**ASSIGNMENT-31.3:**

**QUESTION 1: Hbase is a schema less database. What does it mean?**

\* A schema-less database does not require conformation to a rigid schema (database, schema, data types, tables etc.) that one is required to live up to through the life of a system

\* In traditional RDBMS database the schema of the table is predefined and cannot and be modified easily. The schema is defined while making of the table that is the datatype of the column. But in case of the HBASE the schema is mentioned at time of loading the data.

\* Also the data stored in HBASE is using the bytes hence it is datatype independent. In this way even if new data comes into the database HBASE can handle it smoothly without any glitch. This is one of the pros of HBASE.

\* Does not enforce data type limitations on individual values pertaining to one single column type. Models the business usage and not a database schema, application or product.

**QUESTION 2: What is the minimum number of column family every Hbase table should have?**

The current HBase implementation does not handle a lot of column families well. Basically you should try to stick with one and add a second if you have radically different access patterns. There is a limit to the number of column families in HBase. There is one MemStore (It’s a write cache which stores new data before writing it into Hfiles) per Column Family, when one is full, they all flush. 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.

**QUESTION 3: What is the benefit of using connection pool in Hbase?**

* Database connections are often expensive to create because of the overhead of creating a new connection and initialization.
* Creating connections to a server component from an application is a heavy weight operation and it is much pronounced when connecting to a database server.
* That being the reason database connection pooling is used to reuse connection objects and HBase is no exception.
* In HBase, data from Meta table that stores details about region servers that can serve data for specific key ranges gets cached at the individual connection level that makes HBase connections much heavier.
* So if there are region movements for balancing or if a region server fails, the meta data need to be refreshed for each connection object which is a performance overhead.
* For these reasons, applications need to try to reuse connection objects created.