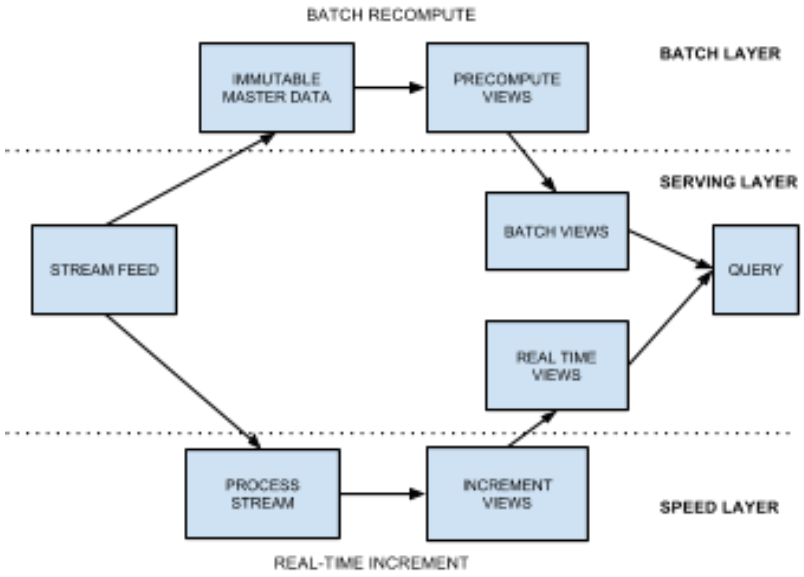
A Call for Sanity in NoSQL: Lambda Architecture

Following are the 2 key drawbacks of NoSQL databases, which affect the robustness of the system:
1) One of the biggest drawbacks of the NoSQL database is not schema check. This might lead to a lot of inconsistent data in the data store due to CRUD operations.
2) NoSQL databases also don't have ACID properties. Which means data at any point of time might not be in consistent state, but eventually can come to consistent state. Also the algorithms which try to achieve this are very complex and error prone.

To overcome these drawbacks and to make the system more reliable, Nathan Marz came up with a better architecture known as Lambda Architecture. Lambda architecture is meant to be more robust, fault tolerant to both human errors and machine failures and scalable to suit real time applications. It mainly comprises of 3 layers as shown in the figure. The batch layer mainly consists of an immutable data store and batch processing which generates views depending on applications for faster real time query processing. Also the batch layer provides high levels of accuracy. The speed layer is a real-time stream-processing layer, which only processes data that is still not updated in the batch layer. Finally serving layer uses these real-time and batch views for faster real time querying.



Lambda architecture is great for real time robust applications. Some applications include social analysis, credit card fraud detection, weather analysis, stock market analysis and so on. All of these applications mainly need to perform analysis over data in long period of time typically in months and years and also have to consider the current real time data like current tweets, current weather and accidents data, current stock prices and trading's, and so on. The final queries might include predict the number of accidents that can happen in next few days, predict the stocks that might gain over next few weeks, good to buy stocks or good to sell stocks, latest trending topics in Facebook and Twitter.

Even though there are some great advantages of this architecture, there are a few disadvantages as well. One of the key disadvantages is that the architecture is too complex. It involves integration of a lot of technology stack. Like the implementation of batch layer will be on different technology stack like Hadoop, hive, Hbase and pig while the real time stack can be on Kafka and Storm. Another major overhead is processing the input data twice, one for the real-time layer and another for batch processing layer. This might reduce the efficiency of the system.