Lorenzo Scaturchio

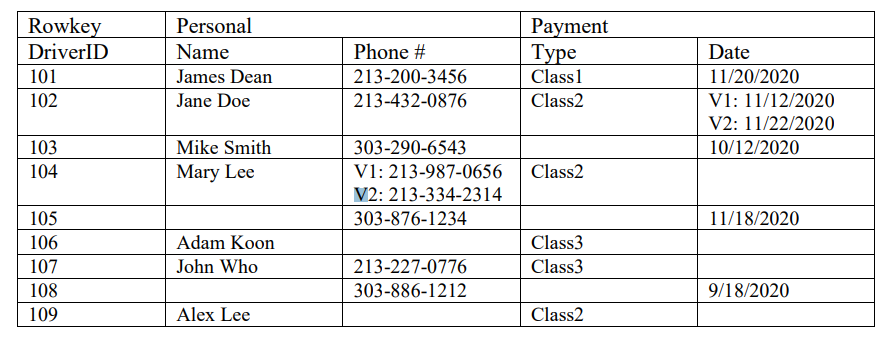
17 November 2022

Homework 6

1. What is sparse data and why is it a problem in a big table based on space and time complexity?

Sparse data is known to be the most unused data in a database / dataset. The reason why it is a problem is because it could create bottlenecks for running models on the data. For example lets say there is an optimal algorithm for sorting through a sparse dataset, even running optimally at a big O(n) or O(1) the space complexity would still be an issue that needs to be resolved. So with that in mind, we want to optimize sparse data as much as possible.

1. Explain how HBase distributes this table over multiple machines in Hadoop cluster. Assume two regions where DriverID 101-103 go to region 1, DriverID 104-106 go to region 2, and DriverID 107-109 go to region 3. You answer needs to provide the details of physical files (i.e, actual content of a file).



The physical files of DriverID 101 - 103 go to region 1 would have to first be assigned by the master node which is done automatically. Within the region, the three IDs would be split into a column family where it would then be sent to the HDFS from there. This set of steps would have to be done for each of the three regions where the ID’s are split up in groups of three. The structure of these tables would be flat-wide tables and have associated keys to easily access any of the objects. When iterating through each of the regions, if we had a lookup for any particular portion of a region, hbase would have to iterate one by one due to the row-oriented approach.

1. What is scale-out approach in NoSQL? Why is this a better scalability solution than scale-up solution of SQL?

The scale-out approach in NoSQL is when you add more of the same types of servers instead of upgrading the same number of servers for your solution. The reason why the scale-out approach is better than the scale-up approach is because we want to avoid a single point of failure, we also don’t want to send a significant amount of data transfer through one link so it is important to distribute that out to multiple servers, hence, scaling-out.

1. When would you choose a NoSQL database over SQL? When not to choose a NoSQL database?

I would choose a NoSQL database over SQL is: sharding. Since NoSQL documents are typically self-containerd, we don’t have to worry about joining rows from multiple databases. Another big factor would have to be scalability which makes NoSQL a lot faster for servers when the throughput could be transferred to multiple nodes. I would choose SQL over SQL when I want reliability. There is no ACID compliance when it comes down to NoSQL which is a giant plus for SQL because that allows for the language to perform complex transactions. The second reason would be persistence. If I wanted a database to be very static and not have many changes in my server, then SQL would be the best way to go.