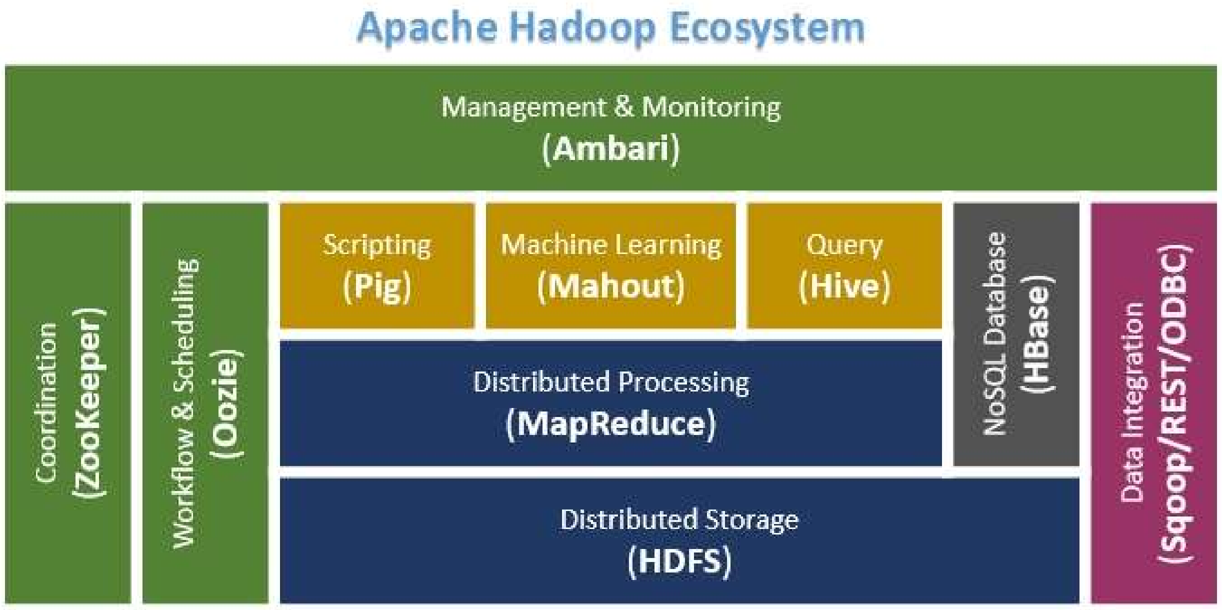
Big Data, Hadoop and Hive

Understanding Big Data and Hadoop

Big data is a term that describes the large volume of data – both structured and unstructured – that inundates a business on a day-to-day basis. Within the big data ecosystem Hadoop is an open source distributed processing framework that manages data processing and storage. Hadoop can handle various forms of structured and unstructured data, giving users more flexibility for collecting, processing and analyzing data than relational databases and data warehouses provide. Any type of data can be stored as is without putting constraints on how the data is processed.

Traditional RDBMS require the schema of the data store to be defined before the data can be loaded (*Schema-on-Write*) but Hadoop is *Schema-on-Read* meaning that raw, unprocessed data can be loaded into Hadoop with the structure imposed at processing time based on the requirements of the processing applications.


Apache Hadoop Ecosystem
Management & Monitoring (Ambari)
Coordination (ZooKeeper)
Workflow & Scheduling (Oozie)
Scripting (Pig)
Machine Learning (Mahout)
Query (Hive)
NoSQL Database (HBase)
Distributed Processing (MapReduce)
Distributed Storage (HDFS)
Data Integration (Sqoop/REST/ODBC)

HBase

HBase is called the Hadoop database because it is a NoSQL database that runs on top of the Hadoop Distributed File System (HDFS). HBase is a column-oriented key/ value data store which provides strong data consistency on reads and writes, which distinguishes it from other NoSQL databases. Data can be simply ingested into HDFS without a schema or pre-processing of data. Sqoop acts as the intermediate layer between the RDBMS and Hadoop and is a popular data ingestion tool. It is used to extract and move data between relational databases and Hadoop. Hadoop data lakes hold raw data across clusters from multiple sources. Data lake systems use extract, load and transform (ELT) methods for collecting and integrating data. Often the raw data is shared across the organization so directory placement and metadata management is critical for easy access.

HDFS and HBase Schema design

Hadoop's Schema-on-Read model does not impose any requirements when loading data into Hadoop but creating a structured and organized repository for the data makes it easier to use. The data model is highly dependent on the specific use case. Traditional datawarehouse implementations will likely use a traditional schema like star schema but other unstructured data will focus on directory structure and metadata management. Data modeling tools like Hackolade can be used for modeling, forward-engineering to create statements and reverse-engineering to generate tables. MapReduce is the framework where the actual data from the HDFS store gets processed efficiently. MapReduce breaks down a big data processing job into smaller tasks for processing.

Namespace

A namespace in HBase is a logical grouping of tables analogous to a database in relation database systems. A namespace can be created, removed or altered.

Table

Tables in HBase can serve as the input and output for MapReduce jobs run in Hadoop. Tables can also be used to store JSON. Tables are declared up front at schema definition time.

Row Keys

Row keys are uninterpreted bytes used to denote both the start and end of a tables' namespace.

Attributes data types

HBase supports a "bytes-in/bytes-out" interface, so anything that can be converted to an array of bytes can be stored as a value

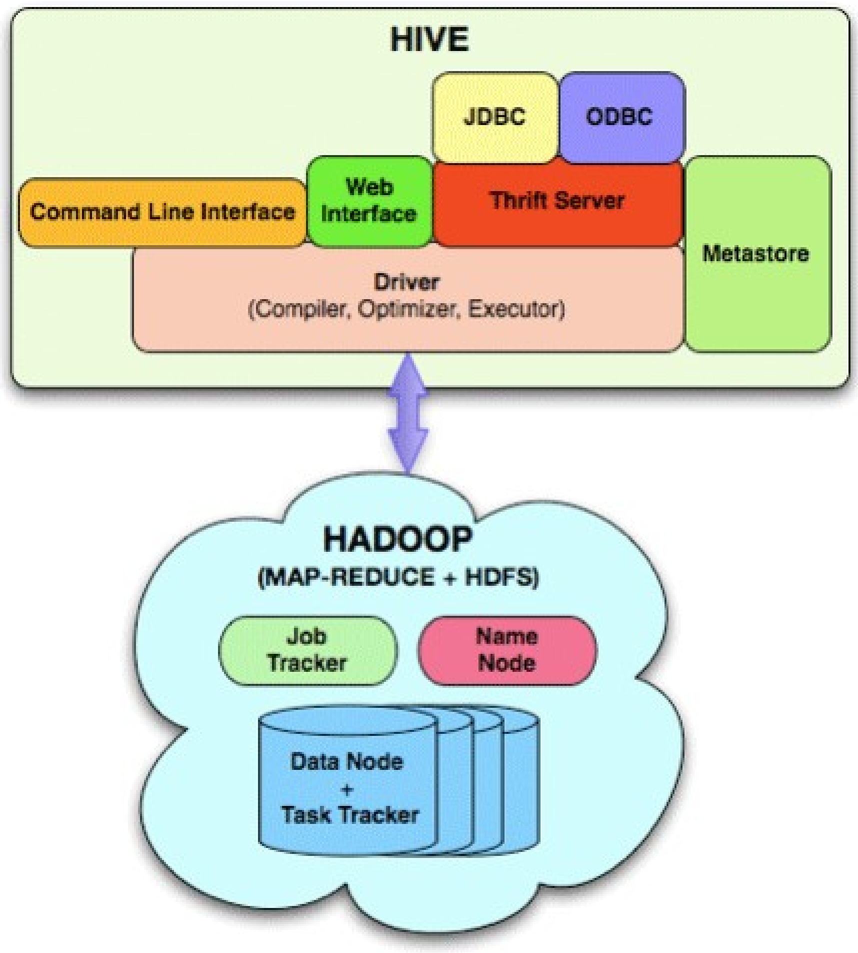
Data Migration between RDBMS and HDFS

Sqoop acts as the intermediate layer between the RDBMS and Hadoop to transfer data. It is used to import data from the relational database such as MySQL / Oracle to Hadoop Distributed File System (HDFS) and export data from the Hadoop file system to relational databases. Apache NiFi serves a similar purpose but provide but has enhanced functionality and provides a GUI. Other ETL tools like Kafka and Flume are also used for data flow and migration and can be used with sqoop to enhance functionality.

Hive

Hive is a data warehousing tool which provides a database query interface to Hadoop and facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in Hadoop compatible file systems. Hive structures data into well-understood database concepts such as tables, rows, columns and partitions. A SQL-like language called HiveQL (HQL) is used to query the data. Tools like Hackolade can be used for Data Modeling with Hive to handle Managed and External tables and their metadata, partitioning, primitive and complex datatypes, and the full HQL Create Table syntax.

Hive Architecture:

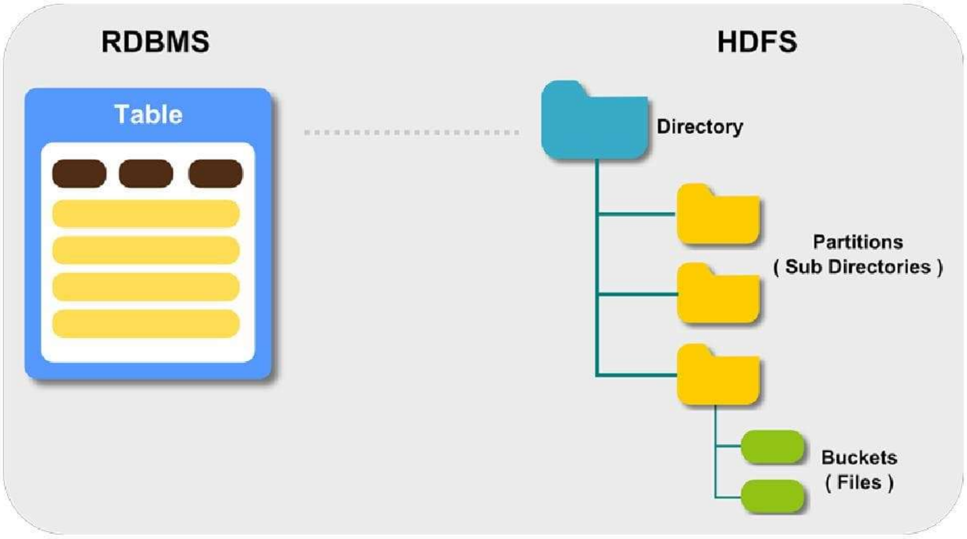


Hive data model

Metadata refers to data about the data (For example: Location of the data set, permission and ownership of files on HDFS, size etc.). Hive stores its metadata in a relational database called the *Hive metastore* which has a fixed schema and provides a tabular abstraction of the data sets. MySQL and Oracle are popular databases that are used as a Hive metastore. Traditional data modeling techniques are applied to the metastore.

Data in Hive can be categorized into three types on the granular level: Table, Partition, and Bucket. Hive organizes tables into partitions and then subdivides partition into buckets. Hive supports Schema on read, which means data is checked with the schema when any query is issued on it. HiveQL automatically translates SQL-like queries into MapReduce jobs.

Hive Data Model



Tables

The table in Hive is logically made up of the data being stored. And the associated metadata describes the layout of the data in the table. In Hadoop data typically resides in HDFS but Hive stores the metadata in a relational database and not in HDFS.

Partition keys, buckets

Hive organizes tables into partitions for grouping same type of data together based on a column or partition key. Each table in the Hive can have one or more partition keys to identify a particular partition. Tables or partitions are subdivided into buckets based on the hash function of a column in the table to give extra structure to the data that may be used for more efficient queries.

Data types

Hive supports different data types to be used in table columns. The data types supported by Hive can be broadly classified in Primitive and Complex data types.