Assignment 9.1:

1. What is NoSQL data base?

NoSQL is an approach to [database](http://searchsqlserver.techtarget.com/definition/database) design that can accomodate a wide variety of data models, including key-value, document, columnar and graph formats. NoSQL, which stand for "not only [SQL](http://searchsqlserver.techtarget.com/definition/SQL)," is an alternative to traditional relational databases in which data is placed in tables and data [schema](http://searchsqlserver.techtarget.com/definition/schema) is carefully designed before the database is built. NoSQL databases are especially useful for working with large sets of distributed data.

1. How does data get stored in NoSQl database?

**Key-value data stores:** Key-value NoSQL databases emphasize simplicity and are very useful in accelerating an application to support high-speed read and write processing of non-transactional data. Stored values can be any type of binary object (text, video, JSON document, etc.) and are accessed via a key. The application has complete control over what is stored in the value, making this the most flexible NoSQL model. Data is partitioned and replicated across a cluster to get scalability and availability. For this reason, key value stores often do not support transactions. However, they are highly effective at scaling applications that deal with high-velocity, non-transactional data.

**Document stores:** Document databases typically store self-describing JSON, XML, and BSON documents. They are similar to key-value stores, but in this case, a value is a single document that stores all data related to a specific key. Popular fields in the document can be indexed to provide fast retrieval without knowing the key. Each document can have the same or a different structure.

**Wide-column stores:** Wide-column NoSQL databases store data in tables with rows and columns similar to RDBMS, but names and formats of columns can vary from row to row across the table. Wide-column databases group columns of related data together. A query can retrieve related data in a single operation because only the columns associated with the query are retrieved. In an RDBMS, the data would be in different rows stored in different places on disk, requiring multiple disk operations for retrieval.

**Graph stores:** A graph database uses graph structures to store, map, and query relationships. They provide index-free adjacency, so that adjacent elements are linked together without using an index.

1. What is a column family in HBase?

A **column family** is a NoSQL object that contains **columns** of related data. It is a tuple (pair) that consists of a key-value pair, where the key is mapped to a value that is a set of **columns**. In analogy with relational databases, a **column family** is as a "table", each key-value pair being a "row".

1. How many maximum number of columns can be added to HBase table?

There is no limit to number of columns in HBase, we can have more than 1 million columns but usually three column families are recommended not more than three.

1. Why columns are not defined at the time of table creation in HBase?

The column qualifiers (columns) do not have to be defined at schema definition time because each row in HBase can have a different set of columns so they can be added on the fly while the database is up and running.

1. How does data get managed in HBase?

The main characteristics that make HBase an excellent data management platform are fault tolerance, speed and usability.

**Fault tolerance** is provided by automatic fail-over, automatically shared and load balanced tables, strong consistency in row level operations and replication.

**Speed** is provided by almost real time lookups, in memory caching and server side processing.

**Usability** is provided by a flexible data model that allows many uses, a simple Java API and ability to export metrics.

* Data in HBase is organized into tables. Any characters that are legal in file paths are used to name tables.
* Tables are further organized into rows that store data. Each row is identified by a unique row key which does not belong to any data type but is stored as a byte array.
* Column families are further used to group data in rows. Column families define the physical structure of data so they are defined upfront and their modification is difficult. Each row in a table has same column families. Data in a column family is addressed using a column qualifier. It is not necessary to specify column qualifiers in advance and there is no consistency requirement between rows. No data types are specified for column qualifiers, as such they are just stored as byte arrays.
* A unique combination of row key, column family and column qualifier forms a cell. Data contained in a cell is referred to as cell value. There is no concept of data type when referring to cell values and they are stored as byte arrays. Versioning happens to cell values using a timestamp of when the cell was written.

1. What happens internally when new data gets inserted into HBase table?

When new data gets inserted, at the first write the data to the write-ahead log, the WAL:

* Edits are appended to the end of the WAL file that is stored on disk.
* The WAL is used to recover not-yet-persisted data in case a server crashes.

Once the data is written to the WAL, it is placed in the Mem Store. Then, the put request acknowledgement returns to the client. When we issue a Put command, the coordinates of the data are the row, the column, and the timestamp and when data is put into HBase, a timestamp is required. The timestamp can be generated automatically by the Region Server or can be supplied by us and must be a long integer. The timestamp must be unique per version of a given cell, because the timestamp identifies the version. To modify a previous version of a cell, for instance, we would issue a Put command with a different value for the data itself, but the same timestamp.