EPISODE 1385

[INTRODUCTION]

[00:00:00] KP: Application observability is a fairly mature area. Engineering teams have a wide selection of tools they can choose to adopt, and a significant amount of thought leadership and philosophy already exists giving guidance for managing your application. That application is going to persist data. As you scale up, your system is invariably going to experience problems beyond common metrics like response time, CPU usage, and throughput. At some point, you'll experience data issues, which in many organizations are discovered manually during root cause analysis, which links a downstream problem to a change in a database that wasn't caught.

Metaplane is a monitoring tool for your data warehouse. It monitors your tables and raises alerts when issues such as anomaly detection occur. In this interview, I speak with Kevin Hu, cofounder and CEO of Metaplaneplane. We discuss how their solution brings observability into the data warehouse.

[INTERVIEW]

[00:01:01] KP: Kevin, welcome to Software Engineering Daily.

[00:01:04] KH: Thanks for having me. Big fan of the show, and really looking forward to talking with you.

[00:01:08] KP: Well, let's get right into it. For listeners who haven't heard of it yet, what is Metaplane?

[00:01:15] KH: We think of Metaplane as the Datadog for data. So it's a data observability tool that continually monitors your data stack, for example, data warehouse like Snowflake and a BI tool like Looker. And we alert you when something goes wrong, and provide you with relevant metadata that you need to debug.

[00:01:34] KP: So what does it mean for something to go wrong?

[00:01:36] KH: Well, in the software world, you kind of know when systems are up. And for data teams today, it's really hard to get a sense of whether your data is correct. And we think of those as silent data bugs.

For example, if a table is usually updated every hour, but it's been over a day since that's been updated, or you have a tool that's loading a million rows every day, but suddenly one day it doesn't load any rows. These are the kinds of bugs. Those are simple examples that it's really hard to know about as a data team.

[00:02:10] KP: And those are like, kind of really obvious points we'd want to track. So it's good to have a tool for it. Do I have to define those the way I write my unit tests? Or are these things you can put some guardrails on for me?

[00:02:22] KH: For Metaplane, we do that for you automatically. We try and get as much data from your database metadata tables, for example, information schema as possible. And then we have the anomaly detection for you in the background that tries to take trends and seasonal patterns into your account. Although you can always have manual thresholds as well.

[00:02:46] KP: Could you expand on that sort of schema exploration? Are you going so far as to look at the columns data types? Or maybe the name of it like this is the revenue column? Can you get some of the context out of it as well?

[00:02:59] KH: We do go pretty deep into the data types, and renaming, and tracking the statistical characteristics of different columns and letting you know how those change over time. So to give you an example, two numeric columns can have completely different behavior, depending on what in the real world it's tracking. So we want to have models for each of them.

[00:03:22] KP: And it's interesting you can do that broadly for a large category of different businesses and data setups. It seems like I could have something that is a Black Friday special. And I don't know that it's necessarily known in the database. Not everyone's in ecommerce. What happens when a explainable anomaly shows up?

[00:03:39] KH: We do have a bunch of customers across a wide set of verticals from ecommerce. Like you said, Black Friday is a huge deal in that world to B2B, to FinTech, to healthcare. And the interesting part is, across all these verticals, they all use the same tools, right? Snowflake, DBT, Looker. And when an anomaly comes up, we tell them, “Okay, these are the upstream tables and dependencies. These are the downstream dependencies. For example, these dashboards depend on your table.” And that hopefully gives you enough information to know whether or not this is a P0. How much should you prioritize this issue? And give you some leads on how to start debugging it.

[00:04:23] KP: Interesting. So Metaplane is going to shoot me an alert. Not just, “Hey, we saw something weird, but here's some downstream effects that you might need to consider.” How do I get that alert?

[00:04:32] KH: You get it in Slack, or an email, or in Datadog, wherever your team lives. Of course, you can get it within Metaplane as well.

[00:04:40] KP: And you see some common early adoption use cases? Are there low-hanging fruit things people find? What are some of the early insights?

[00:04:49] KH: Some of the most common issues across verticals and across team sizes, whether you're a mighty one-person team or a 50-person data team, are schema change issues. For example, if an upstream product team changes the name of an event, that's quite common. We'll send you alerts about that. Freshness issues also come up all the time, right? A table is not updated as frequently as you expect. And volume issues come up too. But there's really a long tail of data issues. Like the saying goes, right? All happy families are alike. All unhappy families are different in a unique way. Same thing goes for data.

[00:05:30] KP: So let's imagine you're considering doing a deployment with a customer and they've got a DBA who's very particular and is concerned that you're going to put extra load on their systems? How do you put that person's mind at ease?

[00:05:43] KH: We try and take advantage of the metadata that's already available as much as possible. So for the most important categories of tests, like volume, and freshness, and schema,

those are typically handled and made available for free from the warehouse. And for tests past that point that do require table scans, we try and take advantage of concurrency and compress those in time as much as possible to minimize the overhead.

[00:06:12] KP: I don't know if you guys have a way of categorizing or any sort of taxonomy for the types of alerts, or I guess the root causes. Maybe you don't necessarily have insight into that depending on your customer. But I guess I'm overall curious about what you're alerting me for. Are these always bad deployments or a site went down? What are the kind of typical root causes?

[00:06:32] KH: Great question. The root causes can include, for example, a third-party dependency, whether it's an event collection system, or an ELT system goes down or has a bug. It can be within your own team. For example, the engineering team or platform team introduces some error that causes downstream data issues. It can also be within the data team. So if there's a lot of people working within one data warehouse continually introducing new transformations and models, they can pretty frequently step on each other. And those relate to systems. Of course, data itself, the data points, can have errors introduced at any point along the value chain. Going all the way up to, for example, a salesperson entering a deal size that's 10 times larger than you might expect, or the wrong currency.

[00:07:27] KP: Hmm, interesting. Is there a common title of the user who's most active with Metaplane at a company? Is it a Director of Engineering, a QA engineer? Who's the most engaged user typically?

[00:07:40] KH: It depends on the size of company. I would say the median title is either a data engineer or an analytics engineer, and the head of that team.

[00:07:49] KP: And what do they do as far as configuration or personalizing to get things really idealized for their use cases?

[00:07:56] KH: we try and make as easy as possible to get started. And once we do you send alerts, you can introduce configuration along the way. For example, to ignore schemas that fall into one pattern, or to not add test to those certain types of tables, the most important thing from

our perspective is being able to give feedback to our models, where if we do market anomaly, you can exclude it, and will continue to alert you on that. Or you can mark it as normal, and we can start including it into our confidence intervals.

[00:08:27] KP: Interesting. Could you talk a little bit about the personalization of those models? I mean, maybe you can do some sort of aggregate type work about the types of things people broadly consider a false-positive or something like that. But it also seems like it's highly custom to each user, and you have to maybe train individual models or things like that. How does it all break down?

[00:08:47] KH: Great question. There's not one type of model. There're actually several models under the hood. And they're a little bit custom to the world that we live in, which is the data world, where using kind of off the shelf models for time series analysis, like we may be familiar with like profit, or any other time series analysis tool. It doesn't apply very well to the data world. Because in our world, data doesn't change as high of a resolution typically as those models assume. And there's a few underlying assumptions as well.

So to give you an example, if you have an incremental model, a table that's growing by 10 million every day, our expectations for what is normal actually depends on what direction the outlier is in. And a decrease in row count is much more severe than an anomalous increase in row count. So that's something that we try and take into account.

[00:09:44] KP: And the feedback you're getting, can you talk a little bit about how valuable that is? If I'm a user, I'm incentivized to give you the feedback because you're going to improve the alerts. And I can't expect that one piece of feedback can necessarily change the whole model. I mean, I guess it could, but sort of asking a lot, you probably need a few data points. What's a typical cycle look like in terms of giving feedback and seeing a change?

[00:10:07] KH: It depends on the inherent cadence of change within the data. So data that changes every hour, we can pick up on the seasonality quite quickly. For example, within a day, you have 24 data points. To give you the Black Friday example in an ecommerce situation, we will only observe that once a year. And the importance of getting feedback in that situation is very important. Because we don't actually observe the anomaly that often.

[00:10:39] KP: I suppose it'll really depend on the, I guess, data hygiene and the type of data someone's storing. But I'm curious if you can set an expectation for someone signing up. How many alerts are they typically getting per week or per month on average?

[00:10:54] KH: I would say anywhere between three and eight alerts a week. We are very conscious about alert fatigue. If you're listening to this podcast, you probably have Slack channels that are going off constantly. So we really try and pair-down alerts that don't require your attention.

[00:11:13] KP: Makes sense. And the alerts, we've mentioned there's a couple places to connect them. And I can get messages in Slack. Can you talk about some of the other integrations? Like what if I have some existing ticketing process? Can I do an integration of some kind?

[00:11:25] KH: You can. We are rolling out features for you to automatically create workflows based on Metaplane alerts. You can use existing alerting logic if you have a paging system, like PagerDuty or Opsgenie. At the end of the day, it all rolls into our number one goal, which is to help data teams save time.

[00:11:44] KP: And where does that savings usually come up? I mean, obviously, getting alerts to problems early is beneficial. Where does the time savings come from?

[00:11:53] KH: One major distinction with software engineering that I would make in the data world, and we can go into a fuller comparison later, is that data issues tend to compound, and they compound very quickly. So to give an example, if your team is using data to predict revenue and churn, and then suddenly you have duplicate rows, then the duplicate rows are propagated downstream to all their different systems. And in many cases, that can impact people make decisions on revenue and churn and how resources are being allocated.

And the reason I bring that up is because, in the data world, time to identification is extremely critical. And with Metaplane, we try and reduce and minimize the type identification. And as a result, that also minimizes the time to resolution because the number of downstream issues is

decreased. And a lot of time savings comes from that bucket of work. The other bucket is work that is spent trying to ensure trust. Because today, frequently, data teams aren't the first to find out about data issues. It's the downstream executives that are the first.

And this makes sense. When you have hundreds or thousands of dashboards and hundreds or thousands of tables, you cannot be expected to audit all of them. And yet, when your head of marketing slacks you saying, “Hey, what is this number like weird? Or why hasn't this dashboard been updated?” It does have an effect on how much your team trust the data. And trust is very easy to lose, hard to regain. And every time that that happens, it takes quite a bit of time to rebuild that trust. So that's the second category of time savings.

[00:13:42] KP: Well, everything I've learned about the product suggests it's generic. And I mean that in the good way. You can apply it to just about any data project you have. Connect your data, and there you go. So pretty much any company or org could take a look into it. I'm curious if you've seen any patterns of adoption. Maybe certain sectors or styles of companies are much more sensitive to their data and it becomes a natural product. Are there any trends like that?

[00:14:05] KH: I would say that just as software is eating the world, data is beginning to eat the world. It's not quite there yet. And the adoption of modern data tools is happening quicker we've found in verticals in which data is directly tied to the company's success, and top line and bottom line. For example, with ecommerce, and financial services, and healthcare. In those verticals. Adoption of Metaplane is almost like a no brainer, right? If you increase data quality, you're impacting the entire company. That said, I think with the trends that we're seeing in the data space, that is becoming indispensable across verticals. And Metaplane kind of fits in afterwards.

[00:14:55] KP: So Metaplane is a relatively new company, but you've been live with some customers for a while and been doing some launches. Can you tell us the narrative? What's the timeline been from inception to current?

[00:15:06] KH: I've been working data for the better part of a decade now. And I'll spare you the details of that personal journey. But let's just say we started Metaplane about eight months ago.

And we brought on some amazing customers, some early partners. And today, we're helping dozens of teams, all different sizes, and across the world catch these data issues.

[00:15:28] KP: Are there any lessons learned from some of those early customers?

[00:15:32] KH: We originally thought that if we didn't catch data issues that we would be not useful. But I think we quickly learned that the value of awareness, we highly underrated it. Because to give you a real sense, like many data teams have very little visibility into the health of their data today. And just having Metaplane there to have a historical record of how this data has been trending, or how the schema has been evolving, or how people have been using the data over time has been extremely valuable to many of our early adopters.

[00:16:10] KP: And do you have to worry about compliance and issues like that interacting with companies data?

[00:16:14] KH: We worry about a lot. Of course, this is 2021, going on '22. There is no excuse for being lacks on security and privacy. However, I would say that Metaplane, we don't retrieve or store any sensitive information like PII. We only store metadata and summary statistics. So the surface area of sensitive data being exfiltrated is minimized.

[00:16:43] KP: It makes sense. Could we get into depth about some of the platforms you – I don't if you call them partners or you sit on top of. You'd mentioned Snowflake. What are some of the other connectors available?

[00:16:53] KH: We support the main data warehouses, Snowflake, Redshift, BigQuery, also transactional databases like MySQL and Postgres. We have a very deep integration with DBT. We are partners with DBT. We love them. And downstream BI tools like Looker, Tableau and Mode.

[00:17:12] KP: And what's a typical – Or maybe nobody's typical? But are there any patterns you see in adoption? Is this easier commonly onboarding with Snowflake? Or are people really consuming lots of services at once?

[00:17:25] KH: I'd say that many companies are using Snowflake, and DBT, and Looker, and that might be the predominant stack. And companies can implement us in like less than 10 minutes. Just put in some read only creds the Snowflake and they're good to go. And then once the warehouse is connected, then you can start adding layers on top of that.

[00:17:49] KP: The analogy is not so – Or was it anagram, the letters you've got? So Metaplane be the M? S for Snowflake. D for DVT. And L for Looker. Is that the **[inaudible 00:17:58]** stack? Do you see it forming in that kind of way?

[00:18:01] KH: We'll have to shop it a little bit.

[00:18:03] KP: Yeah, yeah. I didn't give you the name by any means. But –

[00:18:08] KH: We'll shop it. I mean, our goal is for every data team to have a data observability tool. Like it the moment, you bring on Snowflake, we want you to be able to bring on Metaplane. The same way that the moment you guys are on AWS, you can bring on Datadog.

[00:18:24] KP: In terms of observability it's a topic that's kind of grown, I think, in consciousness. It's always been there. But the tooling has gotten better. The attention has gotten more assigned to it. What are some trends you're seeing in terms of like from large enterprise down to small? Is their early adoption and awareness that observability needs to be there? Or is that something those companies mature they're discovering?

[00:18:45] KH: Both. For companies adopting a modern data stack, it's almost like a ticking time bomb, like a matter of time until data breaks. Now I wish the best for your team. But what we found is that it is an inevitable side effect of data becoming a product. And by that, I mean, data within organizations is no longer just used to feed reporting BI dashboards. Although that is very important. Is starting to go into the product, into your machine learning applications. Being fed back into go-to-market workflows.

And as data is being used across these use cases, the stakes are higher. That data cannot break. Otherwise, it has big downstream consequences. So as we see data becoming a product, we see teams both kind of like reacting to, "Oh, shoot. This is breaking, and teams

being more proactive about bringing on observability." And I think that that is fitting into a bigger trend of the engineering furcation of data, where a lot of the best practices developed in the software world. So to give you an example, like version control, test-driven development, infrastructure as code, is starting to impact how the data world delivers IT services and products.

[00:20:01] KP: When data breaks, and your customers are going to rely on you to help them find that. Do you have any sense of the obviousness, I guess, of some of these breakages? If a table is suddenly half the number of rows, it makes sense to throw an alert or something like that. There could be subtle shifts that are below some threshold or something like that, although maybe that's just the ebb and flow of the business. And the real alerts need to be around the sort of more in your face alerts that you would see if you monitored every table all the time. I guess my question is what is the nature of these data breaks?

[00:20:35] KH: Some of them are very obvious. For example, a table completely disappears. Other ones are more subtle, like you said, and we need a human loop with a domain expertise to evaluate whether or not this is expected. But across the board, it shouldn't be silent. And to give you a sense, can you imagine if you're on a software engineering team and the only way you could find out that your software was not working well is if you didn't hear anything from your customers? Or to inverse that a little bit. You don't want to find out that your API is slow when your users complained to you, right? You want to use application performance monitoring, right? Pop into Datadog to find out that it's a slow Postgres query and you haven't properly indexed this field. That is an example of a bug that would have been silent. But with the right tooling, you are able to find out about it first, right? We just want to make it so that data teams don't have to deal with the silence.

[00:21:38] KP: I think in software observability, the challenges might be easier. Response time, like you'd mentioned, sort of natural measure. We can look at the CPU and memory and stuff like that. They're well established patterns. And it seems like they're relatively consistent from project to project. Data observability feels harder to me, and that you have a different context at every project. Do you face challenges along those lines?

[00:22:04] KH: As an old computer science professor used to say, things are exactly the same, but completely different. And data is different from software fundamentally, right? Of course, data is often generated and transformed by software systems. But there are unique aspects of the data world. For example, the compounding problems that we talked about before. If data breaks, it gets worse and worse and worse. And data has lineage, right? It has a place that I came from and steps that it was transformed along the way. And data has a weight. It's kind of difficult to move data from one system to another. It takes time to transform it. So I think those are some differences.

And to get to what you're saying, there is a lot of context and interpretation that goes into data. And I would say that that probably applies most at the end state when it is aggregated into metrics that are interpreted by business stakeholders. But along the way, there are some commonalities kind of across businesses, right? Just by the way that data is being modeled and stored, for example, the number of rows probably shouldn't tank versus historical expectation, no matter what kind of company you work in.

[00:23:20] KP: Let's dig deeper into that comparison of software and data observability, and maybe why Datadog isn't the tool that allow you to do this?

[00:23:27] KH: Well, in the software world now, there are a lot of great workflows and tools to help you identify the bug, fix software issues. You have CI/CD tools to run into plane river tests. You have continuous monitoring and APM with Datadog. So when an issue occurs, there are sophisticated tools to know what's impacted, and you can backfill issues, communicate to the users of your software. And we think of that, like there's a great playbook both technically and operationally, great playbooks, for understanding and fixing software issues.

In the data world, there are no playbooks, where that the teams are often the last to find out about issues. It's very difficult to confirm whether an issue has occurred and finding out what the downstream impact is and what the upstream causes might be. And the reason why it's difficult to use Datadog for this is, well, on one hand data dog is amazing. And it's definitely possible if you spend the time to wrangle Datadog to your use case. There's a couple of differences, for example, being able to pull in lineage from downstream and upstream tools that are used by data teams, and the ease of adding actual data tasks without having to write code.

[00:24:44] KP: These aren't necessarily new ideas that no one thought of before this year. Why is it that the data observability marketplace is less mature perhaps than application observability?

[00:24:56] KH: I think one angle is the need has increased, where data is being used in organizations across a much wider set of use cases than it has been used in the past. For example, AI/ML use cases for improving the product, like go-to-market use cases for sales, marketing and support. But on the technology side, what really made this possible is data warehouses with infinitely scalable compute that you can like run separately.

So in previous generations of data warehouses, there is always this conflict between integrity and performance, where if you run hourly checks to query your database, frequently, you'll step on the toes of the people who are actually using your data. But today, if you use Snowflake, you can create one warehouse that is used by an observability tool like Metaplane to make sure that data is in good shape and high-quality without impacting the rest of your data consumers.

[00:26:00] KP: Definitely. Well, to make another similar analogy, software versus data. Software quality testing seems like a pretty advanced science, right? We write unit tests. We look at code coverage. And it's not perfect, but there's a pretty formal process. And usually if you make a mistake, you know the day of the release, or quickly thereafter. Data quality, on the other hand, is something I think everyone's committed to. But not everyone has a clear picture of what that even means. Do you have any good definitions or maybe qualitative ways that we can measure and look at data quality?

[00:26:33] KH: Data quality, ultimately, is operationalization of trust? How much do the end users trust the data? And degree to which end users trust the data kind of depends on the properties of the data itself? So there's a whole like list, like dozens or hundreds of potential data quality metrics that you can use to kind of apply metrics to your metrics, ironically.

And to give you some examples, some of them are intrinsic metrics. For example, does my data store respect referential integrity? Or is my data complete? And there are also some extrinsic metrics that depend on your use case, such as is the data in my dashboard being refreshed in

the right amount of time? I would say it really depends on the use case that the data is being used for.

[00:27:32] KP: When you're seeing early customers, are they coming typically with an expectation for these? There's maturity of data quality and a recognition of these are some of the metrics we want? Or are you often a discovery platform where people are kind of just figuring these things out, they found you, and now they're getting more into their data?

[00:27:50] KH: I think it's a journey to introduce and learn more about data quality. I think data teams know that it is a Sisyphean problem, right? You're pushing a boulder up a hill. Data quality will never be perfect. But you can get stronger, and the boulder can get smaller. And you can get a sense of where you are on the hill at any given time, which makes it manageable.

[00:28:14] KP: And speaking of adoption, who's the typical, I guess, champion bringing this in? I've been in positions where I was sort of an analyst role and really wished a tool like this had been around, because I was seeing so many data quality issues and trying to ad hoc find them. But I don't know if that's typical case or if it's more management down. Who's the first interested party?

[00:28:34] KH: First of all, sorry, you had to deal with that. Next time you deal with that, hopefully we can help you out. Typically, it's the practitioners themselves, right? Boots on the ground, SQL in the console. That are the first to face the problem. Of course, it boils up and impacts the rest of the business, in which case, the data owner starts getting involved and start treating it like business priority. But really, like you're suggesting, it's the person who is working with the data. That's the first to be impacted.

[00:29:06] KP: So to get into prioritization then, maybe I have to champion the product. Do I have to go and ask for budget? Or what's a typical onboarding for maybe a medium enterprise company?

[00:29:17] KH: You can just try it out. Metaplane plane is completely self-serve. We have a pretty generous free tier that lets you monitor your warehouse alert to Slack, and paid tiers that make sense for different sides of the data teams. From our perspective, you shouldn't have to

talk to a salesperson to bring on a data observability tool. You don't have to talk to a salesperson to bring on anything else in the data stack, from Snowflake, to DBT or Looker. Maybe eventually. But to try it out, you don't. And we want to do the same thing for observability.

[00:29:51] KP: Makes sense. Yeah. For you to try is always a great entry level point. I'm imagining now someone who's, let's say, a manager at a company who hasn't necessary really taken data observability very seriously. But he's got a champion on the line who wants to try out Metaplane? In terms of time and resource allocation, what should they be thinking?

[00:30:12] KH: I mean, to be honest, that if you have a data practitioner who's able to get the credentials and set us up with Slack, you can get everything connected in less than 10 minutes. Of course, if your company has different compliance requirements, you can work with us to work through that to set up an SSH tunnel or a reverse SSH tunnel. That can take a little bit more time. But you should be able to do it within a day for sure.

[00:30:38] KP: Where do you see Metaplane going in the next, I don't know, five years?

[00:30:43] KH: So, for us, we think of data within companies as there's three levels here. One is you'd have data. And there are amazing tools like Fivetran, and Snowflake, and DBT that are making that possible. Then you need to use it. And that's where BI tools and reverse ETL tools come in. But the third level is you need to trust the data. And the moment the business loses trust and data, then it all goes out the window. What was it all for? It's very hard to get it back.

So we have to be a platform that you can bring on to build trust with data. And for us, that means giving observability into it very broadly, not just for quality monitoring. Quality monitoring is a use case for observability. And for us, we think of observability as do you have visibility across your data stack throughout time? Can you say confidently that seven days ago that your Looker dashboards were all up? So we try and pull in as much metadata as possible to be as useful as possible so that you're very confident in answering that question.

[00:31:49] KP: Are there any customer use cases you're able to share? Even if we don't go into details, what are some real-world scenarios you've seen the platform is made visible?

[00:31:58] KH: One funny real-world scenario is one customer, and ecommerce customer had the average value of an order increase from, let's say, hundreds to tens of thousands. And this was traced as a Metaplane notify them saying that this metric had anomalous increase. These were the downstream Looker dashboards that were affected in the downstream BI models. And eventually, they traced it all the way back up to a data entry problem where it was reporting in the wrong currency. So this is a kind of a small situation, but it happens. You'd be surprised all the time.

[00:32:38] KP: Absolutely. Well, let's talk a little bit about the technology behind. Can you tell me a little bit about how Metaplane is built?

[00:32:46] KH: Definitely. So Metaplane in the backend is all on Amazon ECS. We fire up containers every hour that handle the task to query a database and as compressed amount of time as possible. We use our queueing through Amazon. And every hour, we retrieve what we call an observation from your data stack, whether it is like the row count from info schema, or is the schema the database, or **[inaudible 00:33:18]** from Looker. We store that. And then we train our models and run a prediction to determine whether or not it was an outlier. The frontend is on React. And the backend is in TypeScript and Node.

[00:33:34] KP: Nice choices. Could we zoom in on the outlier detection? I'm curious to know what you're able to share about the algorithms or approaches.

[00:33:42] KH: They’re pretty custom. It depends on the type of tests. So to give you an example, row count tests, which I mentioned before, you never want those to drop, or verily you want it to drop if it's always been increasing. So for row counts, we've developed kind of like an asymmetric time series analysis model, in which like, over time, every single delta, we have an expected amount of change. And we have one upper confidence interval, one lower confidence interval. And every time that's violated, you get an alert.

For freshness tests, we do something closer to survival analysis. It’s not where we try and predict like the time until an event. Time to an event. And in this case is predicting the time to a refresh with more strict or penalization for every unit of time that elapses since what you're

expected. For some of the more generic models, we use some things like off the shelf plus a bunch of rules that we have on top.

[00:34:44] KP: What's coming next? Can you talk about the roadmap at all?

[00:34:48] KH: Definitely. We are trying to make it as easy as possible to set up tests and automate it. And we want to take advantage of the data that you're already have within your warehouse to do that. So if you have query history that's available, for example, if you're using Snowflake, then we're parsing your query history to determine, "Okay, these are the columns and the tables that are most used by your end users. You should probably have a test on those." Or, conversely, these tables are not being utilized. Maybe you can reduce the number of tests on those tables.

We're also parsing lineage for many of our customers using the query history so that you don't necessarily need to use a tool like DBT to have upstream and downstream lineage. Now, that's a difficult problem. And I think that we're taking a good stab at it. So that at least when we send you an alert, that you can be quite confident that the lineage that we provide is accurate.

[00:35:47] KP: And is there a vision for expanding into other databases and tools like that? I mean, you've got a good suite that pretty much covers all the major use cases. I'm just curious if there's a reason you want to add MongoDB or some of these other certainly fringy but smaller market share of databases and such.

[00:36:06] KH: For sure. We have a lot of customer pull in two directions. One of them is to go more and more downstream. So for example, to monitor metrics that are being reported on dashboard. But especially we have a lot of pull to go more upstream. Like you're saying to MongoDB, or even to SaaS applications, like Salesforce and Stripe, and doing some validation before it gets into the transactional database or into the analytics database. And kind of go all the way upstream to do much better root cause analysis for you. Because at the end of the day, Snowflake, and Databricks, and your data warehouse, they don't produce the data, right? They may be the center of gravity for data, but the source of truth is ultimately upstream. So as much of that reconciliation we can do for you, we want to.

[00:36:53] KP: And in terms of the downstream, could you speak to maybe some of those use cases? What are the types of metrics that customers would be interested in beyond what you already offer?

[00:37:02] KH: We have a lot of pull into tracking which reverse ETL destinations are being used. For example, many of our customers use high-touch and census and polyatomic. So like tracing it all the way from a Snowflake table or field to a Salesforce field. We also get asked a lot to track metrics within Tableau and within Looker. Now, that's a little bit complicated, but I think we can make some real progress there.

[00:37:34] KP: Interesting. Yeah. Well, Kevin, where can people learn more about the product online?

[00:37:38] KH: You can go to metaplane.dev or follow us on Twitter. That's twitter.com/metaplane.

[00:37:43] KP: Thank you so much for taking the time to come on Software Engineering Daily.

[00:37:46] KH: Totally. My pleasure talking to you, Kyle. Take care.

[END]