

Relational Operations on MR and Hive

Relational Operations using MapReduce

1. Selection
 - a. Map : Read each row t of the table and check if it satisfies condition C , and if so, output (t,t)
 - b. Reduce : Do nothing
2. Projection
 - a. Map : Read each row t of the table and calculate subset of attributes t' and output (t,t')
 - b. Reduce : Eliminate duplicates
3. Union
 - a. Map : Reads in one file and send tuples as (t,t)
 - i. If multiple files exist, combine into one, but this combination is subjective
 - ii. Files may not have the same structure
 - iii. So each mapper reads in one input file and writes a key-value pair
 - b. Reducer : Merge the mapper outputs by eliminating duplicates
 - c. Schedule another map reduce job for selection
4. Intersection
 - a. Map : Read one file each and send tuples as (t,t)
 - b. Reducer : Consider only one duplicate per set of duplicates
5. Difference
 - a. Map : Read one file each and send tuples as (t,t)
 - b. Reducer : Consider non duplicates from mapper 1
6. Natural Join on attribute set B
 - a. Map : Each mapper reads $(B, (R,A))$
 - b. Reducer : Output (A,B,C)
7. Grouping and Aggregation
 - a. Map : For each line, output the attributes needed
 - b. Reduce : Aggregate each line

Hive

Hive is a system for querying and managing structured data built on top of MR and Hadoop. It has a SQL-like syntax for querying unstructured data as structured data, based on MR workflows.

1. Create -
 - a. CREATE TABLE table1 (count INT, bar STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LINES TERMINATED BY "\n" STORED AS TEXTFILE
2. Load -
 - a. LOAD DATA INPATH <hdfs file> INTO TABLE <table name>
 - b. Loads data without partitioning it
 - c. Keyword PARTITION can be used to partition this data
 - d. Makes access more efficient
 - e. Can specify LOCAL INPATH for local files
3. Select -
 - a. SELECT [ALL | DISTINCT] select_expr,... FROM <table> WHERE <cond> GROUP BY <cols> HAVING <cond> CLUSTER BY <cols> | SORT BY <cols> | DISTRIBUTE BY <cols> LIMIT <number>
 - b. Cluster by, sort by and distribute by all sort the data
 - c. Limit specifies the number of records to retrieve
4. Insert -
 - a. INSERT INTO TABLE <table> <select statement>
 - b. INSERT OVERWRITE TABLE <table> <select statement>

Hive uses a Hive QL Process Engine and an Execution engine running MR jobs based on the query, and uses a metastore to store metadata about the tables contained by the HDFS.

It's key components are

1. Hive CLI
2. Metastore
3. Compile - Translates statements into a DAG of MR jobs
4. Execution Engine
 - a. SerDe - Serialisation and Deserialisation is used.

The compiler is divided into the following components

1. Parser
 - a. Converts the query into a parse tree
2. Semantic Analyzer
 - a. Adds semantic information
 - b. Checks schema
 - c. Checks type of fields
3. Logical Plan Generator
 - a. Converts operators to logical operators
4. Optimizer
 - a. Optimizes the plan
5. Physical plan Generator
 - a. Converts plan to DAG of mapreduce jobs