1. Runtime Schema :
   1. In traditional SQL you design your data model to represent your business objects.
   2. The problem with above approach is that it is very rare for end users to say with certainty what they want.
   3. In Cassandra, the data model can evolve over time and you are not tied to a hard schema, you have to update your data model around the queries you plan to run.
2. Write Performance:
   1. In SQL, If the tables are indexed, all writes require a lookup in the index, which in most cases is a O(log(n)) operation (n is the number of unique values in the index). Over time, when table grows, this is going to slow down writes.
   2. In Cassandra, all writes are O(1) and do not require a *read-before-write* pattern. You write your data, it gets stored into commit log and control is returned back to your application. Eventually it is available for reads, usually very quickly.
3. Read Performance:
   1. A traditional SQL database requires an index scan before it takes you to the right row.
   2. Cassandra primary key lookups are essentially O(1) operations and can work equally fast across extremely large data sets.
4. Operation Costs:
   1. There are some services out there that will manage a Cassandra cluster in the cloud for you OR use Amazon Dynamo DB (But dynamo and mongo are document db model, Cassandra is Column db model).