise 2

Formulate map-reduce programs to handle the following tasks:

1. For the multi-sets $X : [(a, b)], Y : [(c, d)]$ find all combinations where $b = c$.
2. For the documents $D : [(name, [w])]$ (where D is a list of documents, in which each document is a list of words w), find the words which all documents have in common.
3. Compute AB for the two matrices A and B.

Exercise 3 This exercise is about getting familiar with the Spark Dataset API.

To get started, open a spark shell and load the *song dataset* into a dataset. The data is an excerpt from the *Million Song Dataset.* Make sure that appropriate data types are used for each column. You can check the types e.g. with the `printSchema` function.

For a list of functions offered by the dataset API please refer to the *dataset documentation.* To get started with importing a csv file, have a look at *this tutorial.*

Once the dataset is loaded, use the API to answer the following questions:

1. List all songs from the year 2000.
2. In which year is the first song from this dataset published? (For some songs in the dataset, the year is set to 0. In this case, the year is unknown.)
3. Which artist is the hottest, according to the `artist_hotttnesss` column?
4. Which album has most songs on it? (Use release text and artist to identify album.)
5. Find song pairs with equally familiar artists. That is: `a.artist_familiarity = b.artist_familiarity` How many pairs are there?

6. Find song pairs with similarly familiar artists.
   That is: $|\texttt{a.artist\_familiarity} - \texttt{b.artist\_familiarity}| < 0.001$
   How many pairs are there? You may notice, that when trying to run this query, it does not terminate (in reasonable time). Call `explain` on the final dataset. This will show the execution plan which will be used to retrieve the result. Most likely, the root node of the plan you are seeing is a `CarthesianProduct` operator. It produces a cross product of all its input and applies a filter operation on the result. Thus, it is quite slow for larger amounts of data. Try to reformulate the query so that no `CarthesianProduct` is used.