UE Large Scale Processing @ ESIGELEC
2019/2020

09 – Hive

Abdel Dadouche
DJZ Consulting

adadouche@hotmail.com
@adadouche

What's Apache Hive?

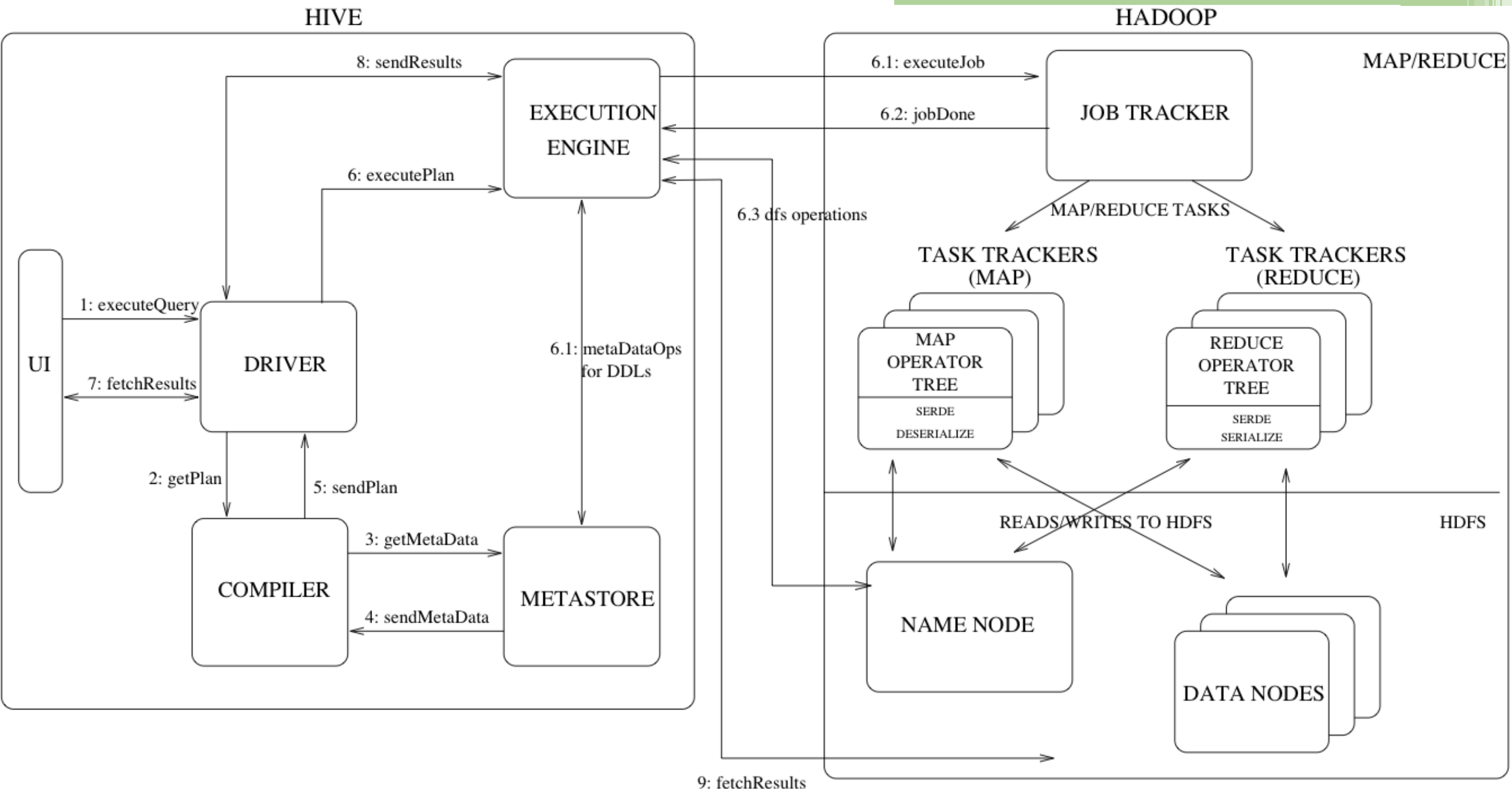
- It's a data warehouse software that facilitates reading, writing, and managing large datasets residing in distributed storage and queried using SQL syntax

- Provides the necessary abstraction to integrate SQL-like queries (HiveQL) without the need to implement programs using the low-level Java HDFS/MapReduce API for example

- Hive aids portability of SQL-based applications to Hadoop HDFS/MapReduce but supports other DFS like AWS S3

Why Hive?

- It provide a SQL-like abstraction with SQL types
- Designed for OLAP, and a good fit for ETL jobs as well
- Easier to use for business users
  - MapReduce is a low level API that requires skills to be able to achieve simple queries like a Join for example
- Can easily be plugged into traditional BI tools


HIVE
HADOOP
MAP/REDUCE
8: sendResults
6.1: executeJob
6.2: jobDone
EXECUTION ENGINE
JOB TRACKER
6: executePlan
6.3 dfs operations
MAP/REDUCE TASKS
TASK TRACKERS (MAP)
TASK TRACKERS (REDUCE)
UI
1: executeQuery
7: fetchResults
DRIVER
6.1: metaDataOps for DDLs
MAP OPERATOR TREE
SERDE DESERIALIZE
REDUCE OPERATOR TREE
SERDE SERIALIZE
2: getPlan
5: sendPlan
READS/WRITES TO HDFS
HDFS
COMPILER
3: getMetaData
4: sendMetaData
METASTORE
NAME NODE
DATA NODES
9: fetchResults

Hive Architecture

- Command-line interface (CLI) & UI:
  - Submits queries or instructions and monitor the process status
- Driver:
  - Acts as a controller which receives the HiveQL statements, create a sessions, and monitors the life cycle and progress of the execution.
  - Stores the necessary metadata generated during the execution of a HiveQL statement.
  - Collect data or query results obtained after the Reduce operation.
- Compiler:
  - Performs compilation of the HiveQL query & converts the query to an execution plan (abstract syntax tree (AST))
  - After checking for compatibility and compile time errors, it converts the AST to a directed acyclic graph (DAG)
  - The plan contains the tasks and steps needed to be performed by MapReduce to get the output as translated by the query

Hive Architecture

- Meta Store
  - Stores metadata for each tables such as the schema, location, partition metadata…
  - The data is stored in a traditional RDBMS format (derby)
  - The metadata helps the driver to keep track of the data
- Optimizer:
  - Performs various transformations on the execution plan to get an optimized DAG
  - Transformations can be aggregated together or split tasks etc
  - The transformation logic can be modified or pipelined using another optimizer
- Execution Engine:
  - After compilation and optimization, it interacts with Hadoop to schedule tasks to be run

Configuring Hive with Hadoop

- HDFS requirements
  - A /tmp & /user/hive/warehouse directory
- Initialize the Meta Store RDBMS
- Start the Hive server
- Start creating tables from scratch or load HDFS data files into your tables
- Write your first queries

Recap

- Hive has been key to demonstrate the value Hadoop
- Allowed non-dev users to access the data and fulfill a data warehousing/BI scenario
- Leverage the Hadoop HDFS/MapReduce paradigm
- Optimize the execution based on the SQL query (you still nee to pay attention to what you write)