**The Problem**

The main aim for this project is to build an emulator that allows emulation of BASE properties backed over a traditional ACID Database. Acid transactions are the most commonly used form of database transactions in use today.

Acid transactions have four main properties. The first of these is that they should be atomic. This means that these transactions are ‘all or nothing’, in other words if one part of the transaction fails, then the whole transaction should fail, and the database should be left exactly as it was before the transaction started. The second key property of an ACID transaction is that of consistency. This is the requirement that all invariants and validation checks, such as primary and foreign key constraints, must still be valid at the end of the transaction. In other words, any ACID transaction must bring the database from one valid state to another, equally valid state. The third property of ACID transactions is that of isolation. This property is probably the simplest to understand, as it just means that any database transaction going through the system must have no impact on any other transaction also going through the system. The fourth database property that should be respected by ACID transactions is that of durability. Like isolation, durability is also fairly straightforward to understand. All it means is that once a transaction has happened, the end users should never be able to see the old state of the application unless they wanted to.

There are only three properties to take into consideration when it comes to BASE tractions. The first of these is basic availability. This property states that the system should never go down, even when one part of the system fails. Another property of most BASE systems is soft-state. This means that the consistency rules on BASE backed databases are more lax than their ACID counterparts. There is a catch however, and this is that a BASE database must be guaranteed to be consistent at some point in the future, which is the third BASE property of eventual consistency.

BASE transactions have proven extremely useful for the development of big cloud storage databases such as Amazon’s DynamoDB. This is because systems like Amazon might not need the latest data all of the time and so therefore consistency isn’t as important. The basic availability is also helpful as well, because every second something like Amazon is down, they are losing money.

The downside with BASE transactions is that because they are unpredictable, it makes it harder to perform good quality research experiments on them without spending a lot of money on a well known Cloud platform. This is where the emulator that I am going to be developing will come into play.