EPISODE 1401

[INTRODUCTION]

[00:00:00] KP: If you haven't encountered a data quality problem, then you haven't yet worked on a large enough project. Invariably, a gap exists between the state of raw data in what an analyst or machine learning engineer needs to solve their problem. So many organizations needing to automate data preparation workflows look to Trifecta as a solution. In this episode, I interview Joseph Hellerstein, professor of computer science at UC Berkeley and cofounder at Trifecta.

[INTERVIEW]

[00:00:30] KP: Joe, welcome to Software Engineering Daily.

[00:00:32] JH: Thanks, it's great to be here.

[00:00:34] KP: Can you tell me a little bit about both your academic and industry background?

[00:00:38] JH: Yeah, happy to. So I go back originally as a database guy. My first job out of college was working at IBM Research, and the stuff I did back then in the early 90s rolled out into IBM Db2 at some point. So like roots in software engineering, relational databases. Did a PhD part at Berkeley, part at Wisconsin with Mike Stonebreaker, if you know his name, founder of the Database Field. And Jeff Notton, who's still around and plays a research role at Google.

Thereafter, I went and joined the faculty as a professor at UC Berkeley. So I've been a professor of computer science at UC Berkeley since 1995 working on all things, data, systems, machine learning, human computer interaction around data. So really anything that touches on data stuff and systems around data.

In 2012, we took a research project off campus called Data Wrangler. And we started a company called Trifecta. So the Data Wrangling project and the Trifecta company are trying to address the question of can folks with good computer science allow people who aren't computer scientists, who aren't software engineers, to do their own data engineering? To be able to get that data to transform it in the form they need it so that they can do their analytics and their modeling. And so Trifacta has been a journey of trying to get empathy with folks who are data engineers, folks who are data analysts, and build out computer tools that can empower them.

[00:02:03] KP: So that persona, when I think of the data analysts, maybe someone just getting started, they're probably pretty proficient at Excel. What do they level up to when they switch to Trifecta?

[00:02:14] JH: So that's a really, really awesome insight, because Excel is an incredibly powerful computational tool that those of us who are trained to write C++, and Rust, and whatnot, don't always appreciate. Excel is a rich programming model. You can do lots of stuff with it, and people do. But it's got a very sort of two-dimensional data model where you can reference cells, and you can reference them relatively, and you can reference them sort of spatially. And those are your variables. And then you can write code on top of that. So folks who are master Excel users are often like pretty smart. They're real sharp.

What happens when they move to learn looking at bigger datasets is they have to stop thinking positionally, because data moves around, right? If you have terabytes of data, you're not going to think of it as a grid with row A and column three, or whatever, row one and column C. So you have to start thinking of collection types, right? You have to start thinking about programming in terms of collections of data, which is where languages like SQL and so on, Excel, so to speak. So that transition can be tricky for folks coming from Excel.

But sort of the bridge there, people typically start with Excel, and then they end up in BI tools like Tableau where they're doing charting, because Excel's charting only goes so far. And at that point, they start thinking about mappings of data to bars and stuff like that. And they start thinking kind of in these collection-oriented ways. And that often leads them to SQL, because oftentimes, you're building charts on top of relational databases. So that's been kind of the traditional arc of how do you get from Excel to more programming. Where Trifacta comes in is trying to bridge in that gap, particularly when the work you have to do the data is pretty complex.

So in a modern data environment, you tend to have data from lots of sources. And it comes in a million different formats. It's often missing information. It often has duplicated information represented in more than one way, like International Business Machines versus IBM. And so getting in there and really cleaning up that data is not a single simple SQL query. It's a whole set. It's a whole sequence of transformations. And so doing that in a rich environment often involves a little bit of visualization to keep in mind and be able to see if you're making progress. Is the data getting better? And a little bit of light coding. And the way that you do that in Trifecta is very visual. And then you have AI assistants behind the scenes helping you to figure out what your next steps would be, and then showing you what those next steps would do to the data.

[00:04:39] KP: Could you elaborate on those AI assistants? How do they help me?

[00:04:43] JH: Yeah. So all over the place, let's start with the simplest thing, which most tools for data cleaning do a little bit of, which is I just loaded, let's say, a CSV file, and somebody is going to have to figure out what the data types are in the columns. Why? Well, if you're a loaded database, you have to. But more to the point, if you're going to give me like a bar chart for each column that tells me what this column looks like, you kind of have to know like is this numerical data and it's ordered? Or is it categorical data? It's names, and it's not ordered. And we need bars from like most frequent to least frequent. So even just getting your eyeballs on the data, you kind of need to know something about its data types. And then you might want richer models like, "Oh, these are credit cards, or these are social security numbers, or these are drug names from my catalogue of drugs, because I'm a drug company," right? So typing data is one kind of place where a little bit of inference, a little bit of AI can help.

Another really common example, I missing information in a column. And I want to fill it in with the most likely values. It's called data imputation. So that's a sort of sequential model that you try to figure out if I've got a sequence of numbers that I'm missing some things. What should I put in the holes? So you can use a variety of sequential models to do that. So data imputation is another.

So a third example is what we call entity resolution, and that's this issue where you have different names for the same thing, or possibly the same names for different things. So for example, IBM International Business Machines, and the Indiana Beer Manufacturer, right? And you need to figure out which things are which. So that's another kind of modeling or AI technique that you'd use in one of these tools. And there's a dozen more.

One that we use a lot in Trifecta is what we like to call predictive interaction. So this is actually how you interact with the tool graphically. In Excel, if you double click on a cell, it's because you want to edit it. In Trifecta, if you point at almost anything on the screen, whether it's a chart, or a value in Excel, we give you a list of possible interpretations of what that might mean. So you click on a column, and maybe that column is missing a lot of data. We might say, “Hey, are you trying to fill in the data that's missing from this column? Or is it that you want to change the data type for this column? Or is it that you want to delete this column?”

And the way that we show that to the user is we show them a panel of the possible things that would happen on the screen if you chose these different things. So you can see what would happen if you chose each of these different choices. And it sort of speeds you along your way. So you don’t have to try out 17 things. You can see in a rank list, essentially, 17 possible things you might do to this column just by clicking on the column. And it's the same for clicking on a bar in a bar chart, or clicking on a cell in the spreadsheet. You get suggestions for what it is that, because you highlighted this feature, you might want to do next.

[00:07:17] KP: Well, the common acronym most people use is ETL, extract, transform and load. Although I think there's some debate about the proper order there. Where does Trifecta fit in the typical data pipeline in architecture of a software company?

[00:07:33] JH: So let's start with E and L. Traditionally, we had operational data stores. Think of transactional databases. This is back in the 90s when the ETL moniker came up. So you do transactional databases. And then you have your analytics platform, which was built on your data warehouse. You need to extract data from your transactional databases. Typically, there was more than one. And you need to integrate it and load it into your data warehouse for analytics. Much has changed since the 90s. The first thing that's changed is you don't kind of have to necessarily have this IFK operational databases and one data warehouse. Typically, you have a whole bunch of stuff going on. You have data coming from all over the place. You have different people who are trying to interact with that data.

Another thing that's changed a lot since the 90s is that we're in the cloud now. And you can put your data almost anywhere. You can put it in cheap storage, like block storage, like S3 at Amazon, for example. Or you can register the data with a cloud data warehouse, like a Snowflake, or Redshift, or BigQuery. And you can have that data either ingested into the data warehouse, or you can leave it in black storage raw, and have the database interpreted on demand when you run queries.

So what this means is that transform part, the T, gets to be done kind of however you want. You can do it with the databases query engine, or you can do it with programmer tools like Python Pandas, or you can do it with like big data frameworks like Spark. There's lots of ways you can do T, right? And so the traditional ETL was extract from the operational data warehouse by a specialized system to do T, like an ETL system, and then load the data warehouse. Today, it's much more fluid. It's much more like there's going to be sources, there's going to be destinations, and there's lots of ways to do transformation.

Trifecta fits into the world by being flexible. And so we can do what's called ELT, where you do extraction, and you load a SQL data warehouse like Snowflake. And the transformation comes third, because the raw data is now in the data warehouse. And you're doing SQL for transforming it to your final state. And if you want to do that pattern, you use Trifecta's user interface and Trifacta will generate the SQL transformations. You can also choose to do a more data lake style where you take the data and you dump it into blob storage like S3, dump it into a data lake. And then you can use programmer tools like Spark to transform the data. And in that environment, Trifecta's user interface will generate Spark code for you if you like and run the transformation before loading the data warehouse using Spark.

So we are basically an authoring environment, a monitoring environment, and a runtime environment that would kind of keep tabs on what you need to do and allow you to specify what you want to do. The execution engine and the order of weather happens before E, after E, after L, that's kind of up to you based on the infrastructure that you run.

[00:10:17] KP: When you look at the titles of the people that engage with the platform, who's the typical operator? What job profession are they?

[00:10:24] JH: Yeah, so it's broad. But let me say that Trifecta starts out with the hypothesis that we should allow lots of people to do data transformation and data engineering, simply because there aren't enough highly technical people to do it, relative to the number of people who want to work with data. So if you look at all those super users of Excel, if you look at all those users of BI tools like Tableau, if you look at all of the people who want to use AutoML to build simple models, many of those people are not programmers. Many of those people are not even data engineers. They're data scientists, they're analysts. They have these other role titles. And we need to empower those people, because darn it, the folks who can handwrite the code are too busy to do it for everybody else.

So Trifacta is expressly in the business of using computer science, visualization, AI, etc., to empower all those folks to be able to do self-service data engineering, while also integrating with the folks who are the operational developer-centric folks, the data engineers, the software engineers. What that means is that in addition to the visual interfaces, we need to make sure that what we're generating at the end is code that can be checked into Git, code that can be managed, code that folks can edit if they're code types.

You hear the phrase democratization of data a lot. And for me, what democratization means is that you're bringing lots of different kinds of people together. I think, traditionally, people use it in this space to mean democratization is to make something easy so that people who aren't technical can use it. I don't think that's right. It's about bringing people who are sort of data and computer folks natively, and people who are application domain experts natively, and allowing them to work together, alright? So it's both ease of use and sort of the programmability, the operationalization. All that stuff needs to be served. So I think that's the future of data engineering particularly in the cloud where everything's easy to get going on. And Trifecta is squarely in the business of trying to serve that multiplicity of users.

[00:12:21] KP: I'm going to throw some buzzwords at you. We got no-code, low-code, serverless. How does Trifecta square against these ideas?

[00:12:29] JH: Well, architecturally, Trifecta is serverless. So let's start there. When Trifacta is running in the cloud, you don't need to know how many instances of it are running. You don't provision a Trifacta server in the cloud. You just sign up and use it like software as a service.

And for the user, the fact that it's serverless, is more or less basically is use-based. The pricing is not making you set up servers and rent them on long term. So it is serverless.

In terms of no-code, low-code, it is a low-code environment. But it's also an environment in which code is being generated and can be manipulated and edited. And this speaks to my last point. Low-code users who would prefer to interact graphically are very empowered in this tool and given a lot of hints. Folks who like to code can see code, they can edit code, while they're also getting tons of visual feedback from the tool about like, “Alright, you wrote this line of code. Here's what things look like now in your data.” So that interaction with the visual is available to real coders and it's also available to low coders. And we try to span that gap. So I would say Trifecta is unique in being a powerful low-code environment for data transformation that also embraces the idea that coders are part of the team.

[00:13:37] KP: So I see a lot of opportunities as I'm working on my data pipeline. But if it's just a data pipeline, that's kind of set it and forget it. I don't want to know about it after I built it, unless there's a problem. It sounds like Trifecta can be more of an analytical workbench for me as well. Could you describe what the day-to-day is like for a lot of users in that regard?

[00:13:56] JH: Yeah. So day-to-day, there's no such thing as real set and forget with data pipelines, because as you alluded to, you need to keep track of whether it's doing what you expect it to be doing. So daily data quality, for example, is a really important piece of owning a data pipeline. And so part of using Trifacta is the ability to do data quality in rich intuitive ways so that you can get your eyeballs on things and quickly see if they're working or not working, establish data quality rules, check how much drift from what you want to expect it in the data, etc.

So part of the operationalization aspects of Trifecta are data quality, and scheduling, and all those things that make – That allow you to pretend you're going to forget it, even though, honestly, you said it, and then you keep an eye on it over time. And then the other thing that's important to note is that everybody's data environment is on the moving dynamic. People are always getting new datasets, new use cases. And so we rarely see customers who set and forget in any realistic way, even with respect to authoring transformations. Sort of a constant maintenance and constant continuous development process really around data pipelines.

[00:14:58] KP: If I'm maybe a slightly more sophisticated user with a computer science background who still just wants a tool to be efficient, how much fiddling of knobs do I get to do and things like that when it comes to the imputation, or the entity resolution, and these sorts of steps?

[00:15:14] JH: That's a good question. So from a transformation language perspective, you can do as much fiddling as you want. So if you want to write your own SQL queries and string them together, you can absolutely do that. If you want to program in Trifecta's internal domain specific language called Wrangle, you can write your own Wrangle and paste it into the tool and check it into GitHub and so on. So all that stuff is very much in the hands of the developer in a way that any coding tool would allow. So you can see it as a super-duper IDE if you like.

As regards some of the specific models we use, like our entity resolution models, or our type induction models, those are more or less extensible given the model. And so it depends. So for example, the entity resolution model allows you to choose your similarity metric, and you're clustering strengths and variety of parameters in the model, if you're familiar with that stuff. Our type induction model allows you to register your own dictionary. So if you have a list of your vendors' names, that could be a data type in your system that you can register. If you have a regex for what your customer IDs look like, that's a data type you can register. And then the system will use that in the model to induce like, "Oh, I think this column is of type your customer."

For all the models, we've tried to make them extensible. But AI modeling is an art and a science both, and some of these models may or may not suit the use case. So another really important part of Trifecta is it's very API-driven. If you want to integrate Trifacta into a longer pipeline with other tools, you can absolutely do that. And we have tons of customers who use Airflow and other things to kind of put Trifecta into a pipeline and then step out of Trifecta to call out to external code and then jump back in.

[00:16:44] KP: Are there any customer stories or use cases you can share about Trifecta in action?

[00:16:50] JH: Yeah, I mean, my favorite use cases recently have to do with COVID-19. So the US Center for Disease Control, the CDC, has been using Trifacta to understand kind of how outbreaks and transmission chains spread. So they're using Trifacta to automate your time consuming labor intensive work of stuff like entity resolution, stuff like date cleaning, things like that. So in all these examples, there's a myriad of dirty work that trifecta is helping with. So CDC is one example that we're super proud of that they're using Trifacta to help map COVID.

Similarly, at the University of Oxford, they have something called the Infectious Diseases Data Observatory. It's one of the largest international collections of clinical data for COVID-19. And they're using Trifacta to rationalize this data that's coming in from spreadsheets and packages and a whole bunch of other sources. And yet a third one is Genomics England, which has been looking at the genomic spine patients who've been diagnosed with asymptomatic and long haul COVID and so on. And they also are cleaning up multiple data streams using Trifacta. So that's just one sector around healthcare specifically in COVID. Obviously, we have customers across multiple sectors, because data is such a pervasive thing. So in banking, in pharma in nonprofits that are doing things like environmental work, just all over the place, we have use cases. But I love the COVID use cases, because they're so personal right now.

[00:18:10] KP: What sorts of insights do people gain from the projects?

[00:18:14] JH: I mean, all kinds of things, right? Because what's happening is once you clean up the data, then you do your downstream analytics. So sometimes were being used to build charts that scientists to look at. Sometimes were being used to build clean rosters of data. A favorite example, there is a firm called Nation Builder that took voter rolls in the United States from all the counties and all the states in the US to build out voter rolls for the entire country that were normalized. And there they were comprehensively building a dataset that was clean.

So depending on your downstream use, you might be generating like a reference data set. You might be generating charts. You might be training an AI model. We see that quite a lot too, where you're doing essentially feature engineering with Trifecta. So there's lots of downstream use cases that follow from the work of data engineering.

[00:18:56] KP: a lot of the data that flows into our systems was either put there by a data engineer, or through a process created by a software engineer. And data quality isn't always paramount as the value for someone like that. Sometimes it's a person worrying more about encodings, and quality, and how to get the throughput fast enough, and all this sort of thing, which can be one of the sources of some of the data quality issues we might face. Not to say that those two roles couldn't be the champion bringing Trifacta in house. But I guess that's my core question. When an organization starts to adopt your solution, how do they get in the doors? Who's the first customer?

[00:19:32] JH: That's interesting. I don't want to throw too much shade on software engineers, having been trained as one myself. I got to say that like there's a lot of mindsets that you can bring to data to build quality pipelines. And it has multiple aspects including the operational concerns of kind of the software engineering.

Having said that, Trifecta, because it's sort of positioned as being a graphics forward tool, tends to come in through the data scientist, the data analyst, and the data engineer. And in the data engineering space, it's the less programmy engineers who tends to go for it first.

The primary issue here that I see is that software engineers are not trained to look at the data all the time. We have a tendency to sit down, write a lot of code, debug or write unit tests. But we don't tend to stop and say, “Hey, does the data look right at line 12? Does the data look great at line 14?” Because guess what? Our tools make it really hard to draw a chart. You have to write another little program each time to write a chart.

So if you look at people's Jupyter notebooks, data scientists, there's long cells that are just generate a chart. Trifacta always has a chart in front of you. Every time you write a single line of code, the charts change to reflect what that line of code did. And I believe this is super important not just for data analysts, but for software engineers, for hardcore programmers, to get that continuous visual feedback about their data.

[00:20:49] KP: I’d love to talk a little bit as well about your work investigating bias in the criminal justice system. Can you give me the high level? What's the nature of what you're doing?

[00:20:59] JH: Yeah. So we have a project at UC Berkeley. This is actually separate from Trifecta entirely. We're building a lab at UC Berkeley called the EPIC Data Lab, which stands for effective programming interaction and computing on data. But mostly, it's a backronym, because we thought it was epic. And I'm working with some colleagues in human computer interaction, in program synthesis, which is a sort of AI-oriented branch of programming languages that generates code. And we're looking at Trifacta-like problems. And the context we're looking at it in is a context that came to us on campus from the Innocence Project, which is a nonprofit that's trying to make sure that innocent people don't go to jail, and from the National Association of Criminal Defense Lawyers, NACDL.

So those folks are interested in making sure that the justice system works. And what that entails is that the prosecution and the defense both make the best case and the jury and the judge decide on the results and the sentencing. Now, if one of those sides doesn't have access to the same sort of tools and data that the other side does, the justice system doesn't work. And we're in a state right now where prosecution tends to have more access to technology than defense.

So these defense lawyers are interested in finding ways that they can use data to do their side of the job. And in particular, this has come up, for instance, with police misconduct. So when you have someone that you're trying to defend who's going on the stand, police evidence is considered to be truth, unless you can discredit the police officer on the stand. Otherwise, it's taken into the record as truth.

And so, for instance, if you have cops with a history of doing wrong stuff, you want to know that that cop is that person you think it is. So if you have Officer Smith who's making a statement about what they saw, but it turns out that Officer Smith was discredited in a previous case, in a previous like different jurisdiction, he may have gotten let go of that police force, gotten rehired in a new county, and you need to establish it's the same Officer Smith. So it's exactly entity resolution is one of their problems.

But generally speaking, they just have tons of paperwork and data that they're trying to sift through, and they are lawyers, and they're underpaid, and they are in a rush. So what can we do to help them look at scans of police document? Look at body cam data from cops? Look at

911 call records, and quickly get to the bottom of are the officers in this case to be trusted or not?

And so they came to Berkeley as a place where state of California just has a state bill that releases a lot of this information, this public information. And they're trying to build up a capacity to let lawyers do their own data analytics and data transformation. And the data ingest problem is a big one for them. And it's multimedia and kind of all the kinds of new stuff we're looking at with different formats.

[00:23:37] KP: There's a lot of rich data set you described there. The body cam is a computer vision problem. The transcript data is a natural language processing problem. And we've seen some pretty impressive advancements in those areas over the last decade, yet nothing's passing the Turing Test quite yet. What is the state of the art? And what kind of questions are reasonable to ask of the data today?

[00:23:59] JH: I'm so glad you asked the question in the terms you did. Because AI is absolutely not a panacea, even though we're seeing enormous progress. And what that means is that we have to look at AI as an assistive technology in human workflows. If we're going to take body cam data from officers, and we're going to give it to an algorithm, if the algorithm just says good cop or bad cop, you can't really work with that. That's not trustworthy. What you need to do is you need to have something more like the algorithm comes back, it tags moments in a video. It says, “Here's moments where there seems to be excessive use of force. You should review,” right? So that's just a simple example where the AI is actually in service of making the human go faster in scrubbing through this hours and hours of videotape.

And of course, that's fraught with ethical concerns too about recall and precision. But at least the agency is in the hands of people. And I believe that it's going to be a long, long time before AI technology is such that we'll just say, “Hey, algorithm, figure stuff out for us on the large.” Almost always going to be human in the loop AI is embedded in assistive tools.

So the spirit of what Trifacta has built, which is AI in the spirit of cleaning up kind of tabular, semi-structured and structured data, we're going to see more and more of that as we start looking at new data types that we have to use AI to do feature extraction, like audio, like video, like OCR, like natural language. But I absolutely agree with you. Like this is not going to be magic. This is going to be better tools with a give and take between human intelligence and artificial intelligence.

[00:25:28] KP: And when you think about that intersection, for people who are really want to focus their careers there, do you have any advice for what the opportunities are?

[00:25:35] JH: Yes, I do. And my strong advice to folks – And I say this as a Berkeley professor, as well as someone who's in the industry, is don't focus myopically on AI. Focus on the human computer interaction around techniques like AI, because that is where the real impact is. Certainly, true in products. You can't take AI techniques off campus and just sell them. They're just pattern recognizers. They don't actually solve a problem to completion. So everybody who starts a company realizes this at some point. But you can easily go and get a degree in AI and think that you're solving the world's problems only to have that cold water in your face when you take it out into the real world.

And I hear over and over from my colleagues who are full time AI researchers that this human in the loop aspects of AI are the next frontier. So I strongly encourage people to think about how does AI fit into a workflow? How does human perception and computer perception complement each other? What are people good at? What are computers and algorithms and models good at? Those are deep and interesting questions that lead to really useful outcomes.

[00:26:36] KP: Well, we touched briefly on your time working on Db2, when database systems were these monoliths centralized. And that model still exist. But we also live in this distributed world. You'd alluded to some of these upcoming data sets and things like that. I'm curious where you see Trifacta going in the next few years. Are there any things that you can share on the planning or horizon? Or how will the product evolve as data evolves?

[00:27:00] JH: It's a great question. Well, I do think that more and more of our databases, obviously, will be in the cloud rather than on-premises. And it's been a big focus of Trifecta over the last few years to make sure that we are a cloud native product. And we are. We're offered on all three of the major public clouds in the Western hemisphere. And Google has chosen us as their preferred data platform. It's called Google Data Prep by Trifacta.

So being cloud native is like absolutely required. At which point, distributions are given, right? The cloud is not one computer. Cloud is lots of computers. So it's absolutely distributed data, distributed compute. And we've been working on that for some time.

Having said that, where I see things going forward is this definition of democratization is spanning developers as well as subject matter experts. So think developers and business users, if you will. That spectrum is, I think, squarely where we are focusing over the next few years, is in making sure that we're serving not only the Excel spreadsheet types, but also the Python programmer, also the SQL programmer, also the pipeline management ops person, that all those personas are democratize, which is to say they all have a seat at the table in this tool chain. They get to bring their languages. They get to bring their favorite tools, and interoperate in a smooth way that doesn't lock people out. And that's actually not as simple as it sounds. It's actually pretty – Takes a lot of finesse to build an ecosystem where those folks can work well together. And that's something that we're laser focused on.

[00:28:27] KP: I'm wondering if you'd imagine with me a small startup, founders bootstrapped it, they're doing well, they're selling subscription widgets, the thing runs on GCP. They got three people and they're ready to grow. At what point do they turn Trifacta on?

[00:28:40] JH: The minute they have messy data, basically. And we get this with like nonprofits that are one people. If you have a really yucky looking spreadsheet, and you want to clean it up, you might just go up to use Trifacta for a day. And that might be a way you get started. It'll tell you what's in your data, what's not in your data. It'll help you figure out how to clean it up and give you the tools to do it. And it's serverless. So you're kind of paying for usage at that point. It's cheap.

So we see it like at the very beginning scale. We also see people set up multi-terabyte pipelines with Trifecta that run every day, where the operationalization features of the product are being used. And there's multiple people who are checking in on the health of the pipeline and managing it. Different people authoring it and they're all collaborating within the Trifacta ecosystem. So really, it does span from the individual to the large enterprise.

[00:29:29] KP: And then do you sit on top of my existing data infrastructure? Or are you ingesting into Trifecta and mirroring something in some way?

[00:29:38] JH: Yeah, I'm so glad you ask that. It's been a tenant of the design from the beginning that we do not ingest data. We do not process data. We look at data, and we give you examples of it so that you can manipulate it visually. But when we're done, we generate code. And that code runs in your native environment. So if you're running over, say, snowflake, what's happening is we ingest some samples from Snowflake. You work in your browser. You clean that up. You generate a series of SQL queries that we push down to Snowflake. Similarly, with Google and BigQuery.

If you're doing this in a Spark environment, we're generating Spark. But the bottom line is like wherever your data is stored, the runtime you use, that's your choice. We don't pull the data out other than to put it in front of your eyeballs in the browser. But beyond that, we're pushing code to data. We're never pulling data to us. And that's so important. If you're a database person, you know this deep in your bones. You always push code to data. You never move data to your system. And many of our competitors got that wrong and are now by the wayside, because you'd have to stand up a cluster in front of your cluster to run their tools. No cluster required for Trifacta. It's just an environment for authoring, monitoring, visualizing.

[00:30:45] KP: Well, tools like these make the data more accessible to a wider group of professionals, as you said, not just a computer scientist, but someone with some analytical skills can now come in and explore. With that in mind, what do you think the landscape of what a data scientist, or a data engineer, or a business analyst does a few years out? How is that evolving?

[00:31:05] JH: You know, I think that could go a couple different ways. And as I asked this question of colleagues and friends, I get different answers. So let me paint two pictures that are a little bit different. And frankly, these are pictures you could find today in the wild. So which one of them will be more common is unclear. In one picture, what's happening is that folks who started out as analysts and data scientists become increasingly technical and go further upstream into the data. So they started out just building models or just building charts. But then they're doing their own data transformation. And then they're doing their own scheduling and operationalization. And then they're doing their own acquiring of data from data sources, and they own a full pipeline.

And in that world, data analyst, data scientist, data engineer becomes kind of just one job category. The tooling, if it can support that, allows the person at the end of the pipe who understands the business problem to own everything. So that's one picture.

A different picture, which you see in other shops is what we really want is we want our domain experts, the people who understand, let's say, the business use case, to focus on the business use case. We want the data engineers to focus on operationalization. And we want the software engineers to write any custom code we might need for our particular business. But we want all of those groups to be able to interoperate super cleanly, and transparently, and seamlessly. So I want the data engineers and the business person to be able to talk to each other very fluidly about what's working, what's not working, what they need, and how to get it. And that the tooling needs to support really agile work across specialties and across organizations. Because frankly, when you hear these stories that like 80% of the time in a project is spent in wrangling the data, a lot of that 80% of time is like translation error, where the business user said, "I think I need a chart. Will you cut me a piece of the data that will build that chart?" Data engineer goes and does it and give it to the business user, and the business user is like, "Actually, this is not what I wanted," right?

So you got to close that loop tightly, either by making it one person, or by making those personas able to interoperate really efficiently.

[00:33:02] KP: Well, for listeners who want to see if Trifecta is the right solution for their data quality issues or other data processing needs, what's the best place to go and learn more?

[00:33:11] JH: Simplest thing to do is to go to trifecta.com. And there's a button on there that says, "Start wrangling," and it'll let you set up in your favorite cloud and get going. It's free trials. So you can get your hands on it and your eyeballs on it most importantly and see what it looks like. I have to say Trifecta is a story that tells itself better in pictures than in words. It's a very visual tool. It's very intelligent. And it will speed you along your way in ways that it takes too long to describe. It's not your typical boxes and arrows drag and drop ETL tool. It's something quite a bit richer and higher tech.

[00:33:43] KP: Yeah, definitely a very visual experience. I absolutely agree. Well, Joe, thank you so much for coming on Software Engineering Daily.

[00:33:49] JH: Thanks for having me. This was fun.

[END]