How to scale the uber database?

Use sharding here, shard by the trip id.

**How does surge pricing work? And why is spark needed for this?**

Higher demand in an area, the price will increase here. SO how does this work?

Can implement a streaming pipeline to process the orders?

Store the order demand in redis here, we use spark to take in big data (and then do processing query, grouping of data) A diagram of a diagram

Description automatically generated

And then push stuff to our redis data here

1. Spark will aggregate the data and then push to our redis here

A diagram of a computer

Description automatically generated

How do we calculate the ETAs?

1. We can use google maps?

How do we efficiently find the drivers in an area?

Use H3 geospatial indexing

H3 divides Earth’s surface into cells on a flat grid, giving each cell a unique identifier with a 64-bit integer.

An H3 index is a reference to a specific cell on the grid.

Using longitutde and latitude is very slow here as said

algorithm that Uber developed called H3 and what H3 does is it splits the world

into a bunch of hexagonal areas and it assigns an ID to each one of these cells then it groups all of the drivers into these cells

**One cell can be used to look up adjacnet cell ids**

interesting point is that one

sale ID can be used to calculate

adjacent cell IDs so if I'm in cell D I

can look up all the other cells around

it based on that

so if I'm a rider

located here I can simply look up all

seven of these hexagons then I can draw

a circle around me and I can find all of

the rides that are within that radius

then using the ETA service that we have

here we can look up the ETA for each one

of those and find out which one is

closest to us now

How does uber handle the crazy writes updates?

1. It’s done with Cassandra usually here,

Redis stores the recent locations of each driver. Also it buffers enough data points to do map matching.