**Sequence file Input format**

It is a flat file consisting of key,value pairs in binary format.

It permits efficeient transfer to data among nodes

It is the reason why temporary mapper outputs are stored as sequence file format

Moreover we know that Hadoop is convenient to process small no of large files as large number of small files cause memory over head in name node , Sequence files are used as a container to store the small files. Sequence files are flat files containing key, value pairs. A very common use case when designing ingestion systems is to use Sequence files as containers and store any file related metadata(filename, path, creation time etc) as the key and the file contents as the value.

It provides block level and record level compression

**N line Input Format**

**It is used to read first N line from the format**

(ie)With NLineInputFormat each mapper receives fixed number of lines of input, unlike TextInputFormat and KeyValueTextInputFormat. The number of lines of input to each mapper can be controlled by setting the property, mapreduce.input.lineinputformat.linespermap .The default value is 1.  
  
Alternatively, we can  also set as  
conf.setInt(NLineInputFormat.LINES\_PER\_MAP, 1000); // sets N value to 1000   
  
NLineInputFormat is used in applications that take a small amount of input data and run an extensive (that is, CPU-intensive) computation for it, then emit their output.

**DB Input Format**

**It is used for getting input for Hadoop from a databases like mysql etc.**

DBInputFormat emits LongWritables containing the record number as key and DBWritables as value. The SQL query, and input class can be using one of the two setInput methods.

It can be set in Driver class by

Example

DBInputFormat.*setInput*(

job,

DBInputWritable.**class**,

"studentinfo",   //input table name

**null**,

**null**,

**new** String[] { "id", "name" }  // table columns

);

**DB Output Format**

**It is used for writing data from Hadoop into Database like my sql**

DBOutputFormat.*setOutput*(

job,

"output",    // output table name

**new** String[] { "name", "count" }   //table columns

);

System.*exit*(job.waitForCompletion(**true**) ? 0 : 1);

}