Use cases of HBase

HBase in Production
First we have Groupon. Remember, it's some of the companies, we are not going to take it through a list of the entire companies that are using HBase. It would become quite exhaustive and beyond the scope of this course work. Groupon, this basically a deal of the day website that basically features discounted gift certificates or gift vouchers that are usable at a local merchant or at national level companies. And they have increasingly try to tune the deals to be more accurate and relevant to the users that is to provide some more personalized services.

The Groupon user history and the user profiles is basically store as a data on the tables with very wide schemas and they are actually sparsely populated tables because of this, Groupon runs the personalization engine or service on HBase to get the required real time look ups on the history and the profile information to make the users experience smoother.

Then we have Meetup. Meetup, as we know is basically is on a mission to help the world's people to self-organize into local groups. I am sure we must have used Meetup and probably we are members to one or more Meetup groups. Meetup uses Hadoop and HBase to power a site-wide real time activity feet system for all their members and groups. There are around 20 million members in Meetup and there have more than 1,70,000 Meetup groups. So the user base is huge and the main requirement of the Meetup is to ensure that there is a real time activity feed for all their users.

The Groupon activity is directly return to HBase and it's indexed per member with the members custom feed service directly from the HBase for every incoming request. Remember we had mentioned earlier that HBase is used in cases where you would want to get one or few rows out of a billion rows.

This is one scenario in real time when the users log in to the system and it has to look up may be a billion row table and get only those few groups or activities that are of interest to the users.

Finally, and perhaps, one of the heaviest users of HBases is Facebook. You would have probably read somewhere that Facebook has introduced or you could even be using that the Facebook's new Social Inbox which basically integrates your emails, instant messages, SMS, text messages in one single site. Facebook messages what it is called as. All in all they need to store over 150 billion messages in a month and this was a statics as of 2015. That's a huge number and that one is really critical for Facebook and all these messages are actually stored in HBase.

Now what do we do with this? We will let us look at some of the requirements of the messaging system and how Facebook addressed it.

The typical requirements of messaging system are that you will have massive amounts of data especially for companies like Facebook that boast of a billion users currently. The system will need to be able to handle the request at very high rates of row-level updates over massive volumes of data and the messaging system must also be able to fetch the rows or messages by key typically by the user name or user id.

So, imagine filtering out a few messages from a billions of messages and displaying it to the user whenever he or she logs in. So, Facebook needed a system that could handle two types of data patterns. One set is where there is a short set of temporal data that needs to be volatile. For example, these are messages that are latest in your inbox. The other pattern is that an ever growing set of data that rarely gets accessed but still needs to be stored.

These are your old messages in your inbox that you would rarely access it again. But you might need it sometime when you never know when?

Now let us briefly look at the data store that they used to address the requirements of such a system.

One of the HBases primary competitors is Cassandra. Cassandra was created by Facebook and it was built specifically as an Inbox type application. Facebook choose HBase over Cassandra because they found that Cassandra's eventual consistency model was not a good match for their new real time messaging system which was a product of Facebook and also it has an extensive my sequel infrastructure but they found that their performance suffered as dataset and indexes grew larger.

These are a couple of reasons coupled with the fact that HBase is a scale out table. It's a scale out table basically supporting very high rates of row-level updates over massive amounts of data.

HBase is also a column based key value store built on Google's big table model. It's very good at fetching rows by key or scanning ranges of rows and filtering them down from a billion row table or even much larger table. a few are the reasons as to why Facebook choose HBase as a part of their technology stack was for the very reason that it has a simpler consistency model than Cassandra.

It has a very good scalability and performance for the data patterns and HBase at that time and even currently is the most feature rich for their requirements such as auto load balancing and failover, compression support.

HDFS which is the file system used by HBase supports replication, end-to-end checksums and automatic rebalancing. So it's reliable. The operation teams of Facebook have a lot of experience using the HDFS because Facebook is one of the biggest users of Hadoop and Hadoop uses HDFS as its distributed file system. However even these are the features that HBase provides over other competitors and it serves Facebook requirements. Complex queries are not reported in HBase. I repeat queries are generally given to analytics tools like hive which Facebook has created to make sense of their multi petabyte warehouse.

There are a lot of other used cases and how various other companies are using HBase, you can take a look at this link to give you a better inside as to how companies are using HBase.

To quickly summarized, we try to understand what is HBase at a very high level, when to use HBase and when not to use HBase and some of the use cases of HBase by giving an example of a few companies using HBase for their business requirements.