Problem Statement:

Explain in brief

● Hbase is a schema less database, what does it mean?

● what is the minimum number of column family every Hbase table should have?

● What is the benefit of using connection pool in Hbase?

Q1. Hbase is a schema less database, what does it mean?

What this really means is that the "schema" is stored with the record, not the table. In a RDBMS, the schema is defined and that table has the schema. In HBase (and other BigTable implementations) data is labeled with its types.

Q2. What is the minimum number of column family every Hbase table should have?

HBase currently does not do well with anything above two or three column families so keep the number of column families in your schema low. Currently, flushing and compactions are done on a per Region basis so if one column family is carrying the bulk of the data bringing on flushes, the adjacent families will also be flushed though the amount of data they carry is small. When many column families the flushing and compaction interaction can make for a bunch of needless i/o loading (To be addressed by changing flushing and compaction to work on a per column family basis). For more information on compactions.

Try to make do with one column family if you can in your schemas. Only introduce a second and third column family in the case where data access is usually column scoped; i.e. you query one column family or the other but usually not both at the one time.

Q3. What is the benefit of using connection pool in Hbase?

For applications which require high-end multithreaded access (e.g., web-servers or application servers that may serve many application threads in a single JVM), you can pre-create an HConnection, as shown in the following example:

Example 9.1. Pre-Creating a HConnection

// create a connection to the cluster.

HConnection connection = HConnectionManager.createConnection(Configuration);

HTableInterface table = connection.getTable("myTable");

// use table as needed, the table returned is lightweight

table.close();

// use the connection for other access to the cluster

connection.close();