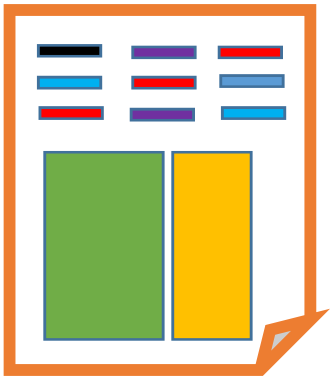
Assignment 9.1

1. What is NoSQL?

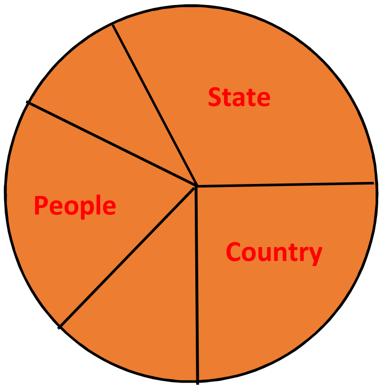
NoSQL database provides a mechanism for **storage** and **retrieval** of data that is modeled in means other than the tabular relations used in **Relational Database**. NoSQL databases are increasingly used in **Big Data** and **Real-Time Web Applications**. NoSQL systems are also sometimes called ”**Not Only SQL**” to emphasize that they may support SQL-like query languages.

NoSQL Database Types:

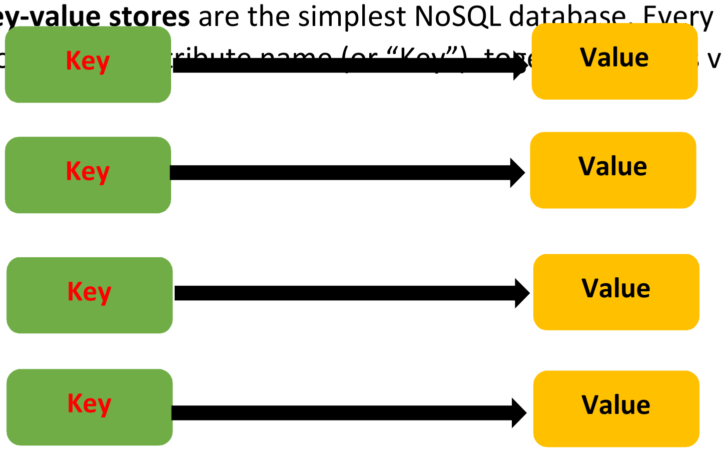
- **Document databases** pair each key with a complex data structure known as a document. Documents can contain many different key-value pairs, or key-array pairs, or even nested documents.



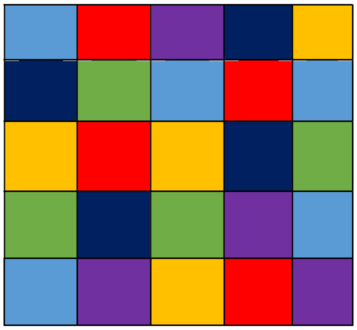
- **Graph stores** are used to store information about network of data. Graph stores include Neo4J and Giraph.



- **Key-value stores** are the simplest NoSQL database. Every single item in the database is sto[illegible]ribute name (or “Key”), toge[illegible] value.



- **Wide-Column stores** such as Cassandra and HBase are optimized for queries over large datasets, and store columns of data together, instead of rows.



2. How does data get stored in NoSQL database?

There are various NoSQL Databases. Each one uses a different method to store data. Some might use column store, some document, some graph etc.., Each database has its own unique characteristics.

Imagine you have an ordering system, storing customer details and product information in a MSSQL database and a NoSQL Event-Storing solution. **Since there is no schema limitations you can easily archive your events including all the relevant data as their properties and have them serialized for you automatically.**

Let’s assume we are storing an “Order Processed” event. Now, each time you create/store a new instance of this event, all the product and customer data will be serialized “as is” at the current point of time in your SQL database and stored right into your NoSQL database along with other events. A clear benefit from this would be the fact that if a customer decides to change his/her invoicing address or any other data for that matter, this event’s details would not change as it holds serialized data with no relation to the outside world or other entities. This fact makes it perfect for reporting purposes since the data is always accurate to single point in time when it was created!

3. What is a column family in HBASE?

Columns in HBase are grouped into ***column families***. All column members of a column family have the **same prefix**. For example, the columns “*courses:history”* and “*courses:math”* are both members of the “*courses”* column family.

The colon character (:) delimits the column family from the. The column family prefix must be composed of ***printable* characters**.

Column families must be declared up front at schema definition time

Physically, all column family members are stored together on the File System. Because tunings and storage specifications are done at the column family level, it is advised that all column family members have the same general access pattern and size characteristics.

4. How Many maximum number of columns can be added to HBase table?

Generally, column families remain fixed throughout the lifetime of an HBase table but new column families can be added by using administrative commands. The official recommendation for the number of column families per table is three or less.

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

HBase maintain the database properties by associating timestamp with each columns. And, we define column family and inside that we can have multiple columns, and each column can have different structure. That us how it maintains Dynamic Schema. If we initially define the column family while creating the table, the schema will not be dynamic.

6. How does data get managed in HBase?

- ❖ PARTITIONING
    - ➢ A table is horizontally partitioned into *Regions*, each region is composed of sequential range of keys

- Each region is managed by a *RegionServer*, a single RegionServer may hold multiple regions.

- PERSISTENCE AND DATA AVAILABILITY
  - HBase stores its data in HDFS, it doesn't replicate RegionServers and relies on HDFS replication for data availability.
  - Region data is cached in-memory
    - Updates and reads are served from in-memory cache (MemStore)
    - MemStore is flushed periodically to HDFS
    - Write Ahead Log (stored in HDFS) is used for durability of updates

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

To create data in an HBase table, the following commands and methods are used:

- **put** command,
- **add()** method of Put class, and
- **put()** method of HTable class.

Using **put** command, you can insert rows into a table. Its syntax is as follows:

Put '<table name>','row1','<columnfamily:column-name','<value>'

columns

Specifies the order and column names of those columns into which data is to be inserted. If no column names are specified, data is to be inserted into all columns that were listed, and in the order that was specified, when the named table was created.

VALUES (expressions)

Expressions supply values for every column. If a column list is specified, the expression list is evaluated to provide values for those columns. NULL is inserted for those columns that are omitted from the column list.

DISABLE WAL

Disables write-ahead logging for HBase writes and puts. Disabling write-ahead logging increases the performance of write operations, but it can result in data loss if the region servers fail. Your client application is responsible for ensuring data consistency when you use this option.

We use the **alter()** to update or make changes to the table