Tricia Camaya

June 28, 2022

Voltron Conference

The Voltron Conference was held on June 23, 2022. The conference focuses on the latest innovation happening around the Apache Arrow ecosystem. The conference was first introduced by Marlene, developer advocate and python programmer, and she talked about the main objective of the conference and the name of the speakers. She also mentioned that there would be two different segments which are a series of live sessions and a range of pre-recorded talks. The keynote speakers started with Wes Mckinney and Jacques Nadeau who are the co-creators of Apache Arrow. They talk briefly about the different innovations that are happening across the ecosystem. There was a panel session on high performance computing and getting to see how an arrow is being used in that space. Python community speakers were the CEO of Anaconda Peter Wang and the CEO of Voltron Data Joshua Patterson. Jacques Nudo is the CEO and Co-founder of Sundik which is a company focusing on enhancing existing cloud data. He is joined by Wis Mckinney who is the CEO and co-founder of voltron data and a software engineer and entrepreneur focusing on analytical computing. He created the very famous pandas library. Both speakers briefly talk about how they started and their future plans. Some key takeaways that I got from this talk are large data sets in python dealing with memory issues running out of memory having to do with alot of development burden building custom implementations of query processing analytics data serialization. The basis of collaboration is key to solving critical problems.

The early days of arrow started with the concept of the deconstructed database. It is a subsystem of an analytic database system which is a memory format that develops algorithms. The critical subsystems that are necessary to build a distributed computing engine which was an aspiration to build all of these things to support connectivity. Better database connectivity, distributed system connectivity, and creating compute primitives was the earlier goal. Embeddable execution engines expression compilation building have all of these things. Hardening the memory format first was highly dependent on the community shift to these connectivity and in-memory computing initiatives. The early challenges were changing the format a bunch of times. The production number of production customers was in place so changing the memory format of the engine multiple times was challenging. The outcome of that was the 1.0 release. The in-memory format itself has not changed at all, which was a testament to the amount of work that went into getting that right. Building the right foundation to start makes it a lot easier for these other things to be vibrant right. Very challenging thing in the community is that people started to build a lot of these other kinds of components on top of the memory specification when there wasn't a really stable foundation in that specification. It would have been very disruptive and frustrating. Many people start building something and then all of a sudden specification would change and they end up rewriting the algorithm.

It was time to focus on the core to make it successful. Apache software foundation has a mechanism for open source projects to merge together which is like an open source acquisition. The significant increase in the diversity of technologies represented in the project focused on C++, JAVA, and Python. These community emerges were able to expand the number of programming languages as well as expand the types of software that was being developed in the project. Development of the R package and ecosystem for arrow was a fast growing rust ecosystem. Aero software components provide a foundation for the next generation of query processing systems. Multiple query engine projects happening inside aero. Meta has a developing columnar query which is a database acceleration library called Velox. A Kafka competitor astro streaming is powered by apache pulsar which is a coffee competitor and allows machine learning. Lineage information is used to better understand how data is moving through different transformations and systems. Ibis is a pythonic data analysis api that's data engine agnostic or engine backend agnostic. Scientists are able to write their analytics code once and then seamlessly migrate their data from one supported engine backend to another accelerating the portability of their data analytics stack the value of tools like ibis is immense where no longer do teams have to manage or engage in massive rewrites right to execute on a new engine back end. They can merely update the connection information and the rest of their code will submit their query for execution. Today ibis supports 10 different backends and is currently working to support even more including the arrow C++ engine Acero so users of popular data frame libraries like pandas are able to do the same work with ibis. The Data Thread brings together the Apache Arrow community and industry solutions built on top of Arrow. The Data Thread brings together the community.