# Hive Data Definition

Hive data definition assigns relational structure to the files stored on the HDFS cluster. You can easily query the structured data to extract specific information.

# Example

Data definition for log files would contain columns like: CLASS, FILENAME, MESSAGE, LINENUBER, etc. Now if you want to check for the classes in which exception occurred, you can search for the term 'Exception' in the 'MESSAGE' column in a relational way. You can run SQL like queries for your files on cluster to search for the required data.

# Hive Data Manipulation

The ability to manipulate data is a critical capability in big data analysis. Manipulating data is the process of exchanging, moving, sorting, and transforming the data. This technique is used in many situations, such as cleaning data, searching patterns, creating trends, and so on. Hive offers various query statements, keywords, operators, and functions to carry out data manipulation.

They are

* Data exchange using LOAD, INSERT, IMPORT, and EXPORT
* Order and sort
* Operators and functions
* Transaction

# Example:SORT

Hive uses the columns in *SORT BY* to sort the rows before feeding the rows to a reducer. The sort order will be dependent on the column types. If the column is of numeric type, then the sort order is also in numeric order. If the column is of string type, then the sort order will be lexicographical order.

# HiveQL Manipulation

There are multiple ways to modify data in Hive:

* LOAD
* INSERT
* UPDATE
* DELETE
* MERGE

# Example :Loading files into tables

Hive does not do any transformation while loading data into tables. Load operations are currently pure copy/move operations that move datafiles into locations corresponding to Hive tables.

**Syntax**

|  |
| --- |
| LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename [PARTITION (partcol1=val1, partcol2=val2 ...)] |

**Synopsis**

Load operations are currently pure copy/move operations that move datafiles into locations corresponding to Hive tables.

* *filepath* can be:
  + a relative path, such as project/data1
  + an absolute path, such as /user/hive/project/data1
  + a full URI with scheme and (optionally) an authority, such as hdfs://namenode:9000/user/hive/project/data1
* The target being loaded to can be a table or a partition. If the table is partitioned, then one must specify a specific partition of the table by specifying values for all of the partitioning columns.
* *filepath* can refer to a file (in which case Hive will move the file into the table) or it can be a directory (in which case Hive will move all the files within that directory into the table). In either case, *filepath* addresses a set of files.
* If the keyword LOCAL is specified, then:
  + the load command will look for *filepath* in the local file system. If a relative path is specified, it will be interpreted relative to the user's current working directory. The user can specify a full URI for local files as well - for example: [file:///user/hive/project/data1](file:///\\user\hive\project\data1)
  + the load command will try to copy all the files addressed by *filepath* to the target filesystem. The target file system is inferred by looking at the location attribute of the table. The copied data files will then be moved to the table.
* If the keyword LOCAL is *not* specified, then Hive will either use the full URI of *filepath*, if one is specified, or will apply the following rules:
  + If scheme or authority are not specified, Hive will use the scheme and authority from the hadoop configuration variable fs.default.name that specifies the Namenode URI.
  + If the path is not absolute, then Hive will interpret it relative to /user/<username>
  + Hive will *move* the files addressed by *filepath* into the table (or partition)
* If the OVERWRITE keyword is used then the contents of the target table (or partition) will be deleted and replaced by the files referred to by *filepath*; otherwise the files referred by *filepath* will be added to the table.