SDS PODCAST EPISODE 1021: HOW DBT WON ANALYTICS ENGINEERING, WITH DBT LAB'S CEO TRISTAN HANDY

Jon Krohn:	00:01	Today's exceptional guest coined the term analytics engineering. He built the tool a hundred thousand data teams rely on and now says agents can compress a million dollar year long data migration into just six weeks. Welcome to episode number 1021 of the SuperDataScience Podcast. I'm your host, Jon Krohn. Today's guest is Tristan Handy, CEO and founder of DBT Labs, the company behind, you guessed it, DBT, which is the beloved open source tool that brought software engineering rigor to data transformation. A decade after founding DBT Labs, Tristan is now merging the business with Fivetran and becoming president of the combined company. In this episode, he explains why he turned down acquisition offers for years, how the semantic layer keeps AI agents from confidently getting your metrics wrong, and how skill files let agents do work that used to take humans weeks of training to learn.
	00:55	Enjoy. This episode of Super Data Science is made possible by Anthropic, Notion, and Gurobi. Tristan, welcome to the Super Data Science podcast. A treat to have you here. Where are you calling it from?
Tristan Handy:	01:07	Oh, yeah. Thanks for having me. I am in the great state of Pennsylvania. I live in the mainline outside of Philadelphia. That's
Jon Krohn:	01:15	Right. And your company was originally called Fishtown Analytics. Is that Philadelphia is also known as Fishtown, I guess?
Tristan Handy:	01:22	Well, Fishtown is a neighborhood in Philadelphia. When I started the company, I lived there. Yeah.
Jon Krohn:	01:30	Gotcha. Yeah, that makes a lot of sense. I probably should have looked into the facts behind that before asking you. It's

Tristan Handy: 01:35 Been a long time since, but originally it was the biggest cod fishing area on the East Coast. I came to learn. Who knew?

Jon Krohn: 01:45 There you go. And so that was a decade ago that you founded Fishtown Analytics. Now DBT Labs as most people would know it. You've raised hundreds of millions of dollars over the year and your platform is trusted by over a hundred thousand teams today. That's a wild journey. In 2016, you articulated a vision that data teams deserve the same rigor and tooling as software teams. And you coined the term analytics engineering. What led you to that insight in 2016? And what has the journey been like to today?

Tristan Handy: 02:20 It's a good question that I don't often get asked. Why did I think that that was true? It was not like some bolt of lightning insight. The prior company that I'd worked for had a bunch of smart people that worked at it. I ran marketing. The company was called RJ Metrics. And we were a kind of prior generation BI. We could, in our numbers, see the market shifting in the 2014, 2015 time horizon. A lot of the really early adopting tech forward companies were leaving. So we did a study. A bunch of us on the exec team talked with, I think the grand total was something like a hundred companies. And we found out that they were moving to what eventually became known as the modern data stack. Back then it was people were storing their data in Amazon Redshift and they were using BI tools that were purpose built to run on top of Amazon Redshift.

03:23 But the funny thing is they were using this new technology, but they didn't. A lot of the problems that they were experiencing were the same problems that they had experienced before. New technology, same problems. And you're like, "Well, what's going on here? Have we failed to learn something important?" And I could talk about this for about forever, but the conclusion that I

came to along with some of the other folks on this team was that we were asking data people to build production systems. The cloud meant that everything was always on and there was an expectation that you could hit the button and run in the top right of every dashboard and that it would produce new data and that data was always correct. And so this is really a production software system, but we were not equipping them with the tools to do that.

04:09 And so that's kind of where everything flowed from that has happened in my little neck of the woods over the past 10 years.

Jon Krohn: 04:17 And more recently in 2020, you said that DBT's purpose is to empower analysts as first class owners of another transformation process. So in 2016, you saw this transformation process, this opportunity to be providing tooling to enable analytics engineering to form it all. And then in more recent years, there's this desire to empower analysts to be transforming whole organizations. It seems like power has shifted as data capabilities, analytics capabilities, AI capabilities have become so important in organizations. There's been this power shift more and more towards analytics type people. And yeah, so how are you balancing empowering us technically without overwhelming us with software engineering complexity?

Tristan Handy: 05:09 The thing that I think is the most magical about people who call themselves data analysts is that they're not afraid of complexity and they tend to dive into kind of multifaceted problems. Now that sounds kind of like high level and vague, but the thing is that data analysts are a role that spans two big disciplines. One is the actual quantitative discipline. You have to actually know something about math and statistics. And I would even put the software engineering part in here, like the kind of more technical mathy brain stuff. And then on the other side, you have to know about the actual business

domains that you're working in because a data analyst that doesn't actually know anything about the domain that they're working in is really just an order taker. They're not adding any value beyond their technical capabilities. And especially today, when anyone can have technical capabilities with your agent, those kind of can't stand on their own as a source of value.

06:19 Almost everybody finds their way to the data analyst role in some kind of weird data syncratic pathway because you generally become an expert in the one side or the other, and then you learn the other side on the job. The way that I've always thought about building dbt is this idea that we should empower these magical unicorn humans to be able to live up to their potential, but not overwhelm them with technical details at the outset. Now, that's not to say that data analysts are not capable technically, but that's not the first order concern. That's not like why somebody becomes a data analyst. Back in 2016, the kind of cool kid way to do data transformations was Spark. And Spark is exactly the wrong tool if you want data analysts to be able to do any of this work, because right at the outset, it is just incredibly challenging to get it set up.

07:22 The syntax is very complicated. Nobody with a data analyst background kind of knows how to write Spark code at the outset. And so one of the reasons that I chose SQL was to make it accessible to data practitioners, to data analysts. And it kind of goes from there, but the idea inside of DBT is progressive complexity. You start out and everything's super simple. And as you need it, it turns out that there's features that you can take advantage of that help you scale up from there.

Jon Krohn: 07:52 Let's double click on that experience of using dbt for our listeners who haven't had it before. And it's kind of interesting in a podcast format where we have to be able to have an explanation that's easy for people to

understand and potentially an audio only format. Even if people are watching this video of us talking, it's not like just visual cues as to how dbt works. So this will maybe test your storytelling abilities, but I'm sure it's a narrative that's well worn for you over the past decade of running dbt labs. Tell us about the experience of using dbt and how that might be different. Walk through a user story or two of how it's different from SQL or how you recommend analysts getting into dbt. Just fill our brains with what the DBT experience is like.

Tristan Handy: 08:41 The core idea behind DBT is not particularly new. In fact, in the first five years of the journey, we probably came across literally dozens of, I think maybe 50 plus tools that had been built like this before. So the core idea is not unique. The idea is that databases became really powerful for analytics with the launch of Amazon Redshift and then successively with things like Google's BigQuery and Snowflake and Databricks. And increasingly you could rely on databases that primarily spoke SQL to do work that had previously been done by like Hadoop or Spark. And so that meant that you could express it in SQL, but data transformation or taking the raw data that comes from your systems of record, whether those are kind of business systems like Salesforce or whether these are like logging systems like Snowplow Analytics or something like that, that raw data being transformed into something that you might actually write a query against as a business user is a many, many stage process.

09:56 And what you don't want to do is you don't want to have these super, super complicated, monolithic transformations, but you want to kind of stage things out. You want to take your raw data and at first, make sure the column names are reasonable and that you have basic data guarantees like foreign key integrity and not in all constraints and these kinds of stuff. And so you clean it up just a little bit. And then maybe you start to kind of consolidate a couple tables together. You join customers

and orders so that you can get first purchase date or this kind of thing. And as you go, you maybe get to a place where you have a single table that has all of your customer data in it, but really that customer data has to be sourced from like 50 different tables because it turns out that customers touch every system that you have.

10:44 And this process in dbt is accomplished by a series of what are typically like fairly modest SQL select statements. Ideally, you don't have any select statements in there that are over, I don't know, 100, 120 lines long. And each of them does something really discreet. And then they form what's called a directed a cyclic graph dag that goes from left to right and it does all these subsequent stages of processing. And the magic of dbt is that it makes all of this stuff really simple to implement. Building your first table takes basically nothing. You then type dbt run and dbt manifests all of this stuff in your data platform.

Jon Krohn: 11:32 I gotcha. So it allows people to be doing kind of in a simple, this is probably going to be an oversimplification. And you can correct me on what I get wrong here, but it's allowing people to do similar kinds of things as they might want to do in SQL, but easier and at a larger scale faster.

Tristan Handy: 11:51 In SQL, if you are writing a select statement, that is something you might do in your BI tool. You want to build a chart. So you write a select statement that does some group buy and you get some numbers back and you put them on a chart. And that doesn't persist anywhere. But what dbt does is it actually takes all of these different tables that you have defined and it persists them in your database. And there's a lot of work involved in that. How do you do that well, et cetera. But it persists them in your database. And that means that when you go to your BI tool or when you load up your analytics agent or wherever you want to consume this data, the data that these front

ends have access to has all been really nicely modeled and it's all there in the database waiting to just be selected from.

Jon Krohn: 12:43 Cool. And all of this stuff is available open source, right. So listeners can right now make their way. You can very quickly, we'll obviously have a link in the show notes as well, but it's as quick as typing DBT three characters into Google and you will make your way. What does DBT stand for, Tristan, actually?

Tristan Handy: 13:04 Originally it stood for data build tool. We didn't really know what to call this thing. And so the first person to ever write a commit to DBT needed to give the repo a name. And he was just like, "I don't know." He typed DBT on the repo. And at the outset, we thought it was kind of a bad name because it, I don't know, it didn't have any character. But over the years, the DBT community has grown huge and the logo and the brand have a lot of recognition. And so we realized that we had kind of missed the boat to ever change the name. And this

Jon Krohn: 13:43 Is

Tristan Handy: 13:43 In fact how people though of it.

Jon Krohn: 13:45 Oh yeah. No, I mean, I feel really lucky to have you on the show. To me, the DBT brand is like this kind of rockstar name. And so to have the founder and CEO of DBT Labs on the show, it's a great honor. Speaking of the community that you built, dBT Labs is built around an open source foundation and a community driven identity. Even as the company has expanded commercially through acquisitions like SDF Labs. More recently, you had an all stock merger agreement with Fivetran that will see you take on the role of president of the combined company, according to our research. So how do you balance this vision of open source and community

stewardship with practical realities of managing now kind of multiple large, successful commercial organizations?

Tristan Handy: 14:37 Well, hopefully they're not distinct commercial organizations. I think they fit together pretty nicely. But there can be on a kind of tactical level, tension between open source and capitalism. Open source wants things to be free and free isn't beer and free isn't speech. But then capitalism wants things to be monetized. And so I think that that is a common way that people who are not deeply involved in open source see the world and they're like, "I don't really get it. How does that work?" In fact, companies do things all the time that they don't directly monetize. I mean, just for example, the large tech companies in the US have written academic papers for, I don't know, forever, for as long as I'm aware of. The Google's technology that eventually became Hadoop, they wrote about it in a paper. Google also did a paper, some folks inside of Google wrote a paper called Attention is All You Need, which created the entire transformer revolution that we are living in today that's led to large language models.

15:55 So like this stuff happens all the time and there's very discreet business reasons to want to do it. The commercial justification for having people write open source software is that you get to train an entire industry of practitioners how to do their jobs. And because like, trust me, people did not do data work 10 years ago the way that they did today. And we're a big part of how that transition happened. But then they learn how to do it on your tools that you can then like sell commercial versions of. So there's a real commercial justification there. But the nice thing that's almost like a side benefit is that it's just like highly motivating as humans to feel like you're making a much bigger impact than your P&L might otherwise show.

Jon Krohn:	16:46	Cool. Yeah. It sounds like you're striking a great balance. You're certainly enjoying a lot of success. Something that we noticed in our research, something that you said a while ago, I don't have the exact quote in front of me right now, but we read about how over the years, over this decade of growing dbt labs into this large organization, you've had various opportunities over the years to be acquired and to basically experience a big wealth event for yourself and probably lots of other people in the business. How do you end up kind of deciding? It kind of reminds me of watching HBO's Silicon Valley TV show where you're always cheering for them to stay independent, to not be acquired, to make their next big leap on their own. And you continue to do that despite presumably the temptation of the capital inflow that could happen through an acquisition.
Tristan Handy:	17:50	You started off this conversation asking where I was calling in from. And honestly, I think that this has at least something to do with the answer to this question. Culturally, the East Coast and specifically Philadelphia does not have the kind of. Let's just say the culture inside of Philadelphia and the culture inside of San Francisco are like wildly different. Nobody cares how much money I have here beyond some relatively modest number. The housing prices are. For the prices you can get a reasonable home in San Francisco, you can buy a castle in the Philadelphia suburbs. And so I just don't need that much money in my life and generally optimizing for. And I started a consultancy. All of this is a surprising upside to me. The thing that I've chosen to optimize for over the last decade is creating an impact in the world and primarily an impact on users of our product and data analysts, most especially.
	18:59	And so when thinking about any type of corporate transactions, the thing that I've thought about first is how would this impact the users of DBT?

	19:15	The merger that you mentioned earlier with Fivetran was the first time that something hit the bar for me where I was like, "Oh, actually this would be good for users of DBT." Generally, I think that people don't set out with a specific intent to transform data. What they set out with an intent to do is like, "Hey, I have some data. It's sitting over there. I need to bring it over here and do some stuff with it." And so Fivetran is the pipes for that and we are the meaning creation part of that. Thousands and thousands of companies use these products together. I can't remember. I feel like our best estimate was something like. It was like over 10,000 companies used these products together already. And so it just made a tremendous amount of sense, whereas conversations we've had like this in the past might have made sense financially, but they didn't make sense for users.
Jon Krohn:	20:16	Well, we all thank you for continuing to be concerned first and foremost about this community. And I think it will serve you well in the long run. I think that that kind of drive to be as open source as possible, to be community minded, to continue to grow in a way that allows everyone, all of your users to benefit is going to, in the long run, be great for you. Probably financially as well, but you don't need to think about it as your primary goal. You mentioned there in your most recent response how DBT adds meaning. So that Fivetran is like pipes and DBT adds meaning. There's a term that came up a lot in our research for DBT labs, which is semantic layer. Do you want to explain how DBT acts as a semantic layer for your data?
Tristan Handy:	21:08	This is a topic that is particularly hot right now as analytics agents are very in view. The problem that the semantic layer solves is not a problem for small organizations. So if you imagine that you're a part of a, whatever, a 20 person company, a 50 person company, you probably don't need a semantic layer. But now imagine that you are Siemens. You're a global company.

You have 2,000 data engineers that are serving 300,000 employees globally. It is not possible to just know the answer to random questions that you might need to know the answer to without getting meetings together of people that you search for in your Outlook phone book. You get everyone together in a room and you say, "How should we be measuring this thing?" And there's a lot of conversation and everybody's got to figure it out and which table should we be using and all this stuff.

22:16 And literally that's how big companies have for the past whatever, 30 years, tried to answer questions like this. That's the process. And the semantic layer is tooling that allows those types of decisions to be made and then stored so that successive people when they ask those questions can confidently measure things in the same way twice. They don't have to reconvene the whole group. It is a technically complex problem area, but the problem that it solves is really an organizational problem. It is how do you scale knowledge to increasingly large groups of people? And that is honestly a tale as old as civilization. I mean, we don't have to go too deep on this, but as long as people have been organizing together to figure out how to do stuff, there's been this question of, well, how do we make sure that we know things and that everybody in this organization knows a consistent set of things?

23:25 And so the semantic layer is an attempt to do that. And it's particularly relevant today because absent these types of cues, how do you measure X thing? AI agents have to try to re-derive that for themselves at every turn. And oftentimes they do one of two things, or almost always, they do one of two things. One is that they, in re-deriving how do you measure something, they just take a tremendous number of tokens to do that. They just have to look into a lot of stuff and think a lot and that becomes slow and expensive. And then the other outcome is that they just get it wrong or they come up with an answer that may be kind of reasonable, but it's actually not the

way that your organization measures these things. So the semantic layer extends very, very nicely into a world of agents.

Jon Krohn: 24:15 Yeah, really cool. It seems like one of the key use cases, and if people aren't aware, that word semantic basically just means understanding, just means like the underlying meaning of some data. And so by having this common playing ground or this common lingua franca, this common agreement on what meaning is across data sets, across agents, there's efficiencies, especially across the large organizations that you were describing. They're like 200,000 person companies with 2,000 data engineers, that kind of thing.

Tristan Handy: 24:50 Yes, totally. And there are these successive layers of meaning where you start off with raw data, you go to modeled data, then you go to semantic layers, which are typically they help you understand how to join these tables together and how to measure certain metrics. But then you can even go one step further. And this is not an area of particular expertise for me, but it's a very interesting conversation happening in the industry, is ontologies. So ontologies are another layer of meaning making on top of data that are not just how do you measure a thing, which is typically how we think about the semantic layer. But it's also how do you model business processes? How do you model causation? And I think that most of us do not operate in an environment where it's appropriate to say we need ontologies for this because the world changes quickly and sometimes it's hard to keep up.

25:57 But in very specific high value domains, think like genetic research or like famously ontologies are employed a lot in defense. So I think about these four layers as kind of like the knowledge or meaning hierarchy.

Jon Krohn:	26:14	Yeah. The ontology thing is a bear and you definitely want to have a system that can be flexible or do it automatically, I suppose. For five years up until a couple of years ago, I was a co-founder and chief data scientist at a business that was working in HR tech. And in HR tech, you're constantly, there's lots of companies out there that make their whole bread and butter on creating these ontologies and updating ontologies so that you have like, okay, what are the skills that make up a data scientist? And then it's this constant, okay, so you have the skill ontology that you're needing to update on a constant basis like, oh, now there's PyTorch instead of just TensorFlow and we've got to add that in. And then whole new disciplines emerge. I might argue that data science is kind of fragmented into lots of these different kinds of specialized careers like AI engineer, LLM engineer, context engineer, all of these kinds of very specific roles.
	27:10	And it's somehow some ontologist's job to figure out how to understand a domain so well that they can manually create these relationships between entities in that ontology. Skills to jobs and jobs to different job categories and difficult in a constantly evolving world.
Tristan Handy:	27:33	Yes. Yes. And again, I'm not a super expert here, but you could imagine how a domain like genetic research could be really suitable for this type of work because it is very verifiable. It does not change. So yeah, maybe that makes a tremendous amount of sense. Trying to keep an ontology of the definition of a data scientist up to date
Jon Krohn:	28:02	Seems
Tristan Handy:	28:03	Like a very frivolous endeavor.
Jon Krohn:	28:05	For sure. It would be tough. It would be tough. Lots of people out there trying to do it. I would recommend you just use an LLM. That was kind of when we would have these conversations about, okay, I think it's time things

have become complex enough. Our businesses become big enough. We should start maintaining our own ontologies. I was like, that's a really bad idea. You're really not going to have a good time. And yeah, it's pretty amazing. The power of, I think it was about a century ago, Ludwig Wittgenstein, this famous Austrian philosopher, he had this theory that a given word is on average, the average of the meaning of the words around it. And that kind of idea is what allows initially word vectors, word to vector algorithms to work, and now large language models where. And yeah, it's pretty magical. And it means that we probably.

29:04 Yeah, I think for listeners out there, avoid ontologies wherever you can and just rely on the magic of modern large language models to handle it for you. Anyway, don't need to go on ontologies too long. There's another DBT, I don't know if you'd call it a product or a feature. There's something that you have called Fusion Engine that I want to make sure we don't miss talking about. So you've compared DBT's fusion engine to what TypeScript did for JavaScript and what React did for web development, promising to take SQL development from text manipulation toward compiler level understanding while aiming for a universal babble fish, to use a quote from you. So tell us about this Fusion engine and how it can change analytics engineering itself.

Tristan Handy: 29:53 In our quest to bring software engineered best practices to data practitioners, one of the thing Things that I think data practitioners, very much including myself, didn't even know to ask for was type safety in their language. So if most data people speak SQL, type safety is just not a thing that SQL has ever really provided. And in fact, it's not referring to SQL as a language is kind of a non-sequitur because there is no such thing as a, I don't believe that there is such a thing as a SQL independent of a specific database implementation. Every database has its own dialect of SQL and they are not the same. There's

a standards body that issues standards for a baseline level of functionality in SQL, but every implementation is pretty different, not just in its kind of high level, like what's the function signature of some standard function, but in its weird idiosyncratic implementation details such that you could run a syntactically valid SQL statement on database engine A and database engine B, and you could actually receive different answers.

31:16 And that is bad for the ecosystem. I mean, if you are old enough, you can think back on how miserable it was to develop web apps in the early 2000s when browser compatibility was basically nonexistent. It was a really thankless job going from this works in browser A to this works in all the other browsers. And so that ends up meaning that historically data people have just said like, "I'm developing for this database." And then you have Oracle as results, which is essentially a business that thrives because they've made it really hard to switch away from them, which is not good for anybody. What TypeSafety does is it says, "No, I really understand the language. I understand that the code being written here. I understand the operations that it's being asked to perform. And I understand it at such a deep level that if I needed to, I could actually rewrite this to be executed on a different engine and be able to prove that both of these queries would produce the exact same output." And this deep understanding of the actual code that's being written allows us to do a bunch of things that software engineers are used to, but data people are not used to.

32:41 I mean, as basic as when you are writing code in the VS code extension that we build now, we can automatically highlight, red squiggly underline, any syntax errors before you even run the code. And historically, the way that SQL developers got errors is that they would run it and then whatever the database said, you would then have to try to interpret that and track it down to what the specific error was. And now we'll just tell you and we'll tell you in real

time as you type instead of execute code against the database. So that's like one small improvement, but the overall trajectory of languages and computer science is to provide more and more functionality inside the language so that developers have more and more leverage to build amazing stuff in it.

Jon Krohn: 33:38 Yeah. Makes a huge amount of sense. And the DBT Labs Fusion engine does sound like a big step forward. It's really exciting. When you talk about being able to do more, I feel like I've got to get back to the agents that you were talking about maybe 10, 15 minutes ago in this conversation, because there's still so much more for us to talk about with respect to DBT Labs and the future of the world, which is agentic. To me, there's no question if people are using CloudCode, you're using OpenAI Codex, and you really should be because they're so magical. It's mind blowing. And on an almost daily basis, I think of new ways that I'm like, "I wonder if it could do this really hard thing." And you're like, "Boom, Fable five does it." At the time of recording. Yeah, go ahead.

Tristan Handy: 34:28 This is a complete side point, but I have many of those experiences. We've all had many of those experiences, but I always still enjoy it when I ask it to do something and it just completely falls in its face. So yesterday I've really gotten into cycling and I was like, "Well, maybe I could ask it to make me what's called a GPX file, like a route that you can import into by computer." And I asked it like, "Hey, make me a 20 mile route starts at my house, ends at my house in this neighborhood." And it thought about this for a good solid 15 minutes. It came back to me, gave me a GPX file. I imported it into my bike computer. And it had me riding through people's backyards. And I'm just like, "You just completely failed to do this thing." So yes, the overloads are very good at some things and not that good at others.

Jon Krohn:	35:26	For sure. Yeah. I would have thought it could probably figure something like that out. So that is a. Yeah. It is surprising where it falls down. Yeah, it used to feel because there were so many more places where it would fall down and it was so easy to. A few years ago, kind of GPT-3 level intelligence, when we were only using LLMs to replace us on tasks that took a few seconds long, it was easy to experiment and find out, okay, yeah, this doesn't work, that doesn't work. We're doing pretty well over here. It's great at this. But with GPT-4 kind of level intelligence, and then we're talking about many minute long tasks, and it's quite competent that the seconds longest tasks becomes harder to test. And now that we're kind of in this era of, okay, sophisticated agent harnesses, lots of double checking, gigantic models with lots of different mixture of experts, nodes being pulled in for different tasks.
	36:25	So we're getting these really deep levels of expertise. It allows us to be now replacing ourselves on tasks that would take us hours in some cases. And so it becomes harder to test and experiment and find what are these kinds of hour long tasks.That's a good one. I wonder how soon that'll be figured out.
Tristan Handy:	36:45	The question I think behind your question is how does DNI spend all my time thinking about this, but how does DBT fit into an agentic future? I remember somebody asking me at an onboarding session back in something like 2018 or something, and I was talking about how DBT was really born out of the paradigm shift towards the cloud. And this person asked me, "What's a paradigm shift that could make DBT less relevant?" I mean, I literally don't know the answer to that because I don't know how the future will unfold. But essentially every founder in the world, when ChatGPT first came out, had to ask the question, "Is my product more or less relevant in an AI dominated world?" I think the answer for DBT is

actually a little bit in the middle. We are not a product that because of AI, you're going to stop using DBT.

37:48 That's not how that works. But at the same time, it's not yet clear that because of AI, you're going to use a lot more of DBT. It's funny, our business has been very, very steadily growing over the last, whatever, six years. And you don't actually see the rise of AI show up in that yet. There's a couple of early indicators, but yet. And the reason is that we work really well in this future for a couple of reasons. One is that DBT is code first. And so coding agents are quite good at writing DBT models. And so there's more than ever of them. And especially, you mentioned the Fusion engine before. Fusion is very good paired with agentic coding because it leads to these really tight iteration loops where the agent can test its own code very, very quickly and cheaply. And then on the other end with the semantic layer, it is a really important piece of infrastructural input into analytics agents where people throughout an organization can ask questions and feel good that they are getting trustworthy answers back.

38:55 But we're still in the early stages of deploying analytics agents through an enterprise. So I'm excited about that S-curve coming up.

Jon Krohn: 39:06 Yeah. I think they're going to be creating a lot more data, a lot more data pipelines. It seems to me like LLMs agents are a good thing for dbt labs. I think so

Tristan Handy: 39:21 Too. Honestly, one of the things that we've been doing for as long as we've had a commercial business is we've been replacing kind of old school ETL products, whatever. There's companies that have been around for 30 years that are deep in the bowels of Fortune 500 companies. And agents have done two things here. One is that it's made it very clear that these old school products are not going to meet modern needs, but then also it really helps with the migration. And so it has never been easier to

migrate from some legacy product to a modern thing with a good long running agent.

Jon Krohn: 40:06 Something that you wrote about recently in a LinkedIn post related to talking about agents and using tools like CloudCode is how you describe a 12 kilobyte skill file, very specific number, but relatively not tiny, but relatively small. It could encode hundreds of hours of collective human experience and run a DBT migration flawlessly. So do you want to tell us a bit just in case listeners aren't aware of skill.md files and cloud code or kind of related skill files and other agents? And so yeah, what those are and how they're such a powerful tool for a specific task like a DBT migration.

Tristan Handy: 40:47 I mentioned before that one of the things that we've been doing over the past 10 years is essentially teaching a group of professionals how to do their work in a different way. We called this analytics engineering. We built tooling around it. We also built coursework around it. We have certifications. We have online learning. In order to work in this way, you need to know some stuff. And it's not an infinite amount of stuff. You can generally go, if you know SQL and you know data, you can become pretty expert at DBT if you spend about two solid weeks focused on it. But skills are this magic way. I mean, they to me are the closest thing we have yet seen to the matrix where you plug that giant wire into the back of your skull and you download new skills into your brain. Here, you're downloading skills into your agent's brain.

41:50 So we have authored a set of DBT related skills that are essentially like the knowledge that we were training humans on via all of our online courses and certifications and everything. And we've packaged all that up and we've distributed it to agents. And it turns out that that just works really, really well. Now I'm sure that we're going to continue to learn and evolve that skill, but it is right out

of the gate. It was very impressive and is becoming pretty widely used.

Jon Krohn: 42:26 Cool. And this is probably kind of a dumb, simple question, but what is a DBT migration?

Tristan Handy: 42:32 A DBT migration is imagine that you were using some other product or maybe you just wrote a bunch of stored procedures or you had a bunch of spark code or whatever. And all of that was doing data modeling in your organization. In a small organization, that might be a thousand tables. But inside of like a big bank, I mean, that could be 10 to 50,000 tables that you're talking about here. And sometimes the people that wrote all of this code, not only are they not at the organization anymore, but some of them are retired. It can feel very high risk and certainly very timely, like time intensive to take all of that code and migrate it into DBT. And so as a result, some companies choose not to do it. Other companies who do do it, they hire consultants that take a year and multiple million dollars to do that type of work.

43:39 And now you can do big, big DBT migrations in six weeks and for tens of thousands of dollars instead of millions of dollars.

Jon Krohn: 43:48 Really cool. Thank you agents. Thank you, DBT. Thank you, Tristan, for building such a powerful open source product and developing this analytics engineering community around it. I promised I would let you go early. So despite it having been such an interesting episode, we should start wrapping up. So my penultimate question that I always ask my guests is if they have a book recommendation for us. You got anything?

Tristan Handy: 44:13 I am a big audiobook reader. I don't know. I've been an Audible subscriber for like 20 years now. My most recent book I thought was really fascinating. It's called Musashi. It is about Miamoto Musashi, who was, I think, the most

famous Japanese samurai to ever live. And it almost feels like stories of King Arthur. If you've ever read the Once in Future King, it is almost like the Japanese version of that. And so I found it really. The story was interesting, but it was more interesting. What are the stories that it tells itself? So I found it to be a really interesting read.

Jon Krohn: 45:00 That is really interesting. I don't spend enough time thinking about that great culture of samurais, which as a kid was somehow so fascinating. And so I looked up quickly here because I was like, you know what? I've never known the name of a single samurai. This is my first Miamoto Musashi. And yeah, he looks very impressive. He

Tristan Handy: 45:25 Wrote a book I think called The Five Rings. It's very digestible. If you want to become a master in the art of the two sword technique, read his book. It's like less than a hundred pages.

Jon Krohn: 45:35 There you go. And then you two, like him, can have an undefeated record in 62 duels.

Tristan Handy: 45:42 Yeah, exactly.

Jon Krohn: 45:43 Nice. Final thing before we wrap up, Tristan, is how can people follow you after this episode or follow DBT Labs?

Tristan Handy: 45:51 I am a podcaster myself. I write a newsletter. You can find all of that at the Analytics Engineering Roundup and look forward to connecting with you there.

Jon Krohn: 46:04 Nice. Sounds great. Thank you so much for taking the time out of your super busy schedule with us. We really appreciate it, Tristan. And hope to catch you again in some years to come and see how DBT Labs is coming along.

Tristan Handy: 46:13 Thanks a lot. This has been a lot of fun.

Jon Krohn:	46:15	Extra informative episode today with an inspiring data entrepreneur. In today's episode, Tristan Handy detailed how dbt was born from a study of about a hundred companies moving to the modern data stack, why he chose SQL over Spark back in 2016 so that data analysts, those rare people who span both quantitative skills and business domain knowledge could start simple and layer on complexity only as they need it. He talked about how the semantic layer stores and organizations agreed upon metric definitions so nobody has to reconvene a room full of people to redecide how to measure something and how skill.md files can package a decade of DBT training courses and certifications into a few kilobytes an agent can absorb. So migrations that once took consultants a year and millions of dollars can now be done in six weeks for tens of thousands. As always, you can get all the show notes, including the transcript for this episode, the video recording, any materials mentioned on the show, the URLs for Tristan's social media profiles, as well as my own at superdatascience.com/1021.
	47:19	Yes. Superdatascience.com/1021 for episode number 1021. Thanks of course to everyone on the SuperDataScience podcast team, our podcast manager, Sonja Brajovic, media editor, Mario Pombo, our partnerships team Natalie Ziajski, our researcher, Serg Masís and our founder Kirill Eremenko. Thanks to all of them for producing another excellent episode for us today for enabling that super team to create this free podcast for you. We are deeply grateful to our sponsors. You can support this show by checking out our sponsor's links in the show notes. And if you'd ever like to sponsor an episode yourself, you can get the details on how by making your way to jonkrohn.com/podcast. Otherwise, share this episode with your favorite DBT fan. Review the episode wherever you listen to podcast episodes or on YouTube. If you leave a text review on Apple Podcasts, that is especially helpful for us and I'll read it on air at some point.

48:18 Subscribe obviously if you're not already a subscriber, but most importantly, I hope you'll just keep on tuning in. I'm so grateful to have you listening and I hope I can continue to make episodes you love for years and years to come. Until next time, keep on rocking it out there and I'm looking forward to enjoying another round of the SuperdataScience podcast with you very soon.