EPISODE 1310

[INTRODUCTION]

[00:00:01] JM: Enterprise data warehouses store all company data in a single place to be accessed, queried and analyzed. They're essential for business operations because they support managing data from multiple sources, providing context, and they have built-in analytics tools. While keeping a single source of truth is important, easily moving data from the warehouse to other applications is invaluable. Tejas Manojar is the CEO of hightouch.io, which is an enterprise ETL product company that I have invested in. It's a great company, and they provide tools that easily move data from your warehouse to important business tools like Salesforce, Airflow, Tableau, and more. Hightouch uses SQL to move and transform your data. It tracks every row to avoid moving data that hasn't changed. Failed rows are retried. All changes to rows are logged. And it's quite a powerful product.

In this episode, we talk about reverse ETL, managing data across multiple systems, and how Hightouch helps companies operationalize their data warehouse. There are obviously a number of new companies in this space. There's Airbyte, there's Census, there's other things, there's Hightouch. And it's a competitive landscape, but it's a gigantic and growing landscape. All of these companies are going to be successful. So what is Tejas' competitive positioning? What are the technical problems? There's a lot of meat to this episode.

[INTERVIEW]

[00:01:24] JM: Tejas, welcome to the show.

[00:01:26] TM: Hey, thanks for having me.

[00:01:28] JM: So, okay, we met – What was it? Three years ago at Joseph Jacks' house for dinner, if I remember correctly. Is that right?

[00:01:35] TM: That's right.

[00:01:36] JM: Okay. So the context there, Joseph is the head of OSS capital, which makes investments around open source software. And I remember meeting you that night. Basically, the context was here's a guy who's pretty young and pretty smart from Segment. He's got some ideas. We're just having dinner, blah, blah, blah, blah. Fast forward two or three years later, I put in touch with you by Lenny Pruss, I believe, from Amplify. I think that's how we got connected, right?

[00:02:02] TM: I think it was either Lenny or Joseph. I'm not sure actually.

[00:02:05] JM: Okay. Right. So anyway, you're working on Hightouch. And I immediately understood the worth of what you were working on. So I put a small investment in. So Hightouch is quite powerful. And I want to get into what that is. But I think my first question is actually around this category. So this category that has been termed reverse ETL, I believe. That's the category, right?

[00:02:28] TM: Yeah, that's a term we've been pushing.

[00:02:30] JM: Right. So was that coined by Astasia Myers?

[00:02:33] TM: I wouldn't say it was coined by a Astasia Myers. So actually, I think it's a term that our customers came up with a number of times. So we would always be on calls explaining what we do, how we move data from the data warehouse into different operational systems around the company. And a lot of customers or prospects would just be like, "Oh, so is that basically the reverse of Fivetran? Or is that the reverse of Stitch? Or is that the reverse of an ETL process?" And then the term reverse ETL just started picking up in certain like early adopter user communities online and things like that. We started using it a bit. But I think Astasia from Redpoint was the first one to actually publish online, like a blog post on reverse ETL. And we quickly followed.

[00:03:16] JM: Right. Yeah, this is a meaningful article from Atasia's Medium blog. It's reverse ETL a primer, or primer. And the idea is basically ETL is mediated by Fivetran, or Fivetran like companies, such as Airbyte. ETL essentially means take your data from some location, move it around, munge it and then put it into a data warehouse, right? That's ETL. So you get your data

into your data warehouse, and then you can do fun things with it, like query it, do all the things that the data warehouse platform has become. Does ETL into spark? That's a thing also, right?

[00:03:57] TM: Not exactly. So Spark isn't exactly like a data storage mechanism as well, like a traditional data warehouse is. Spark is more a system for processing large amounts of data in-memory. And it can tap into different data sources to pull out data in-memory. So ETL-ing data into Spark isn't really a thing. You would more ETL data into something like S3 and then use a Spark S3 connector to pull the data out of there into Spark itself.

But yeah, you're exactly right. I mean, ETL is basically the process of bringing data into a centralized repository, typically a data warehouse, but it could also be a data lake like S3, HDFS, anything like that. And what we're doing with reverse ETL is the exact opposite. So taking data from those repositories, particularly models that you built on top of those repositories, and syncing those data models into different systems where teams actually live on everyday basis.

[00:04:52] JM: So the ETL connection infrastructure, the Fivetrans, the Airbytes, the Matallions, those are primarily for getting data from some source, either S3, or Stripe, or Salesforce. Moving that data into a data warehouse, right? So primarily, you might have the source being Salesforce, and the sync being either Redshift or Snowflake. Is that accurate?

[00:05:25] TM: Yeah, that's exactly right.

[00:05:26] JM: Okay. And so for the purpose of this conversation, I guess we'll kind of ignore Spark, I guess. Can we ignore it for now or –

[00:05:32] TM: For the most part, Spark and Databricks, as a company, are getting a bit into the data warehousing space with this new term they've been calling a data lake house, which to consider mostly marketing, but some sort of crossover between a data warehouse and a data lake. But if you were to load data into a system you want to read from Spark, it could just be into Redshift, into S3, or whatever is an input into Spark. But yeah, the terms like all get confused when it gets into engineering territory. That's what we noticed.

[00:06:00] JM: Yeah. I got to ask you, because this is such a burning question for me. I can't figure out the Spark versus Snowflake use cases. I can't draw the Venn diagram in my head. Can you do that for me?

[00:06:12] TM: Yeah. So one thing to note is that Spark and like technologies came much before cloud data warehouses. So obviously, data warehouses existed for a very long time. And those technologies were basically built to scale to be able to store tons of information and select data from there in arbitrary ways. Spark, however, was built as basically a framework, you can think of it, for doing complex computation using clusters of servers with a lot of memory. Typically, people run Spark to do like high-volume computations inside of memory, where they have these computations built in advance, and it's like run in a more optimized way across a cluster of machines. Whereas data warehouses were built as a platform where you can dump a bunch of data into them, and then kind of interactively query it in ad hoc ways.

So a really good Spark job could be like, say, more optimized than what you might run an ad hoc query instead of a data warehouse. Since data warehouses are really built for ad hoc queries, where you might query the data in all sorts of different ways, and you don't really get to index the data in the way you choose all the time and stuff like that. Whereas S spark is built for more doing a specific type of computation over and over, and it's just a framework for processing high amounts of data, particularly in memory.

I would say what's happening is that, just in general, in the data engineering ecosystem, people are expecting. And this is probably true just in the developer ecosystem as a whole. People are expecting more and more abstraction to be served to them when it comes to dealing with high volumes of data. So I'd say Spark is obviously a really robust abstraction. But it's still meant for a very particular task, which is processing high amounts of data in-memory in particular. But what's happening in the data warehousing space is people are saying, "When I want to query high amounts of data, let me just associate that with a data warehouse. Let me just give it a SQL query. And let's let the data warehouse itself figure out the best way to do the query here." So I'd say cloud data warehouses like Snowflake are just more abstract platforms that hide away the implementation details and can just take a SQL query and do the kind of optimization under the hood and take different approaches for the queries and stuff like that, whereas you

reach to Spark when you actually have like a particular challenge that you want to solve on a recurring basis with an in-memory computation framework.

[00:08:37] JM: We're getting pretty far afield from reverse ETL. But I'm going to keep going, because we got 45 minutes. I'm going to take a pretty long detour here. Snowflake versus Databricks on a deeper level. Snowflake is the closed source tiered storage engine with a data warehouse API. And really, what Snowflake is, is a tiered storage engine. Agree or disagree?

[00:09:09] TM: I think that's mostly true. For the most part, that's true. I think what's really valuable about Snowflake is that they separated the compute layer from the storage layer and made a really good UX around –

[00:09:22] JM: Why do people always say that? What does that mean? They separate the compute and the storage.

[00:09:27] TM: What that really means is that you can dump as much data as you want into Snowflake and it goes into a shared set of storage resources. You can imagine this is something very similar to S3, but they actually bring the data into their own servers, and they charge you cost that's actually very, very similar or at par with S3 for storing data. Then what happens is when you submit a query to Snowflake, Snowflake only charges you for the resources you use, and they give you a really granular control of how much resources you want to use. So you can say, “I want to run my queries with this Snowflake instance type,” and you actually pay per second on the amount of time the query actually takes from a compute perspective to execute, and nothing more. And that's a really flexible pricing model, because it means I can store immense amount of data inside of Snowflake and pay almost nothing for it, basically paying the same cost of storing that data in some sort of medium storage like S3. But then when I want to run a crazy compute-intensive query to it, I can still submit it to Snowflake and just pay for the seconds that that query actually runs.

Whereas in a system like Redshift, the first generation of cloud data warehouses that actually created the need for things like Fivetran, this is Amazon Redshift for that matter, you have to provision your instance of a data warehouse just for you and it's running around the clock. If you want to run an expensive query tonight, you need to basically spin up a Redshift, keep it

running, and spin it up with certain resources. All the storage is just for your instance of Redshift. It's basically isolated to you entirely. And just keep it up even just to run that query that's more intensive at night than in the daytime. Whereas Snowflake is really an elastic resource where you're only paying for what you consume.

[00:11:17] JM: Snowflake is Oracle 2.0. Agree or disagree?

[00:11:24] TM: There are some similarities from a company perspective. It is very difficult. I think notably difficult to get your data out of Snowflake, which is becoming a complaint of a lot of customers that we've been talking to. It also does grow to be very expensive at scale sometimes. That being said, I think they really made a head start on the technology and that the product they provide is just so impressive compared to the incumbents. I guess the same was true of a lot of Oracle technology.

[00:11:52] JM: Databricks is the open core alternative to Snowflake. Agree or disagree?

[00:11:59] TM: That's the part I'm not sure I can totally agree with. From my understanding, obviously, Spark is open source. Obviously, standard is open source. But my understanding is Databricks itself has a lot and a lot of closed source software that you can't really get close to emulating with the open source versions. So with that said, I'm not sure that that's actually true from what we've heard.

[00:12:26] JM: Is that a worthwhile line of reasoning though, because that’s how I’m starting to think about this? I'm just trying to like understand what the heck's going on here.

[00:12:33] TM: I think from what we can tell in the market, the better way to think about it is that Snowflake produce a net new technology, which was the cloud data warehouse, that performed incredibly well, and started to perform so well, that it started to take over use cases of data lakes, Spark, and things like that. Not just use cases of Teradata, Vertica, Redshift and older data warehouses. And part of that was probably actually unintentional. But as a result, I think Databricks as a company has realized that they need to play in that space. Data lakes are starting to be considered legacy into a lot of companies due to how good of a solution Snowflake is, and thoughts they need to figure out how to play in the data warehouse space to

some extent. Figure out how they kind of fit into that market. And as a result, their response is things like data lake house, for example.

I think there are some merits to those technologies, but like every developer company, they're kind of shifting their existing foundation to start following the modern trend just because they have to from a market perspective.

[00:13:43] JM: Alright. I see it a little bit differently. I respect the position you just presented. I actually just have one tweak on it. I don't know if you recall the early days of Databricks, but the Spark wasn't really taken seriously to the extent that it should have been. Do you remember that?

[00:13:58] TM: I wouldn't say I was following it too closely back then, to be honest.

[00:14:01] JM: So I heard that the story was essentially. So it came out of the RadLab, I think, or like now the RISELab. But back then I think it was the RadLab. This is the same lab that spun out Mesos and Tachyon. Tachyon now – What is Tachyon called now? Whatever. Next generation Tachyon or something. So it spun out these amazing projects. And Spark was there, but they were having trouble finding the adoption for it that was necessary, because people were like, “Well, I got a Hadoop MapReduce. What more do I need? What was the Spark thing? I don’t work at the Spark thing? Like I don't want to do that.” I think that's what was happening. And Databricks got funding from Andreessen eventually, because Andreessen was like, “Well, this is going to work eventually,” and then they just funded it. And it did work eventually. They figured out you know that – Who's the guy that's in CEO role now at Databricks?

[00:14:53] TM: Ali –

[00:14:54] JM: Ali Ghodsi. So they found out Ali Ghodsi was the best CEO that Andreessen Horowitz had ever seen. He’s been in charge of Databricks for a while. And so the company is revolutionary. So like what happens when you have an infrastructure company that's run by the most visionary, or at least most talented CEO in the Andreessen Horowitz portfolio? What happens is that you get a new kind of company. So like Databricks actually resists analogy. In reality, it resists all analogy, because it's an interactive data science platform and that has a

lower level open core infrastructure. And if you think about what the data lake house really represents, is it encompasses everything you need to have a highly optimized data infrastructure on an open level. So it's essentially the Linux of data infrastructure. You got an end-to-end solution, or the Kubernetes of data infrastructure, whatever you want to call it. It's the de facto open solution. So you have the whole tiered storage configurability open to you. Am I mistaken? Or is that what they're doing?

[00:15:47] TM: That's probably correct to a certain extent. But my understanding is the standard isn't as open as advertised when it comes to Databricks, but I think to some extent. Like they're making a lot of progress when it comes to like how should people implement data sharing. I saw that proposal recently. That was really interesting. How people should implement delta lakes. Like a lot of these open standards are definitely pushing forward progress in the industry.

I think one thing that I've heard repeatedly from customers though is just the UX of Snowflake, error messages. How easy it is to get spun up, how easy it is to provision, how easy it is to get data into it, etc., is just kind of unpared in the market even between like BigQuery, Redshift, Snowflake. But even when it comes to like Spark and Databricks, it's just kind of unpar. And I think that is actually like a huge reason why they've taken off. And I think it'll be interesting to see if Databricks is able to do that on top of their existing foundation, which is Spark.

And I think, from my perspective, having administered Spark before, it's a really great technology at being able to do things that would be very complicated to figure out without Spark. But at the same time, I can say the user experiences is anything near just submitting a SQL query to Snowflake or BigQuery for that matter, from what we've seen though.

[00:17:08] JM: Anyway, not to go deeper on a tangent. How much you know about Ion Stoica?

[00:17:15] TM: Not that much. I've actually sat in on one of his lectures once. I snuck into a class in Berkley.

[00:17:20] JM: What was the lecture?

[00:17:22] TM: It was actually on operating systems, I think. Computer architecture and operating systems.

[00:17:25] JM: Wow! What does Ion Stoica think of operating systems? Do you remember?

[00:17:29] TM: I don't know that far. I sat on the lecture when I was deciding whether I wanted to go to Berkeley.

[00:17:35] JM: I think that I realized in retrospect is how amazing it is to take lectures from legendary professors. So UT Austin is like a pretty good computer science school. It's like top 10. It's not spectacular, but it's top 10. That's where I went. And I took a class from this legendary distributed systems guy. It was the hardest class I ever took. I almost failed it and almost like would have had to take an extra year. But anyway, it was just quite remarkable in retrospect. And I didn't really understand how legendary this guy, at least he was. Anyway, sorry. I'm really getting off track here. But I want to say about Ion Stoica. So Ion Stoica is one of the founding members of Databricks. I guess he runs the RISELab now, I believe. He ran the RadLab. And basically, these labs just spin off commercial technology. Like they make it look easy. You know what I'm talking about? They make it look easy. They're just like, "Yeah, we're going to do like the next generation whatever. Let's just do it."

[00:18:24] TM: That particular lab in Berkeley, or set of labs, has produced a lot of open core companies.

[00:18:29] JM: Yeah. And they have Dave Patterson just hanging out. It's just like Dave Patterson hangs out, talks about what he thinks the future is.

[00:18:35] TM: Scott Shenker is also one of the other professors that's used a lot there in there, Nicira. Actually, I interned in his lab for some time, Netsis. And then actually we hired someone who I met in the lab back there, like four years ago. But at the time, yeah, I sat in on a few lectures as well.

[00:18:52] JM: I mean, it's just an incredible center of innovation. It's kind of its own thing. Like it's like what people say about the like nuclear power research labs. I think Los Alamos or something. Like it's just this weird organization that produces high value, right?

[00:19:06] TM: Fair enough. I think it's like the amount of companies they've outputted that are successful is unusually high for that lab, right?

[00:19:13] JM: Whenever I want to do new shows, there's a few places that I go to just – Because sourcing content is kind of hard for this show. You do five shows a week, 50 weeks a year, 250 shows. Like what are the 250 best topics in software per year? Obviously, Tejas is in there. But like where else do you go? And one of the answers is the RISELab. I just like scroll through their GitHub. This is why people should always make your email address public on GitHub, is I'm looking for you.

[00:19:38] TM: That is hilarious. And honestly that that is a tactic I use to find people's emails pretty regularly, even if it's not on GitHub. I don't know if you know, but you can just look at the GitHub commits, like the Git commits. They actually have an email usually. Oh, like generally people type their name and email into the Git author config. It's like an incredible way to find people's email addresses.

[00:19:59] JM: Oh my God! That's a lot of value. That's a lot of value. Thank you, sir.

[00:20:04] TM: I do that all the time.

[00:20:05] JM: Than you, sir.

[00:20:05] TM: Like a lot of people have their spam email on their GitHub profile, but then they have their real email in Git.

[00:20:11] JM: Thank you, sir. Okay, all right. Look, we're going to get to reverse ETL. By the way, did I tell you I'm doing I'm doing a new company? This is useful because I'm recruiting people now.

[00:20:19] TM: Yeah, he told me.

[00:20:19] JM: Okay. Anyway, are you a Magic player?

[00:20:21] TM: I am not a Magic player, but I have a number of friends that play Magic.

[00:20:23] JM: You're not a Magic player. Okay. Alright. Cool. It's going to be a great game. You're actually going to love it. It's going to make you into a digital card game player. You're love it so much. It's a game that embodies the world of software. It's basically like you get to play as a software person in this game anyway. Anyway. I'm sorry. This is your show.

[00:20:39] TM: No, no. I think that these tangents are actually entertaining.

[00:20:43] JM: You know what happened? I'm totally breaking the fourth wall here. I've been really thinking about – Do you listen to the All-In podcast?

[00:20:48] TM: I don't.

[00:20:49] JM: You don't? Okay.

[00:20:50] TM: I only listen to your podcast, Jeff.

[00:20:52] JM: You're kidding me.

[00:20:54] TM: I'm completely serious. I mean, especially without commuting, I don't really have that much time to listen to a ton of podcasts.

[00:21:01] JM: So like how many episodes per week like out of five do you listen to?

[00:21:05] TM: Honestly, probably one, two max.

[00:21:07] JM: One to two max. Okay. Like be serious. Is it like one – Do you listen to one in nine episodes?

[00:21:11] TM: I would say probably one every two weeks on average.

[00:21:14] JM: Okay, one every two weeks. And is it the full show? Or like first 20 minutes? Does it vary?

[00:21:20] TM: I would say on average, it's the first 20 to 30 minutes. Yeah, I rarely listen to the full show.

[00:21:25] JM: Yeah. I think it's too long. It's either it's too long, or we need to do more of this kind of thing.

[00:21:30] TM: I think if you have like a one hour commute or like a 45-minute commute, which again –

[00:21:35] JM: Which nobody has anymore.

[00:21:37] TM: That really sucks for you because –

[00:21:38] JM: Nobody has that anymore. God! That another topic.

[00:21:41] TM: If you have that, it makes a lot of sense. But if you don't have that, I think it's – I just don't know where people find the time. Maybe people shower for a really long time, but –

[00:21:50] JM: I wear Bluetooth headphones in the shower at this point. Showers are too boring.

[00:21:54] TM: How long are your showers?

[00:21:56] JM: Ah, you know, 5 to 10 minutes probably?

[00:21:59] TM: That's definitely not enough for a Software Engineering Daily episode.

[00:22:02] JM: Yeah. But like if I go from the gym, to taking a shower, I just continue to listen to whatever podcast – Anyway, this is really getting golf course. But on the note of like people commuting, like I hear Amazon, and Apple, and Facebook and stuff are trying to get people go back to the office. That's actually happening? Like are they actually trying to like go back through the one way door of remote work? Like how are these organizations actually expecting their employees to respond to an edict to return to the office?

[00:22:32] TM: I think particularly in a large, large company, it's hard. It's extremely hard to make changes like this. Yeah.

[00:22:39] JM: Seems impossible to me. I don't know. So on the All-In podcast actually. So the first thing about the All-In podcast is it's the best podcast. So actually, just stop listening to my podcast and start listening to All-In. I guarantee you will have a better time. I actually have to figure out how to innovate, because otherwise I'm just going to be worse than them. So that's part of what I'm trying to think through. But anyway, one thing that they discussed is they talked about what is the remote work policy asymptoting towards. And they basically thought that the way that it's going to work is all the best employees are going to stay home, right? Because if you have the choice – I mean, I don't know about you, but I don't ever want to go to office again, I don't think. You're going to stay home, right? Like if you have the choice, you'll stay home. I'm asking you.

[00:23:17] TM: I honestly do enjoy the social environment of the office. Like I'm just thinking back to when I moved to the Bay Area in a joint segment. A large part of the fun was just hanging out with people in the office. Like, honestly, I did a lot of things like hanging out with people across different teams and departments. I met a lot of people in sales, and marketing, and support. And I just don't know if I would have taken the time to do that on Zoom, especially with all this like Zoom fatigue, which is very, very real, in my opinion. I don't know. I'm still a fan and proponent of the office, I think. I think after I've been working, sitting in my own room for a long time, I do also really agree that that brings a level of productivity that's kind of unprecedented when it's just me and laptop and a cup of coffee.

[00:24:05] JM: Exactly. And you're not thinking about this from first principles, right? Because if you can select who you hang out with on a given day, and you're willing to travel, like you are

with the commute to Segment, then you should just be like going for a walk with me, or hanging out with JJ. Like why would you –

[00:24:19] TM: What about the serendipity of just like meeting new people in the office? Like do you not think that's –

[00:24:24] JM: Yeah. I mean, we need to rethink culture, basically, right? Like we need to be doing more meet-up like things. We need to be doing more group dinners. Like I can't wait to get back to San Francisco so we can do a group dinner. It's going to be awesome. But like we need to do more of that, right? Like we don't need to have people commuting to an office. It doesn't make any sense.

[00:24:42] TM: Yeah. I mean, I'm a fan of an optional office. And I think people will still want to congregate. Like young people still want to congregate in cities, which they have a bunch of other like-minded people in, whether it's San Francisco, or New York, or any other city. But I do think even in ideal world, like I would just want to go in like couple days a week. Just honestly, like meet people, have lunch with like the wider team. Like I think that sort of stuff is valuable, especially if you're looking to just expand your horizon tremendously **[inaudible 00:25:11]** and stuff like that.

[00:25:13] JM: But then again – Okay. So then what is the HQ? So the HQ should basically be like a place to go to hang out. It shouldn't actually be a place with desks, right? It should basically look like a shopping mall.

[00:25:23] TM: I also think for certain like group meetings, like whiteboarding, like a lot of these things are just – At least for me, personally, they just get boring online. Maybe that's the problem. Maybe the tooling needs to improve radically. And we all need VR headsets at home or something like that. But a lot of those sort of interactions get very boring online. And I think coming in-person for some amount of time actually allows me to work harder and longer in its own way.

I think a majority of work, you should be focused. If you're not, you should turn that into focus work somehow. And you'd probably be doing a majority of work independently, thinking,

working, etc. And for me, at least, it's unprecedented the productivity I can get, just like being at home for that. But at the same time, I think from like a zoomed-out perspective, going to the office and just being surrounded by a bunch of people, and that sort of social interaction, and like work social interaction does provide me like long term fuel.

[00:26:23] JM: Yeah. Okay. All right. Let's just get into reserved ETL. I mean, I could talk to about for a long time. But reverse ETL, so where you're at strategically? Let's just kind of boil this down. So you're the reverse ETL company that is not Census is how I would describe you. Actually, that's a bad pitch. I'm an investor. I should give you a better pitch. Sorry.

[00:26:44] TM: We are the reverse ETL company that actually has a self-service. I think that's the best way to describe us, the reverse ETL company for developers, by developers.

[00:26:55] JM: All right. When I was talking to Census, they actually convinced me that they are like the Zapier for customer data syncing. I mean, okay, so basically what use case, to tee you up, is you're actually getting data out of your data warehouse and into something like Intercom, or Salesforce, or whatever. So we talked about earlier about how Fivetran takes it out of these random sources and then puts it into your data warehouse. This takes out your data warehouse. It puts it back into the sources. Am I correct?

[00:27:22] TM: Exactly. Essentially, what we're doing is saying, “Look, every company has a data warehouse now. And that data warehouse has all the data and the key data models across the company.” Like the gravity of data is moving towards the warehouse. But at the same time, people are just using the sync for reporting and like ad hoc analysis. When there's so many problems scattered across an organization that are like, “I don't have my data on Salesforce. I don't have an updated data in the spreadsheet. I can't cut my user base by the right data in this marketing tool.” And essentially, what I’m just saying is like stop building pipes to move data into the system. Stop connecting one system to another. You have a database over there with all the data. Just put a SQL query in our platform and say how you want it to look in other systems and we'll figure out how to get it into there.

And I think a Zapier for customer data is a pretty good way to think about it at a high level and like a nice analogy. But one thing it really gets wrong is that a lot of these existing like

integration platforms today like Zapier, Tray, MuleSoft, Workato, like whatever platform you're talking about, they're imperative. You have to tell them like exactly what to do. You're like, "For every row in this database or Google Sheet, go look it up in Salesforce. And if you find it, go see if there's an account associated with it. And if there isn't, then associate one with it. And if there is, then update the contact properties." And you write this like long tree of steps, which gets like very hard to manage. It just becomes like code in a visual UI without functions, without tests, without anything. And what Hightouch translates that to is like a very declarative platform purpose-built for data syncing. And I think that that's actually even the more important part than the data warehouse. Like the beauty of Hightouch is that if you have one schema on the left side, the warehouse, and you have a destination system on the right side, Salesforce or whatever it is, and you just say how you want data from one schema to look in another tool. And we figure out how to actually do it when it comes to like making lookups, batching stuff up, handling foreign keys in the downstream system, and all that sort of thing. And I think that's actually what's really powerful. I think if you could even connect point to point systems in a declarative way, it would actually be powerful in its own way as well.

[00:29:40] JM: This problem space is so deep. And as I try to map it in my head, I think it's kind of like if you think about Cloudera versus Hortonworks. So Cloudera versus Hortonworks was compared to today a terrible time to build software. Cloudera and Hortonworks both are great companies. They both are very successful outcomes for founders and investors. Hightouch versus Census is actually that, but playing at a higher level of abstraction. Agree or disagree?

[00:30:11] TM: I think that's true to some extent. One thing that's very interesting, and you can see it in pretty much every SaaS market today is there's always multiple companies that look pretty similar in the early stage, but slight differences make massive impact. And what we see as something to really capitalize on in this market, especially while it's early, is making a very like self-service and developer friendly product.

So we saw this at Segment as well. When I joined Segment, we had a direct competitor, mParticle. And they're a great company. They've done pretty well in the market as well. And I expect a bright future for them. But when I joined, honestly, we were scared of them in a lot of ways. They were closing logos like, I think, HotelTonight, I think, DraftKings, Airbnb. Lots of the top consumer companies at the time and beating Segment in head to heads in the enterprise.

That's not happening in our case very likely. But at the same time, Segment gained so many more customers and so many more advocates by focusing on the developer and focusing on building a self-service product that people could just jump from company to company and just start using right away. And that's one thing that we've really, really focused on at Hightouch by being the first to release like a public onboarding flow, public pricing that's consumption-based. And just all these small mechanics actually end up making pretty large differences in the long term. I think there are a bunch of other reserve ETL of providers as well other than Census. There's **[inaudible 00:31:43]**, there's Polatomic, etc. But from what we can tell, and this was highlighted in the VentureBeat article recently too. It's like if you're buying reverse ETL, you're basically going to look at Hightouch or Censes in the market today.

[00:31:56] JM: Help me understand why that is. Basically you have to look at Censor or Hightouch because this is an extremely operationally difficult business, because you have so many integrations to write. Is that right?

[00:32:09] TM: Yeah, I mean, one thing is the integrations. And then the other thing is that, because of how complicated this type of software is, you want to pick like a solution that you trust, to be completely honest. So that solution has to have like a very large integration catalog. But it also has to have a deep integration catalog when it comes to like having a ton of features inside of each integration where you can go into like immense depth here.

I think one thing that we really focus on, I guess a few things that we really focus on in terms of the platform layer, besides integrations, is like really providing developers like a high-level of visibility and like control to what's going on and how the software operates. So I think that's one really important thing as you create a developer platform versus say something like Zapier Tray.io, which totally tries to not focus on the technical user.

So for us, that means things like building debuggers that allow you to like dive deep into the requests being made to different systems by Hightouch, alerts in Slack and PagerDuty. API's for developers to control the settings themselves, the Git integration, and just all sorts of things to make developers feel at home using these tools and want to give away part of the work, which is moving data into different systems back to a SaaS provider. And I think that's something that's

really important to focus on for like, basically, all companies that are taking away some part of work from a developer.

[00:33:32] JM: The problem domain is very big. And I'm sure as you have had conversations with the prominent companies that use Hightouch, such as Plaid, and Blend. Blend is the loan company.

[00:33:48] TM: Yup, exactly. They serve like Wells Fargo, US Bank.

[00:33:50] JM: So, Plaid, Blend Autotrader, Retool, these are like very intense companies, Kong.

[00:33:59] TM: Lucidchart.

[00:34:00] JM: What did you say? Lucidchart? I don't know much about Lucidchart. What does Lucidchart do?

[00:34:03] TM: Basically the charting, like draw my infrastructure diagram software.

[00:34:07] JM: Oh, wow! Oh, okay. It's UML?

[00:34:11] TM: It's the alternative to Visio. It's like there's Microsoft Visio that you use if you're –

[00:34:16] JM: You should just standardize on Figma, right?

[00:34:17] TM: I don't think developers use Figma to draw.

[00:34:21] JM: I love Figma. Figma is the best, man. I do that. Okay.

[00:34:25] TM: Does Figma have AWS icons that you can drag in?

[00:34:29] JM: Okay, that's a good point. What I mean –

[00:34:31] TM: If you see any of those diagrams that have like S3 –

[00:34:33] JM: Does Lucidchart have Firebase diagrams? I doubt that.

[00:34:37] TM: Probably. Honestly, they have icons for everything.

[00:34:40] JM: Really? Okay. All right.

[00:34:41] TM: They have literally the whole like Google Cloud, AWS, Azure.

[00:34:45] JM: Do you use them?

[00:34:46] TM: I use Lucidchart.

[00:34:48] JM: Okay. All right. All right. Okay. All right. Shout out to Lucidchart. Use referral code softwaredaily_tejas so we can share the referral for each Lucidchart. Yeah, so these are serious companies that are using you as infrastructure. That's kind of crazy. Like Plaid, I don't even want to know how much data Plaid has. Like got to be petabytes.

[00:35:08] TM: The security reviews at these companies are insane. I think if you're able to balance like –

[00:35:13] JM: I'm sorry, man.

[00:35:13] TM: If you want to use it with enterprises that are super sensitive to use it, that's an awesome pairing. Yeah. Something interesting actually on that point is like one thing that we've done that's been super useful to work with these larger companies, like especially FinTech and like the healthcare brands that we work with, is our architecture. We call it hybrid architecture. And it basically keeps all the data storing inside of their cloud so they can basically provide us an S3 bucket where we keep all the state for our application. And our databases just contain like pointers to their S3 bucket. It's actually a pretty interesting architecture. I haven't seen any other companies actually do this, but allows us to say we're a SaaS service, while all this data is actually stored in your infrastructure, in your region, with your expiration policy, etc., even the

caches and everything, which is pretty powerful for companies that are trying to meet like compliance requirements where data is residing in my region.

[00:36:06] JM: Yeah, you must be SOC 2 at this point, right?

[00:36:08] TM: Yeah. Yeah, for sure.

[00:36:09] JM: Was that hard? Or is it easy for Hightouch?

[00:36:13] TM: Honestly –

[00:36:13] JM: Or just procedural?

[00:36:14] TM: I think it's not super difficult when you're a small company. It's probably going to be harder to upkeep when you have like hundreds of people. But of a team of like 20, it's really just like not as difficult as one would imagine.

[00:36:27] JM: What is it? Just proper permissionings?

[00:36:29] TM: A lot of it has to do with human workflows in the company. So first, it's all about like the tooling. There's a whole there's a whole array of things in the tooling that you're using to FA or using SSR, using all that sort of stuff that prevents like a whole class of things. And are your devices being managed centrally, etc., etc., etc. And then a lot of it comes down to like human processing to lang. How do you offboard employees? Honestly, a lot of process work actually. I don't know all the other details. My cofounder, our CTO, Josh, like actually did a majority of that. But I think it's a lot harder as the company scales than to get it at the beginning.

[00:37:08] JM: You see that company Vanta?

[00:37:09] TM: Yeah, we use them.

[00:37:10] JM: You use Vanta? Okay. Great. Great. Yeah. Wow! What a powerful company that is. So yeah, man, like reverse ETL. Like enlighten me. Like what's going on in reverse ETL?

What's hard about it? What's broken? What's working? Like what's blowing your mind these days?

[00:37:28] TM: Yeah, I think the default use cases which we entered the market with like sales enablement, pushing data about what your customers are doing, things like Salesforce. Helping to push data in the marketing tools, I think those are pretty straightforward, and they're just getting easier. But I think what's actually cool is the new ways people are using our software that we just wouldn't have imagined. Like that's really what blows my mind and really what excites me about this space.

I think one thing we've we found – And a lot of these can be pictured by the integrations we support, which is kind of interesting. So one thing you might notice is we support a bunch of production databases now as integrations, like Postgres, Mongo, Redis, different things like this. And people are like – And you might think like, "Why would I want to move data from my data warehouse into like a production database system?" And the answer is because people want to use data in the warehouse, like models, like what billing plan is this user on? When does their contract from Salesforce renew? How many credits have they consumed this month with my advanced billing formula that I put on the warehouse? And they want to use those values actually in their applications for like in-app flows, in-app personalization, etc. So it's just very interesting to see like demand for this. And other use cases we've seen for that is like in-app personalization, like you build a model nightly of like is this user a high-value user? Or what's their propensity score? You ship that back to the production database that serves your API. So you can actually customize the app experience based on it.

So I think what's most fascinating for me, as an entrepreneur at least, is just being able to see like – And I think this is just especially true of developer software where you just allow people to do whatever they want, is seeing all the weird ways in which people are using the software. Another thing that's been coming up more and more is like really Xavier-like workflows around like business processes. I think one thing we've noticed is like we never sell our Slack integration. It's like a free integration on Hightouch. But pretty much every company ends up using it in one way or another.

Basically, what it allows you to do is put a SQL query in Hightouch and then just build a Slack feed when things are added or removed to that SQL query. So like in Hightouch itself, we use that all the time to basically power a whole feed that every sales rep at Hightouch watches to see what the customers they're tracking are doing, which is really awesome in my opinion. And I think what's happening is just, in the space, people are realizing that a lot of problems they want to solve just come down to automating something off data. And I think in the long term, that's what we want Hightouch to be associated with as just a horizontal developer platform to do automation based on customer data, rather than sales use cases, marketing use cases or anything of the like. And that's what convinces me that it really has to be a developer platform.

[00:40:11] JM: Yeah. I almost think you can go bigger eventually. I mean, I don't know if you're revealing your entire plan. But like, I really think you can go bigger, because you're basically – You're like the middleware company to end all middleware companies is what you really are. You're like an end-to-end workflows company that starts in a time when API infrastructure is getting really insanely good. So you can basically make very strong assumptions about the contours of the infrastructure and the reliability of the infrastructure, because you're just a routing machine between companies. Like you're routing machine between API's and proprietary infrastructure that is each individually built to be performant. So you're really like this routing infrastructure. Like you're kind of like a service mesh between companies is the way I see it. Is that crazy?

[00:41:00] TM: I think that's a really DevOps perspective to the problem. But it does have some truth to it.

[00:41:04] JM: I'm just trying to hype you. I mean, service mesh was so hypie. I mean, the hype is legit. I'm doing another service mesh series soon.

[00:41:10] TM: DataMesh.

[00:41:12] JM: Don't get me started on DataMesh. Don't get me started on DataMesh. I'm sorry. I had to Zhamak on the show to talk about DataMesh a while ago. I need to have her back on because I have some – I think I have some issues with thesis. I think it's interesting, but I have some issues with it. Do you have issues with the DataMesh thesis?

[00:41:27] TM: I'm scared of the number of companies that are going to use this approach after reading blog posts online.

[00:41:32] JM: Dude, it's just not –

[00:41:33] TM: The new microservices.

[00:41:35] JM: I mean, no. Microservices, I think, makes a little bit more sense. DataMesh to me is like kind of coarse-grained and just like not the right – You want the right metaphor to think about these kinds of things. And I don't feel data mesh is a good metaphor. I mean, look, I really like Zhamak. And I get what she's saying there. And I think she's actually making a profound point. I just don't think the vocabulary is on point.

[00:41:57] TM: Yeah, it's fair. I think for the most part, the biggest thing I'm scared about with these kind of trends of things that come out of companies like Facebook, LinkedIn, Uber, etc., is it just doesn't really apply to a majority of companies. And I hope that a majority of companies stick with a more pragmatic, simple approach to their architecture. That'll make life easier for vendors as well.

[00:42:19] JM: I don't know, man. I think it's okay to future-proof. I mean, I think the infra is good enough these days that you should be future-proof in many cases. I mean, I don't know. Maybe I'm crazy. I just don't think you have to compromise. Everybody's future-proofing.

[00:42:32] TM: I think that the mental overhead of future-proofing actually goes pretty crazy, especially as your team scales and you want to educate people. I'm a big believer of really simple infrastructure as long as you can make it, except on the problems that are super core to your business.

[00:42:45] JM: What does your infrastructure look like? I'm sure you learned a lot at Segment about how to run infrastructure. What's your strategy?

[00:42:51] TM: Honestly, at Segment, I think we learned a lot about what we don't want to do as well. So we have actually a pretty unorthodox infrastructure I think for a data infrastructure company. We don't employ a lot of queues. That's something that we hold really, really strongly within the infrastructure. We just strive for very, very, very simple, I think predictable, and observable architecture. Those are like the main three things we focus on. Like our infrastructure, honestly, it comes down to like three services, the service that has our app, the service that has our API that serves the app, and then a background worker. And all three of those are horizontally scalable and basically power all of Hightouch as you see it today. And this is super important to us, because I think, as you get into more and more complicated infrastructure, it becomes more and more complicated to make bigger changes in the future. I mean, just think about if we want to support – We already have this hybrid architecture right now. But think about if we want to support like fully on-prem. Yeah, sure, with like Kubernetes, and Docker and all these things, it becomes easier to deploy those services on-prem even if you have a complicated infrastructure. But it's not easy to debug them when they're running in someone else's environment. It's not easy to figure out why they've been spun-up properly. Like everything just becomes more complicated, complicated infrastructure.

So one thing we focused on is just like very simple infrastructure that's easy to observe and easy to figure out what's going on. And like I think, one, I guess half, I guess, that we've been – Which is pretty common at this point that we've been doing based on our experience at Segment is to use a database as a work queue, which works pretty well. So all the jobs like obviously scale horizontally and pick up like small batches of work, but the database itself, our Postgres databases is what's used as a queuing mechanism. Whenever we need to persist stuff, we don't use like a typical queue, like Kafka, or even though we're processing large, large amounts of data.

[00:44:42] JM: Can I ask you totally crazy question? We're almost near the end of our time. I just want to end on a crazy question. And we'll just like – I think the next time we have to do this in-person. I'm going to message you this right now.

[00:44:52] TM: So you are a fan of in-person.

[00:44:54] JM: Yeah, it's so much better. I've been doing it lately. It's so good. It's so good. I'm going to email you right now just to schedule this. But here's my crazy question. Where does this integrate with crypto? And what I mean by that is – So there's a few crypto related things. First of all, you're playing in the middleware space. Crypto middleware is going to get interesting. Like what that looks like, it's kind of unclear, but maybe it's something like – I don't know. You can imagine cheaper Twilio that's like crypto-enabled, or maybe some sort of IPFS-based solution, or an Etherium, like a smart contract-based solution. Like once smart contracts are synonymous with API's, then, presumably, if you're a routing infrastructure company – I mean, if you're Hightouch, you're routing infrastructure company. Maybe you're hitting crypto stuff eventually. Is that outrageous? I mean, that's like pretty far off, obviously. But like is this something you think about at all? Are you just like, "I'm so ready to get off the phone with Jeff at this point."

[00:45:48] TM: This is not something I have ever thought about.

[00:45:52] JM: Is it credible? Or do you think I'm a troll?

[00:45:55] TM: I can't tell, Jeff. I think you're too smart to consider a troll. So I'm worried to make sure I don't say anything stupid. But this is not something I've ever thought about or our customers have asked us for. They have asked us to call their own API.

[00:46:09] JM: All right. Let me ask you a more general question, more general question. That's too specific, too nerdy, to stupid. The more general question is do you think there's a place for like the cryptoization of the – And this is like totally unrelated to Hightouch, or reverse ELT, or whatever? Is there a room for internal crypto-related reward systems within a company? So imagine if you do something really, really good for the company, like you ship a pull request over the weekend that saves the company. Some sort of like almost tipping system that's like – You know how Google has that thing where you can recommend people for bonuses, for promotions? It's actually quite a powerful system. It sounds kind of like a joke, but it's very powerful. It really gets the organization motivated in the right direction. I feel like you can do that at a fine-grained level. And I feel like crypto can really help enforce that. You can almost do it in a way that like translates from company to company where you have almost like an internal bit cloud sort of thing. Like at corporate bit cloud.

[00:47:09] TM: It's interesting. So one, I think, incentives models, like if you can have like a true goal that makes sense for your company and you can provide some sort of incentive associated with it. That's like incredibly useful as just like a tool. And I think, I hope in the future, more companies figure out good ways to do like a sort of incentive-based equity and things like that. But I feel like the biggest problem is figuring out the right incentives, rather than the technology behind it. I think, for instance, stuff like that, it's pretty reasonable. The biggest problem is figuring out the right incentives that don't incentivize stupid behavior that's actually not productive. Especially at a larger company like Google, I feel like it's very easy to do that at a mass scale by producing some sort of micro-incentives, like submitting pull requests on the weekends gets you more points or something like that. I think that's the hardest thing to figure out. And then there're some friends that had like equity-related bonuses if they hit some like hard goal at R&D companies by the end of the year. And it turns out that that call no longer matters so much for the company and it's just a really bad incentive. I honestly think it's not a technology problem as much of like an incentive and company planning and that sort of problem. But as you said, like I think I'm a huge fan of those sort of ideas where you can have like incentive-based equity rather than just equity or are comp and based on what your resume looks like at the time you're hired.

[00:48:37] JM: Alright, so we got to wrap up. Obviously, I was totally off the beaten path here. I'm kind of trying out like postmodern Software Engineering Daily. So we'll see how that develops. What should we talk about next time? Like what should I start to think about until our next like the in-person interview? What are the topics you'd like to sort of bat around or you're thinking about?

[00:48:58] TM: I think one thing that's just interesting to talk about in general is kind of the future of data warehouses. So one thing I've been thinking about really is this this idea of data cloud that Snowflake has been promoting. So the idea that, I think, as developers expect more and more abstractions. They want a big part of the abstractions of building any software or software platform, even like Hightouch is the database that backs you up. I think something super powerful is, as all data about the user becomes consolidated into a single database, like a data warehouse, that underlying platform itself can get more powerful supporting features like streaming, or low-latency reads, marking certain cables to be replicated in an OLTP fashion,

things like this. Like it unlocks huge value for how like software applications or data integration is written. I think there's a lot of interesting stuff happening in space. I mean, things from Firebolt, which is like why do data warehouses not need to have indexes? We can just add indices to data warehouses. To things like that Materialize, which is like why are SQL queries run once in a while? Why can't they be run in like an incremental streaming fashion? Just Snowflake itself hiring the streaming team from Google to build streaming –

[00:50:11] JM: Are you serious? They hired Tyler Akidau?

[00:50:14] TM: Yeah, exactly.

[00:50:15] JM: Oh my God! You're kidding me. Why doesn't that guy start a company? I've been trying to interview him for years. I can't get him on my show. I just want to talk to Tyler Akidau. This is the godfather of streaming basically, as far as I understand.

[00:50:28] TM: We'll send him an email and see if we can get him on the show.

[00:50:30] JM: Okay. He is the best person in in data streaming, right? Like he knows more than anybody else pretty much.

[00:50:36] TM: That is my understanding. That's what I've heard from people. And that gives me faith that –

[00:50:40] JM: Have you ever tried to read the Dataflow paper?

[00:50:43] TM: I actually haven't. Have you?

[00:50:44] JM:. Yes, it broke me.

[00:50:45] JM: You said tried to read it. So did it just break –

[00:50:47] TM: It broke my brain. I felt very stupid reading it. They talk about these things called watermarks. Do you know what a watermark is?

[00:50:56] TM: Yeah. I mean, that seems like a pretty reasonable –

[00:50:58] JM: Okay. All right. Maybe that's a pretty easy, easy concept. Okay, I'm not a distributed systems guy. Like this is hard for me. Watermarks are hard for me.

[00:51:05] TM: It's basically just a checkpoint, right?

[00:51:07] JM: Yeah. I mean, it's a checkpoint, but like the idea of when you use it in a streaming system. For example, if you're counting. Let's say you're doing a time series in a game, right? You've got a mobile MMO over a bad network, and you're streaming time series data to the backend. What does that look like? That's kind of hard.

[00:51:26] TM: Honestly, I think people think streaming is very easy. They're like, "Oh, but you can probably just like parse a SQL query and split it up into a bunch of aggregates." But then when you get into like how to actually do that on paper without just extremely blowing up cardinality and doing it at scale, it's just like your mind breaks.

[00:51:42] JM: I mean, so this is actually – This is the true reason why data warehouse beat out streaming systems, is because the streaming systems is just terrible at this point. They're not good enough. They're bad.

[00:51:53] TM: Literally a decade ago, there's all this hype around like real time, etc., etc., etc. And then people realize that, "Oh, I think, good software that's easy to use and doesn't break is just reliable and supports all use cases." Like a data warehouse outperforms specialized software any day. And business requirements are flexible based on what software can be written in that scale. It just made batch a lot faster at the end of the day.

[00:52:20] JM: Well, that's what I was going to say. What's going on with Spark streaming? Is that good now?

[00:52:23] TM: I think Spark streaming is still not –

[00:52:25] JM: It's still like micro batch, like micro batch, which actually micro batch sounds kind of practical to me.

[00:52:33] TM: It's pretty practical. I mean, actually a lot of companies are doing micro batch systems on top.

[00:52:37] JM: Actually, if we're talking honestly, isn't streaming a subset of micro batch?

[00:52:40] TM: In the most literal sense, yeah.

[00:52:44] JM: And they're the RadLab people. Like they're literal, right? That's the point is, is like we're actually going to make this thing literal. It sounds worse, but it's actually literal.

[00:52:52] TM: Especially with – There's just a fundamental tradeoff between throughput and latency. You can't you can't –

[00:52:59] JM: There's no streaming. Like streaming is a myth. We need to make a podcast. We should call this podcast episode Streaming is a Myth.

[00:53:07] TM: And then just bring a bunch of guests from the streaming space on.

[00:53:10] JM: Yeah, it's like we're using Kafka to stream the data. Like, "Mmm, okay. That's kind of –"

[00:53:14] TM: And then you just read out batches of that, right?

[00:53:17] JM: Yeah. Okay. All right. Well, Tejas this has been super fun, as usual. Let me know when you can get dinner. And I sent you an email for booking the in-person interview. We'll do it soon.

[00:53:27] TM: All right. Awesome. And then that one, we'll try to go on a smaller golf course. Maybe we'll play a game of mini-golf on that episode instead of a –

[00:53:35] JM: I basically wanted to just – This is just a teaser for the in-person one. That's all I wanted this to be. This is our long teaser. People are so hyped at this point.

[00:53:43] TM: Fair enough.

[00:53:43] JM: All right.

[END]