2-1-2018

Avro: Overview and Implications for Metadata Processing

Graham S. Hukill
Wayne State University, ej2929@wayne.edu

Cole Hudson
Wayne State University, fi1806@wayne.edu

Recommended Citation
https://digitalcommons.wayne.edu/libsp/134

This Presentation is brought to you for free and open access by the Wayne State University Libraries at DigitalCommons@WayneState. It has been accepted for inclusion in Library Scholarly Publications by an authorized administrator of DigitalCommons@WayneState.
What is Avro?

Originally developed for Apache Hadoop, Avro is a system that provides standardized ways to package and send data in big data clusters. For our use, we found Avro to be a useful tool in a metadata processing engine we use that runs on Apache Hadoop and Apache Spark.

Some useful tidbits to know:

- Avro creates compact files (known as Avro data files) in a process known as serialization.
- Avro occupies a similar space as Binary JSON, Parquet, Thrift, and others.
- Avro serialization includes both the data as well as a schema—a JSON data structure that describes column names and data types.
- Avro is fast. In a big data environment with millions of records, Avro has to work quickly and create files that can be parsed easily by machine and human.
- Avro is robust. As schemas have changed over the spec’s evolution, old versions of Avro files can still be read and used.

Using Avro opens the door to some interesting metadata processing implications. Have you ever had 500k records take down your computer as you processed them? Avro and Spark may be a way to perform powerful analysis using underpowered machines. Here is why:

- Spark can efficiently handle files (like Avro) and split them into multiple segments that are reconstituted as needed.
- Avro files, because of their efficient serialization and integration into the Spark environment, can easily be parsed and queried in Spark.

This means you could take 10GB of records with a system that could handle at most 2GB at a time, and, with a SELECT query (see far chart), you could find all records with a certain field. If you were to write this type of query manually, you would need to do a fair amount of programming on your own: loop through records, keep track of all instances of the field you want, and, oh yeah, don’t run out of memory or take an untenable amount of time.

A portable database

Avro files alone do not make a “portable database,” but they get you closer! A SQL dump may contain the same schema support that Avro does, but require a spinning database instance to interact with. Other formats might offer compressible, binary serializations, but might also be difficult to reconstitute and inspect.

Avro strikes a balance between:

- efficient storage and transmission
- defined data types
- ease of analysis through readily available tools...

...without requiring extensive infrastructure. In the same way that MySQL allows querying over a million rows, so too can the combination of Avro and environments like Spark, all while efficiently utilizing computer resources if available, or working within the constraints of what’s at hand.