**Question 1**

There are two main goals to scoring a driver.

1. Ensure that the target shipment will be delivered reliably and on time.
   1. Distance from driver’s location to shipment origin.
   2. Time from accepting offer to pickup of shipment.
   3. Frequency and severity of mechanical and other issues.
   4. Whether the driver was recently offered a job (don’t flood drivers with too many requests)
2. Maximize overall throughput and minimize latency of securing drivers for shipments in the future.
   1. Incentives for suggesting other drivers that sign up for the service (network effects).
   2. Higher incentives for drivers with more successful shipments delivered.
   3. Combine multiple shipments if their origin and destination are compatible and the sum of their required capacities is under the total capacity. This can also be implemented in such a way that a driver can accept secondary shipments on the fly. Let’s say a driver has capacity 50 and just picked up a shipment at point A with capacity 30. His updated capacity will be 20, and we can still send him an offer for a shipment of size <= 20 if its origin and destination will not be too far off route. “Too far of route” can be made more rigorous by taking into account absolute distance as well as the time-sensitivity of the existing shipments the driver currently has.
   4. Percentage of offers driver accepts (though low accept percentage can also hint at deficiencies in scoring drivers).

We would want to gather statistics about the offers we send out right from the beginning. This would give us understanding of the limitations of the basic system and guide us in prioritizing new features. With that said, it’s very likely that incorporating the current location of the driver (and finding the nearest drivers to a given shipment) would be an important and beneficial improvement early on.

If, for instance, drivers were frequently picking up shipments and had a large amount of remaining capacity, it would make sense to prioritize combining shipments, but only if the co-location of shipment origins and destination was high enough to warrant this.

If many offers were rejected or delivered late, we would want to implement some other reliability scores from (1) above. The initial V1 system didn’t have a notion of time for shipments, which probably would be relevant in a real scenario. Shipments would likely come with some sort of destination deadline which can further guide us in providing reliable service.

**Question 2**

If we design a system where frequently adding new driver score types was a priority, we would want to ensure that:

1. It’s simple and efficient to add new score types
2. Querying for the best driver for a given shipment would be efficient regardless of the current score type
   1. This means not designing our system rigidly (with respect to the data model and data structures) around score types that we currently use.

Our V1 system used the capacity of the driver as the first “partitioning key” in scoring drivers. We then performed a linear scan of all drivers to meet the other conditions.

In a real application with many drivers and regions, it is very likely that the initial partitioning would be best done by driver location relative to the shipment. We could use spatial indexing (an R-Tree, for example) to store and find drivers near a shipment. We could then scan and score the (likely small) number of drivers that are near the shipment.

If shipments were scheduled far ahead, we could use knowledge of the future location of drivers in our network from other accepted offers to estimate which drivers would be near the origin of the shipment when it would be ready. It would also allow us to schedule more “tightly,” since we would have a large amount of shipment information ahead of time. This would also integrate well with drivers carrying multiple shipments at a time, since we could approximate a “Traveling Salesman” path in these cases.