EPISODE 610

[INTRODUCTION]

[0:00:00.3] JM: 20 years ago, all of the data in an organization could fit inside of relational databases. Imagine a company like Procter & Gamble. Procter & Gamble is a consumer packaged goods company with hundreds of business sectors; shaving products, toothpaste, shampoo, laundry detergent. 20 years ago, if the chief financial officer of Procter & Gamble wanted to answer a question about the revenue projections within the enterprise, that CFO would ask a VP to find the answer. The VP would contact the business analysts in all of the different departments within Procter & Gamble, and those business analysts would work with database administrators to answer questions for their business sector. You'd get the report for the shaving products, the report for the toothpaste, the report from the shampoo people. You could aggregate all those reports together. In that world, it might've taken weeks or months for the CFO to get the answer about revenue projections.

Today, data engineering has improved dramatically. Data sets within an enterprise are updated more rapidly. The tooling has advanced, thanks to the Hadoop project, leading to a wide range of open source projects that feed into one another. But data problems across an enterprise still exist. Business analysts, data scientists and data engineers struggle to communicate with each other. The CFO still can't get a question about revenue projections answered instantly. Instead of instant answers, we live in a world of friction, and batch processing, and monthly reports, and this is actually not true just of old enterprises like Procter & Gamble or Coca-Cola. It's true of newer startups, like Uber, Airbnb and Netflix. It seems that no amount of engineers and financial windfall can completely cure the frictions of the modern data platform.

Tomer Shiran started Dremio to address the long-lived problems of data management, data access and data governance within an enterprise. Dremio connect databases, storage systems and business intelligence tools together and uses intelligent caching to make commonly used queries within an organization more readily accessible. Dremio is an ambitious project that spent several years in stealth before launching.

In today's episode, Tomer gives a history of data engineering and provides his perspective on how the data problems within an organization can be diminished. Full disclosure, Dremio is a sponsor of Software Engineering Daily.

[SPONSOR MESSAGE]

[0:02:45.3] JM: Over the last two years, I spent much of my time building a video product. We had issues with latency, unreliable video playback, codecs. I was amazed that it was so difficult to work with videos. As it turns out, video is complex and figuring out how to optimize the delivery of video is not easy, especially since there is both mobile and desktop, and mobile users might not have as much bandwidth as desktop users. If you're an engineer working on a product that involves video, you just don't want to think about any of this. I can tell you that from first-hand experience, and that's why Mux exists. Check out mux.com/sedaily and find out how Mux makes it easy to upload and playback video.

Mux makes video hosting and streaming simple. Even if you aren't working on video right now but you think you might work with video in the future, Mux is a really useful tool to be aware of. When I found out about Mux, I just started thinking of interesting ideas that you could build as a platform with video, because video is so hard to work with right now.

Check out mux.com/sedaily, and if you're an engineer who is looking for work, you can also apply for a job at mux.com/sedaily. I have met the Mux guys. I've interviewed them a couple times. You can listen back to those episodes. They have great offices, by the way.

On Software Engineering Daily, we've done to shows with Mux and I know that they're solving some big difficult problems involving lots of data and large video files. To find out more go to mux.com/sedaily. Thanks to Mux for being a sponsor of Software Engineering Daily.

[INTERVIEW]

[0:04:44.8] JM: Tomer Shiran, you are the CEO at Dremio. Welcome back to Software Engineering Daily.

[0:04:49.3] TS: Thanks for having me.

[0:04:50.9] JM: I want to go through a brief history of data engineering with you, because I think you know as much about it as many other people. Well, not many other people actually. You have probably a more authoritative understanding of data engineering than most people. 20 years ago, we didn't really have many tools for data engineering. Things were more standardized in terms of the people who were using databases or the other BI tools that might've been available 20 years ago, and we didn't have that term data engineering. So if we're talking about companies in the late 90s, what were the closest analogs to data engineering back then?

[0:05:34.4] TS: Yeah. I would say that we've had databases now for a while, but the need to move data around and the need to tune those databases and make them work, that responsibility fell – Come on the shoulders of database administrators, so the DBAs as well as ETL engineers. So we started to see products like Informatica that would allow people to, in a more kind of off-the-shelf way, move and transform data. Those are basically ETL engineers. And a lot of these things were done through scripting and things like that as well.

[0:06:08.7] JM: The closest thing to data engineering was probably data analysis. You had a relational database. You had a DBA that administered it, the database administrator. If I'm the CFO of a large enterprise company back in those days, like 20 years ago. Let's say Coca-Cola. I'm the CFO. I decide I want some aggregation of data and it's all in a database. How did that query get carried out?

[0:06:37.6] TS: Yeah. So in that timeframe, basically, you'd have the data in one database or a data warehouse. As a CFO, you're a very non-technical person typically and your expectation is that somebody is creating a report for you. So you're actually not interacting directly with the data. So you'd typically go to somebody in IT and maybe submit some requirements, explain to them what it is that you want the report or the dashboard to look like and they'd go do the engineering works to make that happen. They'd connect a business intelligence tool to that database and then produce that report. So it'd be a very simple kind of on the backend because typically at the time data wasn't that big. It was all in one place. With a combination of one BI tool and one database, you pretty much could solve that problem, although you had to be technical because of the nature of the tools at the time.

[0:07:33.0] JM: As we move along the timeline, 10 years ago, Hadoop was born and in its infancy, and if we fast-forward from that point in time when the CFO of Coca-Cola asks for a report and its pretty straightforward process. Towards 10 years later, where Hadoop comes out, how had the data engineering world advanced from the late 90s to the Hadoop timeframe? Had time just stood still between that relational database era and the point at which Hadoop started to make data engineering more of a widespread challenge?

[0:08:18.3] TS: Well, I'd say even before kind of the rise of Hadoop and the data lake, we had data warehouse, rights? We had companies, technologies like at Teradata and, of course, Oracle and Microsoft, etc., where companies that had data in lots of different places would try to consolidate that data into a single data warehouse. Those were very complex projects because a lot of upfront kind of data modeling and preparation of the data had to take place and you had to kind of figure out what exactly people were going to ask and people would build OLAP cubes and so forth. So that's one thing that was going on even before Hadoop is kind of the rise of the data warehouse as a centralized relational repository of data. A lot of effort went into creating those things and keeping them up-to-date, many millions of dollars typically for a typical enterprise.

But in parallel to that also, we have the rise of much nicer tools from a BI standpoint. So, for example, technologies like Tableau where you do need to be as technical to actually create visualizations, right? We started talking to Agile BI and self-service BI and making that easier. Then Hadoop came along and originally developed at Google and later kind of as an open-source project at Yahoo then commercialized by various different Hadoop vendors. The goal there was to really solve some of the problems that existed in the world of data warehousing where it was just too hard to get data into those platforms, right? With the data warehouse, you had to kind of figure out schemas and models and spend tons of time before you could even ingest a single row of data into the system. Whereas the Hadoop environments came along and made it easier to get data into a platform and kind of centralized it in one place. At least that was the hypothesis.

[0:10:04.4] JM: Hadoop got stable around 2011. You were at MapR around that time in the early days. MapR, I think, 2009 through 2015. In that early point in time of Hadoop's life, 2009, 2010, 2011, when you're working with those enterprises that were starting to adopt Hadoop, what did their infrastructure look like?

[0:10:30.3] TS: Yeah, I actually remember my first meeting with LinkedIn about Hadoop back in 2009. I think they had a 20 node Hadoop cluster in the basement. They also had a 20 node, it was called Aster Data. It doesn't exist anymore, but kind of an MPP database as well. Fast-forward a few years later, they were running thousands of nodes of that system of Hadoop. But companies, they had data in lots of different places, data that was in relational databases, data that was in non-relational databases, like MongoDB and Elasticsearch and so forth, and they started to stand up these Hadoop clusters running on potentially hundreds of servers, sometimes even more, and creating that kind of single repository where they could load data into the system.

Yeah, I was at MapR from 2009, 2015. One of the first employees there, and it is a great time as we were kind of enabling all these enterprises to, for the first time, be able to kind of bring together data from different sources and into one place and be able to do something about that data.

[0:11:31.4] JM: How did the Hadoop adoption change those enterprises? If I'm Coca-Cola back in 2009, 2010, 2011, I decide I want the big data. How does that change how I look at my infrastructure?

[0:11:45.3] TS: It went from – At least the goal was to go from a world of lots of silos of data and lots of work to kind of bring that data into one place into having kind of a centralized data lake where you can throw in lots of data running on commodity servers. You didn't have to run on high-end specialized hardware either or kind of MPP appliances. You could just run on commodity hardware that you would buy from companies like Dell and HP and Cisco and so forth and it was a low very low cost way of storing large volumes of data. That was kind of the significant innovation that came from that technology.

[0:12:27.0] JM: I think to some extent, the silos of data access patterns and data technologies may be got simplified, but there are still silos, and the silos today maybe you could describe them through the lens of the different roles of people who are interfacing with the data and the data infrastructure in different ways. You've got the data scientist. You've got the business analyst. You’ve got the data engineer. Describe the different roles, the different silos that might exist in a data-focused enterprise today.

[0:13:05.3] TS: I think from a personnel standpoint, you have different roles. You have kind of the data engineers who are responsible for kind of processing the data, cleaning it up, making it available and so forth. You have the data scientists who are kind of the more technical consumers of data. So when we talk about data consumers we think of kind of two broad classes of users. One being the business analysts, and the other being the data scientist. So a business analyst is typically using BI tools and kind of drag-and-drop tools like Tableau and Looker, Power BI from Microsoft and Click and MicroStrategy and so forth. Then you have the data scientists who are often using more advanced tools, and that could be things in the Python ecosystem, things like R and so forth, and that goes all the way up to kind of more machine learning and AI, but everything kind of on that spectrum between kind of handwriting complex SQL queries to doing more advanced things.

[0:14:07.4] JM: Today, the Coca-Cola CFO asks a business analyst, may be a data scientist, a question, something relating to revenue probably. How long does that question take to answer and what are the different points along the data access and data aggregation question into the result that gets received by the Coca-Cola CFO?

[0:14:36.4] TS: Yeah. I'd say, unfortunately, the theory of having all the data in one place remain very much something theoretical that we as an industry we’re not able to accomplish, right? Even with kind of the rise of the data lake, whether it's the on-prem data lake with Hadoop or now things like S3 or Azure Data Lake Store, it still very much a complex data infrastructure landscape. So for most companies, they might have that raw data sitting in a data lake, but then in order to get the performance that they want they end up having to extract some of that into a data warehouse or an MPP database and then that's not fast enough so they end up that creating aggregation, pre-aggregating the data in aggregation tables or maybe the they end up creating cubes so they can support faster analysis. Then the golden – Just the raw data as is is not as suitable for kind of a data consumer for an analyst. So somebody has to go and preprocess that data and so forth. You end up having so many different copies of data and so much work that goes on for everything that you want to do.

So when that business analyst to serving the CFO of the company wants to do something, unless it's something that they've already done before, chances are they have to go to somebody in IT, like a data engineer, file a ticket, wait for their ticket to get prioritized. The data engineer then has to go bring together the data, maybe run the queries for that and basically do a bunch of work just for that one simple kind of request that came down from the CFO or maybe from just the business analyst or the product manager. So it's a very complex process, very long. Often takes weeks and sometimes more for most companies.

[0:16:25.1] JM: All those sources of delay that can occur in that significant timeline between the Coca-Cola CFO having a question and that question being answered, you saw these when you were at MapR. You saw these and they were part of the impetus for starting Dremio. When you left MapR to start Dremio, what were the specific problems in data engineering that you thought you might be able to solve?

[0:16:54.1] TS: Yeah. We basically looked at the life of a data engineer and what we saw was that they were not happy. They were overloaded and they were constantly dealing with very tactical reactive work. So was these types of requests from data consumers and others that we're asking them, "Hey, can you run this query for me? Can you do this for me? Can you get this for me?" and that meant that they were constantly busy kind of doing these support tickets rather than doing more strategic work. Then at a higher level, the companies were just unable to find enough data engineering talent as well.

So, I recently had dinner with the heads of data from some of the largest kind of unicorn type companies in the Bay Area and what they were saying is that today, because of the complexity of the kind of infrastructure and the volume of data, you kind of need a data engineer for every analyst that you have to be successful and nobody can even get close to that kind of a ratio in terms of finding the talent. So that made us realize that the solution had to be in technology. There had to be a better way of doing things than having this very IT kind of engineering-driven involvement for every single thing that people want to do with data.

[0:18:03.1] JM: The unicorn companies, those are not as old as Coca-Cola. So these data problems, these don't just exist at legacy. But I don't want to call Coca-Cola a legacy, but it's an older company. They don't just exist at older enterprise. They also exist at a company, like – I don't know. I'm going to throw a name out there. I've no idea if this company was at the meeting, but like an Airbnb, where it's like a big company but still kind of startup-y.

[0:18:31.0] TS: Right, and I think that that's what makes this even more significant, right? So for these companies, maybe they can get to a data engineer for every two consumers of data, but when you get into the more traditional enterprise, those ratios can be 10 to 1 or 100 to 1 and not only that, but their infrastructure, there are so much legacy from just years of operating and acquisitions and things like that. That's make the problem even worse.

So in some cases, people want to ask a question about data and it takes months in order for them to be successful doing that and it takes a lot of engineering involvement for any kind of new question, and that's a big inhibitor for these companies who all realize that they have to be data-driven, right? They're being disrupted by the likes of Google and Amazon and so forth, but they can be data-driven because people can access data and they can't take advantage of data. In most cases, when something takes you a week or month, you give up. You just don't do it, you kind of move on. So that's a big problem and that's something that we're trying to solve here.

[SPONSOR MESSAGE]

[0:19:39.8] JM: This episode of Software Engineering Daily is sponsored by Datadog. With automated monitoring, distributed tracing and logging, Datadog provides deep end-to-end visibility into the health and performance of modern applications. Build rich dashboards, set alerts to identify anomalies and collaborate with your team to troubleshoot and fix issues fast.

Try it yourself by starting a free 14-day trial today. Listeners of this podcast will also receive a free Datadog t-shirt at softwareengineeringdaily.com/datadog. That softwareengineering daily.com/datadog.

[INTERVIEW CONTINUED]

[0:20:26.8] JM: You could solve this problem in different ways at different levels of the stack. So Looker, for example, I remember talking to somebody about Looker. They were asserting that Looker approaches the same problem of the different roles and the long lead time from question to answer, but it solves, I think, more at the BI level. I think of Dremio as, I guess, beneath the BI level or perhaps also encompassing the BI level but short of taking a more full stack approach. Where would you describe Dremio is sitting in the stack of different tools that are – Because like the business analyst is working with the BI tool. The data engineer is working with the data infrastructure. The data scientist is interfacing – I don't know, more in the middle or along the entire surface of everything. Where in the stack is Dremio encompassing?

[0:21:29.1] TS: Yeah, it's a good question. So we actually don't get involved in the visualization layer. We leave that to companies like Looker and Tableau and Microsoft Power BI which do a great job there, and those companies have actually focused on making the visualization layer self-service. You don't need kind of somebody technical to create the visualization or do the report for you. The problem that we're solving is that everything underneath that layer, which today encompasses a lot of ETL and kind of manual data engineering and cobbling together lots of different solutions. That layer is very much not self-service, and so we're at very focused on making the rest of the data stack self-service just like companies like Tableau and Looker and Microsoft Power BI make the visualization layer self-service.

[0:22:13.9] JM: Okay. The interaction between data scientists and data engineers and business analysts, what should that interaction be like?

[0:22:25.9] TS: That's a great question. It's exactly the kind of things that we focus on, right? Because we've built kind of an open source platform that facilitates that interaction. So it's not this file a ticket, wait three weeks to get something. We believe that data engineering should be responsible for doing that kind of initial, maybe collection and processing of data into some kind of a state where it's broadly useful to general audience of data consumers within the company. But then challenge is every data consumer, every business analyst or data scientist wants things a little bit different. It could be as simple as the columns named differently or they want some dataset in the organization joined with their own custom spreadsheet. So all those kinds of things, we call that the last mile similar to kind of last mile in logistics or in the telco world where that's the hardest problem. We want to solve the last mile problem and we want to make it so that the data engineers don't have to get involved in the last mile, because that's so specialize and customized to each individual user that it just bugs them down if they have to serve each of those users.

So we think that data engineers, they should worry about kind of the long haul, the more kind of standardized processing of the data and infrastructure in the company. We want to provide technology that makes the data consumer much more self-sufficient so they're not constantly bothering the data engineer with individual kind of tasks.

[0:23:56.4] JM: And give a little bit more description for the frictions that exist within these specific roles and between the specific roles, like problems that you need to be able to solve.

[0:24:09.4] TS: Yeah. Let’s look at a few examples. One simple example is, let’s say I’m a business analyst or a product manager. I want to do something with data, and maybe that data is in two different places today, two different systems. So, today, without Dremio the solution is I go to – I file ticket within kind of the support portal that I have in the company, and at some point that gets prioritized and somebody from data engineering can start a project to integrate those two sources, maybe load them into a centralized kind of data warehouse or an S3 bucket or something like that. So that's one example of something that we'd like to make it so that the data engineer doesn't have to get involved.

Another example is maybe, as a business analyst, I'm doing some analysis in Tableau or Looker or something like that or maybe using Python. The cores are just too slow. I'm not getting the performance that I want. So, again, today, without Dremio, would become a data engineering task. Somebody would have to pre-aggregate the data, maybe sessionize it, maybe aggregate it by city or something like that so that I can get a faster response time or maybe they’d have to load it into in-memory database like HANA. So there's a lot of work that would have to happen so that I could get the performance that I need in order to be able to interact with the data.

Again, that becomes a multi-week project to move the data into a faster source or to process the data in a specific way that would make quarries go faster. But then chances are I’m then going to want to do something different without data. So that processing that's taking place is not no longer helping me get the performance so something needs to be adjusted or a cube has to get rebuilt. So it's just so much back and forth every time I'm not getting the performance that I want or I don't have access to the data that I need.

[0:25:52.4] JM: When you started working on Dremio, how did you think about addressing those frictions more specifically? At editing engineering level, what did you think that you could build to be able to address those frictions?

[0:26:08.4] TS: Yes. So we envisioned a platform. We now call it a data as a service platform because we see that companies across every vertical, across every industry want to deliver data as a service internally within their organizations. That platform that we envisioned would be something that could connect to any data source that the company has. So we started kind of focusing on that the data lakes that people had already built as well as some of the kind of relational and non-relational databases. Then something that would connect would allow them to continue to use the existing BI tools and data science applications that they have, things like Pandas and Tableau and Power Bi and so forth.

Really, at the core of the system is this idea that the only way to solve this problem is to kind of create this abstraction layer which allows the consumers of data within a company to be able to interact with data and explore data and analyze data through kind of an abstraction layer so that they can do data prep, they can join things, but I'll do all of that without creating copies of data. So do it all in kind of a virtualized way.

Then the system would provide kind of the execution and query acceleration and caching capabilities that were needed to make things go fast irrespective of what was done at the abstract level. So that's really what we built. If you think about it from the user interface standpoint, it looks a lot like Google Docs. Except instead of Docs, it's datasets. So users can create new data sets. We call them virtual datasets. They can then share them with their colleagues who can build on top of that, and the company has the ability to see kind of that data lineage and what's been built and what's dependent on what and all of that is basically at zero cost because it's all virtual datasets. They're not creating copies of data.

[0:27:54.9] JM: Right. Last time we spoke about those virtual datasets, and virtual datasets are the datasets that I want Dremio to be aware of across my organization, whether it's MySQL or Elasticsearch or HDFS. Then you have inflections, and reflections are the joins or the other expensive queries. You could join MySQL data from MySQL with Elasticsearch, for example, and then you could turn that into a reflection as a materialized view and have that expensive query be run and be stored as that materialized view and then you could access it when you wanted to. You have this distinction between the virtual dataset, which doesn't cost you anything, and the reflection that is a cached, a materialized view. Is that right?

[0:28:53.4] TS: Tat's correct. The virtual datasets are – That's that abstraction layer. That's what allows people to go from – By default, you can connect Dremio in your environment. You can get started. You can do joins between the different physical datasets that you're connected to. The directory of CSV files that you have or parquet files in S3 with your Oracle table. But then the virtual datasets are the way in which the users can create new datasets that may be have some kind of data prep, data curation done on them. Maybe there are a subset of the data. Maybe they're filtered. Maybe they're aggregated. Maybe they're joined between two sources. But then as the Bi user, for example, is querying these virtual datasets, we want to make sure these queries go fast. So the way you make something go fast is by maintaining various data structures that make that go fast. Kind of in the same way that you think of when you search the web through Google, you're getting a very fast response time and that's because Google is not actually going in scanning all the webpages on the internet when you run that search query. It's because they've built indexes and various models that they use to support answering that query in a much faster time. Database indexes, if you look at Oracle indexes or cubes in kind of the OLAP world, the ideas the same, right? You have data structures that make it much faster and easier to answer queries.

The idea with the way that we do these data reflections, which is kind of one of the key innovations here, is that the data consumer, the person who's running the queries, is not even aware of their existence. So those are completely behind the scenes. They get managed and maintained by the system and their queries just magically go faster than they would otherwise.

[0:30:31.1] JM: The notion of the virtual dataset, if that's the data that I already have across my organization, like a virtual dataset would be an Elasticsearch index or a MySQL database. Why is it useful to be able to have that as an abstraction in Dremio? If the virtual dataset already exists, why is it useful to have it as something within Dremio?

[0:31:01.0] TS: Yeah. When you connect actually – Let me just clarify. When you connect Dremio to an Elasticsearch cluster and let's say your Azure Data Lakes Store or your S3 bucket, we actually call – The things in those systems, we call them physical datasets and we actually never change them. But what happens most of the time is that people don't just want to analyze or expose the raw data in those systems. They actually want to do some additional work on that before they do their analysis. So that additional work, just to give you some color here, it may be as simple as I have an index in Elastic or maybe I have an S3 bucket that has data about businesses, but maybe I just want to do an analysis of businesses in the U.S. So I'll create a virtual dataset on top of that kind of physical dataset that has some additional filters in it and I can either do that visually or if I know SQL I can actually do it through SQL in Dremio's interface. So that's creating a new virtual dataset.

Another example would be if I wanted to join data between two different systems, so I could take two physical datasets, one in Oracle, one in Elasticsearch and I could then join those two things either visually again and our user interface or through SQL and then save that as a new virtual dataset. So that virtual dataset actually doesn't contain a copy of the data, but if I connect now Tableau or Power Bi or Looker to Dremio, that virtual dataset appears to the BI tool just like any other table. So the BI user can then start exploring and analyzing that virtual dataset.

[0:32:33.5] JM: How does Dremio discover the schema of my different physical dataset sources?

[0:32:41.7] TS: Yeah. A lot of work goes into both the way we connect to different data sources and understand their schemas as well as how we deal with changes in that schema. So, first of all, a lot of systems have kind of self-describing data, right? If you look at, say, parquet files on S3, we can interpret the schema from those parquet files. If you look at Elasticsearch, it has something called mappings, and those determine kind of the schema. Of course, every relational database has a schema. JSON documents are generally self-describing.

That said, in many cases where it's not a simple table, maybe you're connecting to an S3 bucket that has – Or a directory there that has files that may have different structure in them. So we have this kind of schema learning engine where, over time, as we're observing data through the execution of queries and we're kind of adjusting our internal understanding of what that data looks like and what that schema is. So we have this entire kind of learning algorithm around schema. We call it schema learning engine.

[0:33:39.8] JM: The reflections that you talked about, this important smart caching layer that makes the queries that analysts and data scientists are going to have, it's important that this this query system is a little bit intelligent and can do things on its own rather than having the engineers specify everything. How does Dreamio figure out which queries, which reflections to materialize into a file that will accelerate the actual usage of Dremio?

[0:34:19.7] TS: Yes. So there are kind of two aspects to how reflections work. One of them is how to decide which reflections to create. Then the second one is when a query comes in, how do we even figure out that we can leverage one of these many reflections that you might have in the system, right? Actually, by the way, the reflections get stored in something like an S3 or an Azure Data Lake Store or HDFS typically.

[0:34:43.4] JM: So it's cheap.

[0:34:44.2] TS: Very cheap. So it doesn't have to be kind of fit in memory. You don't have to have like loads of memory in the system, which is typically where these things get really expensive.

So to start with, when a query comes into the system, we have kind of a sophisticated optimizer that looks at that query, compiles it into a query plan and then basically runs a variety of algorithms to understand whether one or more reflections that are available kind of in this storage layer could potentially be used instead of scanning the raw data. So that's where we'll potentially rewrite the query plan internally so that instead of scanning a trillion records maybe we only have to scan a billion records and then kind of roll that up and do some additional processing on that to return the answer to the user, which is the exact same answer that they would've gotten if we had scanned the raw data. So that's really kind of the query substitution layer, the reflection substitution layer where we're trying to take query plan and understand whether we can accelerate it by rewriting it to use reflections.

The question of, well, in the first place, which reflections get created in the system? We have two things right now that we do and then something more significant we're working on. So what we have right now is, first of all, users can basically vote on specific datasets in the system. So if you're working with the dataset, whether it's a physical dataset or a virtual dataset and things are too slow, you basically have –It's almost like a Facebook like button where you kind of up vote that specific dataset and then kind of the administrator of the system can see the votes and see which datasets people are more excited about and then they can kind of enable reflections on those.

Then when it comes to an individual kind of dataset in the system and wanting to create reflections on those, we provide some kind of basic recommendations based on things like the cardinality of different columns. There's also something we're working on now, which is more based on kind of the user behavior. So given amount of capacity that you're willing to allocate, let's say in your S3 kind of buckets, basically a quota. Automatically determining based on query history what is the best bang for the buck in terms of the right reflections to create. So that's something we're working on, basically kind of a very sophisticated machine learning engine.

In addition to that, it will always be important to give users and the admins of the system some kind of controls around this because we have customers, to go back to your CFO example, the CFO might be doing something that's pretty unique. Nobody else in the company does it. So it's not very common, but because they are the CFO, they kind of expect things to be fast. They're maybe more important than other people in the company, right? So that's something that would be hard for a system to know maybe without connecting to their HR database. So we'll still always give people the controls to be able to kind of even all the way to manually defining reflections to create.

[0:37:36.2] JM: What are the steps to executing a query against Dremio?

[0:37:41.5] TS: Actually, Dremio looks to external tools. Dreamio appears like a relational database. So if you connect a BI tool like Tableau or Power BI to Dremio, it thinks that it's connected to a relational database. So you can just kind of drag-and-drop things in the interface, create a new chart, create a new report or a dashboard and things will just work. So we provide ODBC, JDBC drivers as well as a REST interface. Some tools just already support Dremio natively and you don't even need to use any of these drivers.

If you're more of a data science type user, data scientist, a lot of our users use Jupiter Notebooks as a way to interact with Dremio. So we have very nice integration with kind of the Python stack and Pandas specifically. Part of that comes from the fact that we created a project called Apache Arrow about a year and a half ago, and Arrow, since we kind of open-sourced that and worked with the Python community, that since kind of taken – Or really grown in adoption, almost 200,000 downloads a month now and it's embedded into everything from Pandas, to Spark, to H2O, InfluxDB, and we're working with various different organizations and companies, like Nvidia, for example, on Arrow. So that ability of Dremio to integrate very well, especially with the data science tools, is something that's very unique here and that's why we see a lot of our users also using things like Jupiter and Pandas and the entire ecosystem on top of that.

[SPONSOR MESSAGE]

[0:39:14.9] JM: Test Collab is a modern test management solution which integrates with your current issue manager out-of-the-box. For all development teams, it's necessary to keep software quality in check, but testing is complex. There are various features to test. There's various configurations and browsers and platforms. So how do you solve that problem? That's where Test Collab comes in. Test Collab enables collaboration between software testers and developers. It offers wonderful features like one click bug reporting while the tester is running the tests, collaboration on test cases, test executions, test automation integration and time tracking. It also lets you do a detailed test planning so that you can configure platforms and browsers and configurations or any variables in your application and divide these tasks effortlessly among your team for quality assurance.

All of the manual tests run by your team are recorded and saved for analysis automatically. So you'll see how many test cases passed and failed with plenty of QA metrics and intelligent reports which will help your application's quality. It's very flexible to use and it fits your development cycle. Check it out on testcollab.com/sedaily. That's T-E-S-T-C-O-L-L-A-B.com/ sedaily. Testcollab.com/sedaily.

[INTERVIEW CONTINUED]

[0:40:54.2] JM: If I'm an engineer at a company like Coca-Cola or an Airbnb or any organization that's large enough to have a lot of engineers and multiple data scientists and multiple business analysts and disparate data sources, it will be nice to have this data tool that stretches across my entire organization and I can go into this data tool and connect data from one piece of the organization to another.

Unfortunately, it's not a practical reality not only from the engineering standpoint but from a data governance standpoint because there is the principle of least privilege as it applies to data, because if you have a 10,000-person organization, you should not have access to all of the data in that organization. There's privacy rules and there's – Just certain teams should not know what other teams are doing. There's, I think, the term Chinese wall is sometimes used at least in financial institutions where one part of the organization can't know about data in another part of an organization.

So I think that's one thing that leads to silos, but in some ways it's good that there are silos there. So if you're trying to build a tool where you can join disparate datasets, the tool has to be compliant with those data governance walls. How do you handle that aspect of large enterprises?

[0:42:30.5] TS: Yeah. I think you're hitting on something very important here, which is companies want to – We're offering really kind of a data as a service platform and that's because companies want to offer data as service internally and there is no practical way for them to go about doing that. So that's kind of the fundamental problem that we solve.

I think a big part of this though is also making sure that users or consumers of data are only allowed to see what they're supposed to see, right? So when we connect to various data sources, first thing is we always observe the permissions of the user within that data source. So when we're talking to – When we're getting data from HDFS or let's say something like a relational database, we’re actually leveraging the user's identity to make sure that we’re only getting things – We’re only returning things that they're allowed to see and that actually works throughout the entire kind of caching layer as well. We always make sure that a user will never get data they’re not supposed to see.

But then also in the abstraction layer here, when it comes to these virtual datasets, we actually make it possible for companies to control who gets to see what data. So as a data engineer, you may be responsible for making sure people should only get access to whatever they're supposed to get access to and maybe you don't want to expose the raw data that you have, let’s say, in your data lake. So using Dremio, you can actually control that further and you may say, “You know what? I'm not going to expose the physical datasets to anybody. I'm going to create some curated datasets that have the social security numbers stripped out of them and I'm only going to expose that virtual dataset that's maybe kind of watered down to the analysts.” Maybe for the data scientists, I’m willing to give them a little bit more and they're allowed to see something else and you can actually do this at the column level and based on the users and groups that the company has.

But I think that the thing to realize here is that users will work around kind of the restrictions or the inability to get things, right? They will get their work done in many cases and they’ll work around kind of IT and the governance controls that people have in place. So that's why we believe that the only way to get security is to provide users with a way to accomplish what they want but to enable them to do that in kind of an IT-governed system.

So when you're exposing data through something like Dremio, one of these data as a service platforms, then IT gets to see who's doing what with a data, who's accessing what data. You get to control what they're allowed to see and you get to see the entire lineage of data. This virtual dataset, literally, you actually see a graph where it shows a virtual dataset, what its ancestors are, what their ancestors are. You can kind of browse that graph like you kind of browse a Google Map, right? So that to me is the key, is self-service in many ways is critical in order to achieve security because otherwise people are – They’re downloading data into spreadsheets. They’re sending them around in emails and downloading them and extracting them into departmental BI servers, and it just becomes a lot worse and you don’t actually know what people are doing with the data at that point.

[0:45:38.0] JM: When I started Software Engineering Daily, I started going to conferences, and when I when I started going to these software engineering conferences, originally I was most interested in going to the talks, and I went to the talks and I learned about the same things that I think people who listen to the show learn about. How databases work. How programming languages work. Software architecture strategies and things like that.

Over time, I actually became more interested in the goings-on in the Expo Hall. I'm sure you've been to enough conferences where you've been to a lot of different Expo halls, but these Expo halls where you have all these different companies that are presenting their products because conference goers are walking between the booth at the Expo halls and talking to these different products and the products are making their pitches and giving their vision of the world because this is how they – This is part of the sales process of selling to the engineers, selling to the CTOs, selling to the CIOs, and that sales process has fascinated me over time because if you're selling a product into an enterprise, you have to know where is the entry point. Where are you getting a foothold? Where are you explaining the value? Because Dremio is not a simple product to explain, and a lot of these companies that are selling to developers, it's often solving a very subtle problem that the enterprise may not even – That is being sold to them. They may not even understand that they have. So you often need to talk to the engineer specifically and say, "Look. You have this problem. You know you have this problem and I need you to go back and talk to your CEO or talk to your CIO and sell them on this idea, because it's important."

So in building Dremio, what have you learned? I'm sure you saw plenty of this in MapR. So you already had domain expertise in this to some degree, but in building Dreamio, what have you learned about the entry point? What is the way that you convince people that this is the approach? This is how you solve some of the data engineering, data science, data access problems at your organization?

[0:47:59.2] TS: Yeah. I think in most of the companies that we work with and we're now deployed in companies, everything from large enterprises like Intel, Trans Union, Royal Caribbean all the way to smaller kind of startups, right? By and large, there is an organization that's responsible for kind of the data infrastructure for delivering data as a service within the company. Often, this is kind of the data engineering team. It's the same group that's responsible for kind of data lakes, the data warehouses, the ETL and so forth.

So I think there's a clear – For us at least, there is a clear kind of, let's say, buyer for the technology that – Now, we do always make sure that we're also interacting with the consumers of data, with the data scientists and the BI users because when they see the product they really want it, and so that helps kind of internally within those companies helps them understand the value proposition as well.

But I think a big thing that we're doing and different here, we're all big believers in kind of bottoms up adoption. So if you look at the our executive team, it's a lot of the executives from MapR and from MongoDB as well, and we very much believe in kind of the open source model. So we actually created Dremio as an open source technology. We allow people to download it, kind of that. We have a community edition they can download, run in production. We now have thousands of companies downloading that every month. So that's been very successful, and a lot of the companies that we've went on to do business with over the last nine months since we've kind of launched the product have actually started by just downloading the community edition from the website. I think that makes things a lot easier and it's also very much how people want to consume software these days.

So that's really been our approach and it's been, so far, working out very well just in terms of the volume of these downloads and the wide range of customers that we've been able to acquire across every industry you can imagine, from insurance companies, to the largest tech companies, to then across every continent, from Australia, to Singapore, to different countries in Europe and, of course, the U.S.

[0:50:13.7] JM: In the last month, we've had a few recent shows about different solutions to this data sprawl that we've outlined in our conversation here. So we had to show about Uber. That's been pretty popular. What Uber does is, at the highest level, they expose Presto, basically, which is a MySQL interface that translates queries into whatever kind of backing store the data is stored in. So that's one approach we've heard. Another approaches is Citus Data, which suggests if you get all of your data in PostgreS, then you can perhaps have the PostgreS extensions system take care of all the variability in queries and you can have optimizations in that world, and I know these are not totally disjoint approaches. There's probably companies that will have Presto and Dremio. There’s companies that will have Citus Data and Dremio. There’s companies that will have just one of these three. There's companies that will have completely other things. But when you look at the spectrum of approaches to solving this data sprawl, what are your beliefs about how things are going to change in the future and how you contrast the different approaches to solving that data sprawl?

[0:51:43.8] TS: Yeah. I think one of the reasons we started Dremio was because we saw that with just SQL engines, whether it’d Hive, kind of in the Hadoop space or Presto, etc., that wasn't enough to really solve the problem, to make users self-sufficient with data and to give them the performance that they want. That may work for some very large organizations that are willing to run systems on thousands of nodes and throw hundreds of data engineers at the problem, but there are very few companies like Uber and like Google that have their own kind of internal solutions and that want to do that, right?

What we’re saying is that data as a service doing that internally is much more than just having a SQL interface, right? It's the ability to accelerate these queries so that the BI user can get a sub-second response time when they do have terabytes of data. It's the ability to join data across different sources and to have an interface that looks and feels like kind of Google Docs where people can collaborate and build on top of each other, right? It’s the ability to visually curate data for people that are not engineers as well, because otherwise they constantly bother the engineers.

So I think we look at a much more kind of a full stack approach, right? In many ways, you could think of –If you thought of the iPhone, let's say, and SLR cameras a is competitors, right? You could think of it that way. They both take pictures and I'm sure the iPhone has taken market share away from some of kind of the traditional camera manufacturers in the market. But I think the value proposition is very different, right? With my kids, I go and use the smartphone to take pictures and videos of them is because it's then very easy for me to share on WhatsApp and on Facebook and it gets backed up automatically on Google Photos and all these additional value that comes from that kind of deeply integrated system, and that's kind of how we think about solving this problem. It's not enough to have 10 different technologies that I kind of cobble together and throw a lot of manual work at. We think the experience has to be a lot better.

In many ways, a little bit – Like if you think about what Splunk did. Prior to Splunk, people wanted to analyze logs. It's not like they invented that problem, but they had to cobble together different solutions and use shell scripts and load logs into MySQL and all these different things and a lot of work that came with that and Splunk came and said, "Hey! Here's a much more elegant kind of dedicated solution for this problem," and that's kind of how we think of what Dremio is doing for the world of data analytics.

[0:54:17.5] JM: All right. I know your time is short, but one other future related questions. So much like Google Docs or your camera application or Splunk for logging, these problems that from a high level may look like just engineering problems that don't require machine learning. They're just kind of figuring out the building blocks and then optimizing them by hand.

In 2018, we're starting to see the benefits of putting machine learning in these kinds of systems and even for data platforms. There was that paper from Google that maybe you saw about learned database indexes outperforming these manually created database indexes. What are the opportunities for machine learning in building a better, more efficient data engineering platform?

[0:55:17.9] TS: I think there's a huge opportunity here because you can do so much by just understanding what people are doing and what they want to be doing, and that goes everything from understanding what are the right data structures to create underneath the hood automatically without asking anybody just by observing, for example, the quarry patterns, right? That's why we really like our position as kind of the tool that sees what everybody's doing, sees all the queries that are running across all the different sources and being able to leverage that understanding. There's a lot more we could be doing there for sure with kind of leveraging that knowledge and utilizing to make future queries go faster.

We already do things like recommending joins. So when you look at a dataset in Dremio and you click the join button it will say, "Hey, you might want to join this with this other datasets based on the behavior of other users who have joined that dataset or maybe even something derived from that data set with other things."

So I think just building a tool is not enough, kind of a productivity tool. You really want to be able to leverage that understanding of what people are doing over time and then also how that's changing and also looking at the data itself, right? A lot of things we do is we observe the relationships within the data as we're running these various queries and joins in the system and we can then make smarter decisions about how to accelerate things just by understanding that maybe that is a one-to-many relationship based on our historical kind of observations. So things of that nature I think are – There's a lot of additional opportunity here when you start thinking about, "Okay. I know what people are doing with it. I know what they're accessing, how they're doing it," and so forth.

[0:56:59.6] JM: All right. Well, to close off, what else are you working on Dremio? What do you have in store for the near future?

[0:57:05.5] TS: One of the things that we'll be announcing soon is a new initiative around Apache Arrow. So arrow, I think I mentioned that earlier, is an open source project that we created about a year and a half ago and has really taken off as a foundational component for dozens of different open source and commercial products out there, from time-series databases, to GPU databases, to Spark, to Python and, of course, Dremio.

So we're working on a number of different new capabilities and kind of extensions of Arrow that will make Arrow-based systems anywhere from 5 to 10X faster and also provide orders of magnitudes, faster integration. So, today, the world kind of systems integrate based on very old protocols like in interfaces like ODBC and JDBC. For data science, we think there's a need for something much, much faster. So we're working West McKinney, who's the creator of Pandas and really designing a next-generation interface for data, columnar data, in-memory to move between systems. So that will be something coming up in the next couple of months.

Then a lot of additional capabilities also inside of our own open source platform that includes kind of really advanced workload management capabilities. Many of our customers, like Trans Union, they have hundreds of users that run on the platform and they want to very intelligently kind of prioritize the use of resources among all those different users for a very high concurrency levels. So that's something we call workload management or kind of a next-generation workload management that we're working on.

The ability, for example, as datasets continue to grow in size. How do you leverage both GPU's as well as kind of the available disk space that you have in a cluster so that even if you run out of memory, you can complete all your queries in kind of an efficient way. So lots of optimizations around kind of performance and concurrency and workload management in addition to what we're doing with Apache Arrow.

[0:59:00.1] JM: Very cool. I just want to conclude, but I think I've said this before, but I think it's impressive, the three year or four year, whatever, lead time between starting Dremio and getting to this point where you've got some serious customers, because I think it says something about like the delayed gratification to getting to this place where you really have good customers, or I shouldn't say good customers, but like really strong kind of the name brands. I think it says something about the vision that you had from the beginning, and I am always impressed when a company is able to take a really, really long vision, and I think three years is not tremendously long, but it's pretty long and in the world of software engineering tools. So I'm really happy to see you doing well.

[0:59:52.3] TS: Yeah, thank you. It's been very exciting. People say it takes years to build a database, but for us to be at this point where just seeing thousands of people download it every month, and I think the last few months it's grown 30% or something like that month over month.

[1:00:08.9] JM: That's awesome.

[1:00:09.8] TS: It's really taking off.

[1:00:10.9] JM: That's some compounding. Okay. Well, Tomer, thank you for coming on the show. It's great to have you.

[1:00:16.3] TS: Yeah, thank you so much.

[END OF INTERVIEW]

[1:00:20.7] JM: If you are building a product for software engineers or you are hiring software engineers, Software Engineering Daily is accepting sponsorships for 2018. Send me an email, jeff@softwareengineeringdaily.com if you're interested. With 23,000 people listening Monday through Friday and the content being fairly selective for a technical listener, Software Engineering Daily is a great way to reach top engineers.

I know that the listeners of Software Engineering Daily are great engineers because I talk to them all the time. I hear from CTOs, CEOs, directors of engineering who listen to the show regularly. I also hear about many, newer, hungry software engineers who are looking to level up quickly and prove themselves. To find out more about sponsoring the show, you can send me an email or tell your marketing director to send me an email, jeff@softwareengineeringdaily.com.

If you're a listener to the show, thank you so much for supporting it through your audienceship. That is quite enough, but if you're interested in taking your support of the show to the next level, then look at sponsoring the show through your company. Send me an email at jeff@softwareengineeringdaily.com. Thank you.

[END]