For 2.5 real-time adaptation, in which I am asked that if my dataset were 'live' and it has live updates on the database how would I set up my architecture to incorporate this real time feed. Firstly, some general guidelines for this live time-series data I would be using tall and narrow tables as storing one event per row makes it easier to run queries against my data. As such I would also be normalising my dataset, splitting the data into smaller tables with relationships between tables(dates) as this would reduce the memory used when dataset is very large. If my dataset got very large now it is incorporating more values for my dataset every day (could be seconds), I would set up an aggregate table(s) for existing data and to keep track of incoming events separately I.e., different queues.

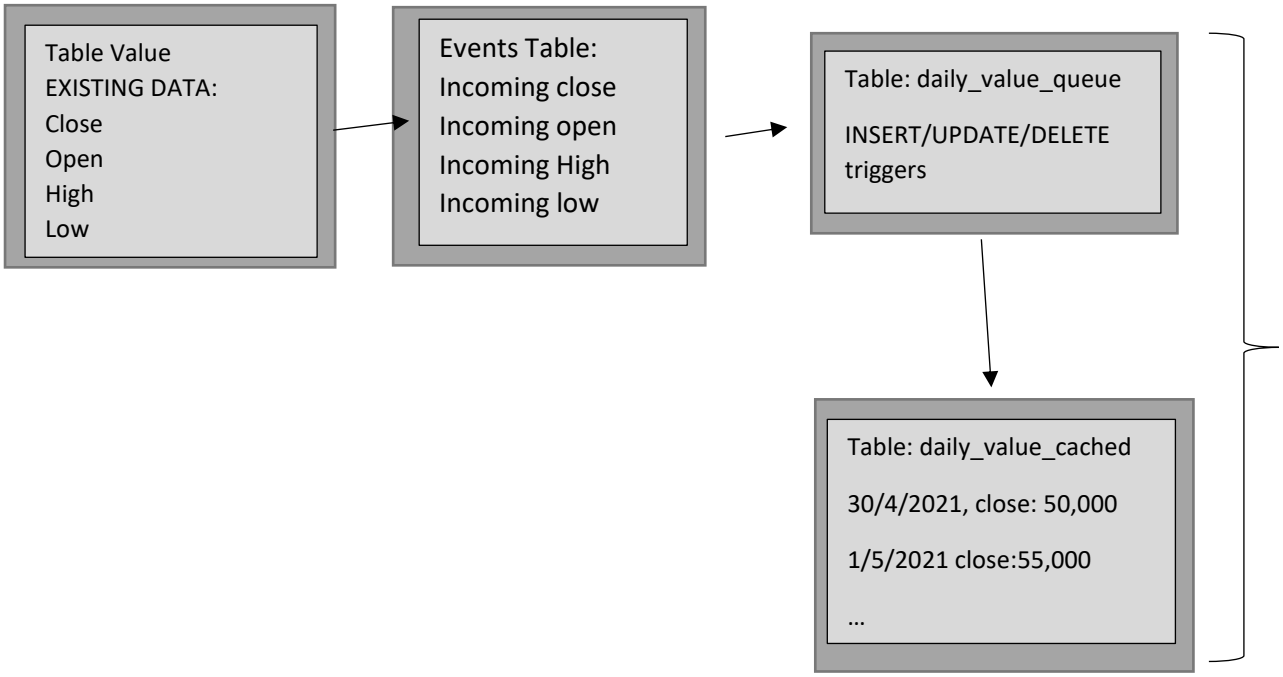

Diagram Explaining Architecture

The above diagram explains how I would implement my new architecture, when my data is live and becomes very large due to the volume of data. As I show the table daily_value_cached contains the aggregated data from the event table while the daily_value_queue is a holder for logging the incoming values which have not been introduced into the cached table. Using the INSERT/UPDATE/DELETE triggers listening on events, it is possible to keep up to date with the values. This data would then be introduced to the existing table of values when the database is not under work from the live data. The reason for the queue table is for future proofing as I do not have to implement this but, when the events are incoming from different machines, each machine would be trying to modify the same rows in the cached and would result in row locking which decreases the performance of the database.