What is Hadoop File System (HDFS)?

Hadoop File System (HDFS) is a distributed file system. Store all types of files in the Hadoop file system. It supports all standard formats such as Gifs, text, CSV, tsv, xls, etc. Beginners of Hadoop can opt for tab_delimiter (data separated by tabs) files because it is –

- Easy to debug and readable
- The default format of Apache Hive

What is the Data Serialization Storage format?

Storage formats are a way to define how to store information in the file. Most of the time, assume this information is from the extension of the data. Both structured and unstructured data can store on HADOOP-enabled systems. Common Hdfs file formats are –

- Plain text storage
- Sequence files
- RC files
- AVRO
- Parquet

Why Storage Formats?

- File format must be handy to serve complex data structures
- HDFS enabled applications to find relevant data in a particular location and write back data to another location.
- Dataset is large
- Having schemas
- Having storage constraints

A distributed file system that handles large data sets running on commodity hardware and used to scale a single Apache Hadoop cluster of nodes.

Source: *Apache Hadoop distributed file system*

Why choose different File Formats?

Proper selection of file format leads to –

- Faster read time
- Faster write time
- Splittable files (for partial data read)
- Schema evolution support (modifying dataset fields)
- Advance compression support
- Snappy compression leads to high speed and reasonable compression/decompression.
- File formats help to manage Diverse data.

Read more about **Data Ingestion: Pipeline, Architecture, Tools, Challenges**

Guide to Data Serialization in Hadoop

- Data serialization is a process that converts structure data manually back to the original form.
- Serialize to translate data structures into a stream of data. Transmit this stream of data over the network or store it in DB regardless of the system architecture.
- Isn’t storing information in binary form or stream of bytes is the right approach.
- Serialization does the same but isn’t dependent on architecture.

Consider CSV files contains a comma (,) in between data, so while Deserialization, wrong outputs may occur. Now, if metadata is stored in XML form, a self- architected form of data storage, data can easily deserialize.

Why Data Serialization for Storage Formats?

- To process records faster (Time-bound).
- When proper data formats need to maintain and transmit over data without schema support on another end.
- Now when in the future, data without structure or format needs to process, complex Errors may occur.
- Serialization offers data validation over transmission.

Areas of Serialization for Storage Formats

To maintain the proper format of a data serialization system must have the following four properties –

- **Compact –** helps in the best use of network bandwidth
- **Fast –** reduces the performance overhead
- **Extensible –** can match new requirements
- **Inter-operable –** not language-specific

Serialization in Hadoop has two areas –

Interprocess communication

When a client calls a function or subroutine from one pc to the pc in-network or server, that calling is a remote procedure call.

Persistent storage

It is better than java's inbuilt serialization as java serialization isn't compact. Serialization and Deserialization of data help maintain and manage corporate decisions for effective use of resources and data available in Data warehouse or any other database -writable – language specific to java.

Stacking of Storage Formats Hadoop

- Use splittable formats
- CSV files lack block compression; using CSV files in HADOOP may increase reading performance cost. Schema support is also within the limit.
- Use JSON records instead of JSON files. For better performance, experiments have to perform for each use case of JSON.
- AVRO file format and data serialization framework.
- Sequence files are complex in reading.
- Write operation is slower if RC (Row-Columnar) files are in use.
- Optimized RC (ORC) files are also the option to use but have less support.
- Parquet Files are using a columnar format to store data and process it.

Guide to Apache Hadoop File Format Configuration

- Hadoop distribution information and architecture type (system information where Hadoop system is working and process jobs) must analyze thoroughly.
- Perform Schema evolution by defining the structure of your metadata.
- Processing requirements for file formats and configure the HADOOP system.

- Engine support for reading and write operations.
- Evaluate storage size at a production level HADOOP File formats selection.
- Text files are lightweight, but splitting those files and reading data leads to huge maps (MAPS needs to create a complicated way to achieve splitting of text (files data).
- Sequence files support block compression. A hive has SQL types, so not worthy of working with Hive.
- RCFILE has a high compression rate, but it takes more time to load data.
- ORC can reduce data size up to 75% and suitable with hive but increases CPU overhead. Serialization in ORC depends on data type (either integer or string).
- AVRO provides features like serialization and Deserialization with file format also. The core of AVRO is its schema. Supports dynamic and static types.
- Parquet provides the facility to partition data into both rows and columns but computationally slow on the right side.
- Avro is a data exchange and data serialization service system. It is based on schemas.
- Always store Schema in an AVRO file with data.
- Support Specific and Generic data.
- JSON defines AVRO schema (most languages have JSON libraries), and data format is binary, which determines the efficiency and makes it compact.
- Set the default schema for some mismatched or failed schema.

Apache Avro Solution Offerings

At the same time, a few questions arise that –

Is it possible to handle any schema change – Yes, in AVRO, schema changes like missing fields, changed/modified fields, and added/new areas are easy to maintain

What will happen to past data when there is a change in schema – AVRO can read data with a new schema regardless of data generation time.

Is there any problem that occurs while trying to convert AVRO data to other formats, if needed – There are some inbuilt specifications in the Hadoop system while using a Hybrid system of data? In such a way, Apache spark can convert Avro files to parquet.

Is it possible to manage MapReduce jobs – MapReduce jobs can efficiently achieve by using Avro file processing in MapReduce itself.

What connection Handshake do – Connection handshake helps to exchange schemas between client and server during RPC.

How is AVRO different from other systems? (Thrift and protocol buffer)

- AVRO has Dynamic Schema stores for serialized values in binary format in a space-efficient way.
- JSON schema and data in the file.
- Parser No compilation directly by using the parser library.
- AVRO is a language-neutral system.

How to implement Avro?

- Have the schema of data format ready
- To read schema in program
- Compile using AVRO (by generating class)
- Direct read via Parser library
- Achieve serialization through serialization API (java)

By creating class – A schema fed to AVRO utility and then gets processes as a java file. Now write API methods to serialize data.

By parser library – Fed schema to AVRO utility and access serialization parser library to serialize data.

Achieve Deserialization through deserialization API (java)

By generating class – Deserialize the object and instantiate DataFileReader class

By parser library – Instantiate parser class

Creating schema in Avro

There are three ways to define and create JSON schemas of AVRO –

- JSON String
- The JSON object
- JSON Array

There are some data type that needs to mention in schema –

- Primitive (common data types)
- Complex (when collective data elements need to be process and store)

- Primitive data types contain null, boolean, int, long, bytes, float, string, and double. In contrast, complex contains Records (attribute encapsulation), Enums, Arrays, Maps (associativity), Unions (multiple types for one field), and Fixed (deals with the size of data).
- Data types help to maintain sort order.
- Logical types (a super form of complex data types) are used to store individual data involved, i.e., time, date, timestamp, and decimal.

Apache Avro Use Cases

- Schema evolution.
- Apture is using AVRO for logging matrices data.
- RichRelevance is using binary formatted key/value pairs.