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

Exercises for *Foundations in Data Engineering*, WiSe 19/20

Timo Kersten (kersten@in.tum.de)
http://db.in.tum.de/teaching/

Sheet Nr. bonus2

Bonus Project 2

Task

This project is designed to give you an opportunity to gain experience in programming systems in the Hadoop ecosystem. In this case, we use Spark to analyze *taxi rides within New York*.

We will use a data set which covers one month. You will find time and location for each trip's start and end. In the following, this is the data that is meant when we refer to a trip.

The general question is: Can we match trips and return trips? For a given trip a, we consider another trip b as a return trip iff

1. b's pickup time is within 8 hours after a's dropoff time
2. b's pickup location is within r meters of a's dropoff location
3. b's dropoff location is within r meters of a's pickup location

where r is a variable distance in meters that is specified as input to the query. In this project we use values between 50 and 200.

To compute the return trips, you may want to break the problem down into the following series of problems:

1. Given the (lat,lon) coordinates
   - $a(40.79670715332031, -73.97093963623047)$
   - $b(40.789649963378906, -73.94803619384766)$
   - $c(40.73122024536133, -73.9823226928711)$

   which trips have dropoff locations within r meters of a,b or c?
2. For each trip a in the dataset, compute the trips that have a pickup location within r meters of a's dropoff location. These are the return trip candidates.
3. For all trips a in the dataset, compute all trips that may have been return trips for a.

Another way to characterize the dataset to be returned would be this pseudo SQL:

```
select *
from
  tripsProvided a,
  tripsProvided b
where
  distance(a.dropofflocation, b.pickuplocation) < r and
```

```
    distance(b.dropofflocation, a.pickuplocation) < r and
    a.dropofftime < b.pickuptime and
    a.dropofftime + 8 hours > b.pickuptime
```

For distance calculations, assume that the earth is a sphere with radius 6371km. Numerical stability of appropriate formulas is discussed e.g. at https://en.wikipedia.org/wiki/Great-circle_distance.

To complete this project, submit an implementation that manages to compute this query in **less than 10 minutes**. The machine used for evaluation will have 64GB RAM and a 6 core Intel i7-3930K processor.

How to submit

Clone the project at https://gitlab.db.in.tum.de/kersten/fde19-bonusproject-2.

Put your implementation into ReturnTrips.scala and assure that `ReturnTrips.compute(tripsProvided, dist, sparkContext)` returns a dataset with all trips and their return trips. That means, each row in the returned dataset must contain a trip and a return trip, so that all trips in `tripsProvided` are returned with their return trips in case they have any.

To try if your implemenation works, install the prerequisites and run `test.sh`. Alternatively, load the trips.scala file into a spark-shell to examine the dataset interactively:

```
spark-shell -i trips.scala
```

If you implementation passes test.sh, commit and push your solution to Gitlab. This will measure the performance and submit the measured time to contest.db.in.tum.de.

Prerequisites

1. **Spark:** In order to run Spark locally, please consider the install instructions from the Spark project: https://spark.apache.org/docs/2.2.0/.
2. **Sbt:** The Scala package manager. http://www.scala-sbt.org/1.0/docs/Setup.html

Learning how to use Spark

1. *An Overview of APIs available in Spark.* This is also a description of advantages that the Dataset API offers.
2. *A getting started guide to the Dataset API.*
3. *API Documentation.* Especially read about the functions **join, select, filter, withColumn, as, explain**.