EPISODE 830

[INTRODUCTION]

[0:00:00.3] JM: Relational database systems have evolved from single node instances to complex distributed systems. Almost any database can be accessed through a SQL statement, but the guarantees of these different databases can vary in terms of consistency, availability, latency, durability and financial cost.

Relational database systems that explore these different sets of trade-offs are sometimes categorized as new SQL. There are also a wide variety of data systems that are not categorized as databases. Kafka is a distributed queue. HDFS is a distributed file system. Spark provides a distributed end-memory working set to process data. Cloud providers offer hosted bucket storage for your data lake and fast processing in the form of data warehousing technology.

Sunil Kamath is a Principal PM with Microsoft and he's worked on database systems for two decades. He joins the show to give his perspective on the current data world and his predictions for how data platforms will become easier to use. Sunil is optimistic about the use of virtual data for unifying the access of data for a variety of operational use cases. This includes dashboarding and ETL and all kinds of things. It was fun talking to Sunil and I enjoyed hearing about this concept of virtual data.

[SPONSOR MESSAGE]

[0:01:37.7] JM: When a rider calls a car using a ride sharing service, there are hundreds of back-end services involved in fulfilling that request. Distributed tracing allows the developers at the ride-sharing company to see how requests travel through all the stages of the network. From the front-end layer to the application middleware to the backend core data services, distributed tracing can be used to understand how long a complex request is taking at each of these stages, so the developers can debug their complex application and improve performance issues.

LightStep is a company built around distributed tracing and modern observability. LightStep answers questions and diagnoses anomalies in mobile applications, monoliths and microservices. At lightstep.com/sedaily, you can get started with LightStep tracing and get a free t-shirt. This comfortable well-fitting t-shirt says, “Distributed tracing is fun,” which is a quote that you may find yourself saying once you are improving the latency of your multi-service requests.

LightStep allows you to analyze every transaction that your users engage in. You can measure performance where it matters and you can find the root cause of your problems. LightStep was founded by Ben Sigelman who was a previous guest on Software Engineering Daily. In that show, he talked about his early development of distributed tracing at Google. I recommend going back and giving that episode a listen if you haven't heard it.

If you want to try distributed tracing for free, you can use LightStep and get a free t-shirt. Go to lightstep.com/sedaily. Companies such as Lyft, Twilio and github all use LightStep to observe their systems and improve their product quality.

Thanks to LightStep for being a sponsor of Software Engineering Daily. You can support the show by going to lightstep.com/sedaily.

[INTERVIEW]

[0:03:48.9] JM: Sunil Kamath, you are a Principal PM at Microsoft. Welcome to Software Engineering Daily.

[0:03:53.3] SK: Thank you, Jeff.

[0:03:54.3] JM: In this episode, I'd like to discuss the different waves of relational databases that we have gone through, as well as talking about some of the distributed data platforms that we're working with today. You have been in the industry for a pretty long time. You've been looking at relational databases for a long time. Take me back to the beginning of your career and describe the first wave of relational databases as you understand.

[0:04:20.4] SK: If I have to go back, I mean, obviously, I can't go back too much, because I entered the industry in the early 2000s, or 2000 is when I started my career in databases. During that time, distributed databases were flourishing and we had just entered the internet era, if you will.

Having said that, I mean, I got a glimpse of a lot of databases and how databases were built prior to these distributed systems, or distributed databases, particularly with lot of exposure around mainframe databases. Obviously, I mean, there were a lot of key learnings behind the type of applications that were running on the mainframe systems and then the type of workloads that were enabled by the internet era and requiring to process lots and lots of volumes of data. This one, I think I got the early taste of what does it take to actually build a very scalable and a performant relational database systems.

[0:05:25.4] JM: If we go all the way forward to the present and we think about relational databases in the past relative to the present, how do the workloads differ today relative to how they were in the earliest days?

[0:05:38.5] SK: It has changed a lot. I mean, and that's mostly a byproduct of the new ways in which data is now generated, as opposed to a few decades ago. As I was saying, I mean, databases were always asked to do more based on the amount and volume of data that have been generated. It was always the challenge of whether we have the databases to handle the volume of data, especially if you go back to the internet era and the digitization of ledgers and applications that started to now generate more and more data and that needed good performance, high-availability, because the databases started to be getting used in to many mission-critical applications.

Now we see that explosion of data and the types of problems we are forced to address is of even higher magnitude with machine generated data and data that is generated on social platforms born in the cloud applications, which now require a very different way in which you need to solve these complex database challenges.

Cloud is really helping to make it possible to run this massive scale of databases and empowering a whole new set of applications, which two decades ago we wouldn't have then thought of, or we wouldn't have even imagined that these applications would ever be built.

[0:07:04.5] JM: You alluded to the volume of data that first, we saw from the social networks and more recently, we've seen from, I guess you could call it IoT, which is sensors in agricultural fields, or factories, or self-driving cars. The volume of data has really changed how we need to approach things like buffering and batch versus streaming and basically, the ingestion portion of handling database systems. Tell me about how the volume of data has changed how we need to architect our data systems.

[0:07:42.4] SK: Yes, Jeff. That's a very interesting question. Before I answer that question, I mean, I think it'll be good for us to even time travel back. Like I was talking earlier about how the internet era actually propelled database applications and database workloads with the continuous need to process more data, that the processed data, and there was always more data to be processed.

That's where I believe, also we saw a shift in the database industry and also that drove the creation of what we call as the NoSQL databases. The pioneer of that, I have to recollect, started with Yahoo. I mean, a social media company had to process huge amounts of data and they did not have an off-the-shelf database that could go and support their workload needs, which gave rise to MapReduce and Hadoop systems.

The next follow-on to that was Google, which entered into the search engine market, again trying to mine all the data, tried analytics. They went ahead and created their own database to support the search engines. That actually propelled what we then fondly referred to as the NoSQL way. Now that was a very interesting way and we also saw birth of many, many NoSQL databases, each targeting a specific niche area based on the problem that they were trying to solve and application developers started feeling that hey, now they have access to some cool new NoSQL technologies that can solve all the sins of the relational database systems and maximize their productivity and create new applications.

The relational databases though was not sitting quite either. It evolved and has evolved, in fact a lot more quicker, if you will, starting with creating scale-out systems, which are your disk architecture, inspired from the IBM mainframe Sysplex, where you could just scale your database by adding more computers. Then leading into shared nothing architecture, which was essentially allowing workloads to shard that database across the different nodes and do distributed processing of queries.

All of these, actually one of the most important thing that happened is as much as the SQL workflows, or the relational databases lost its charm with this NoSQL workload, the big challenge the NoSQL workloads faced was a lack of ecosystem and a lack of language, where every NoSQL database came with their own language, which means the application developers have to go and learn and get a new language. More importantly, these did not come with the large ecosystem that relational database with SQL had already garnered.

That's what we see now is the consolidation of these different databases with around SQL as the language. now you see SQL enabled on top of Hadoop, you see SQL enabled on top of Kafka and many other NoSQL data stores as well. We see that now, more and more database systems with SQL as the language, as the dialect, to go and query their databases is really fueling the next wave of applications that we are going to be seen.

[SPONSOR MESSAGE]

[0:11:25.0] JM: Heroku is a large-scale cloud infrastructure provider. Yesterday's episode, we had John Daniel from Heroku on the show to talk about running cloud database workloads. John works on the relational database PostgreSQL at Heroku. PostgreSQL is often used for critical workloads.

It was quite an interesting show to hear about how Heroku manages these database workloads at scale. You can listen to it as well by listening to yesterday's episode. Heroku is a sponsor of Software Engineering Daily and I am a frequent user of Heroku, so I'm very happy to have Heroku as a sponsor.

I hope you check out that episode. You can also listen to our previous episodes with Heroku engineers. We did a show on Heroku Kafka. We've done a show about Heroku CI. You can check those out. Thank you to Heroku for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:12:34.6] JM: Unfortunately, SQL as a dialect is about the only consolidation we get in this world. On the front of queueing systems, I guess we've consolidated around Kafka. On the front of distributed stream processing, we're consolidating around Spark. On the front of which database to choose as the underlying OLAP data store, there seems not to be much consolidation. How do you view the rapid expanse of tools that people are using? Can you forecast any consolidation in the types of tools that we're using?

[0:13:14.9] SK: Like the way I see this is the – and what history has taught us that first of all, the consolidation has to happen around a language. It's very hard for programmers to learn new languages and new tools. That's where my second point about the ecosystem comes into play. Now with SQL as the language and the tools has enabled over decades, now are increasingly being brought into both the relational database world, as well as a non-relational database world.

While we see that eventually for – if you look at from enterprises and the businesses standpoint, it's less about data. It is more about what they can do with the data. How can they intelligence to the data, so that they can use the value of that data that they have stored in the database to drive the necessary business outcomes? Therefore, those type of application needs and those part and the pursuit of driving value from data and the data where it could be anywhere, or any systems and which is where data virtualization will start to become extremely prominent, where I see a world where – to data virtualization, no matter where the data lives, you should be able to run a SQL query against these disparate sources without having to replicate these data from one system to another system. In fact, I see that's the only possible way to actually solve the complex analytical business processing needs with a volume data that we continue to see and then continue to grow.

[0:15:03.3] JM: That is a really exciting and somewhat decisive answer. I'd like to unpack it. Could you define virtual data?

[0:15:12.8] SK: Yeah. I mean, virtual data is not like a virtual core, but data virtualization is really about in giving application a similar view, or being able to query using one SQL statement and automatically figuring around that to access this data and on these tables, it could be on my Oracle system, or in these tables and the subset of data could be on a SQL server and some others and PostgreSQL database.

From an application developer perspective, they don't need to care about where this data resides and which repository these data resides. I as a application developer, need to be able to write a SQL statement and everything is just taken care of me. The intelligent layer will automatically do the necessary mapping, distribute the data, do the distributed processing and get me all the results and satisfy the single query.

[0:16:10.7] JM: Have you looked at the Dremio Project at all?

[0:16:13.9] SK: I'm not aware of that.

[0:16:16.4] JM: Tomer Shiran who used to work at MapR, he started a company called Dremio, which was based on the Apache Drill Project. The Dremio product has become – it's basically all about this data virtualization subject. It's been really interesting doing some shows with him, because he talks about this problem where he have used data virtualization and data sharing within an organization, as I would say – not to take words out of his mouth, but very related problems. Because I guess, you may not have access to certain datasets within another of the organization. If you could somehow virtualize that data, or have access to a subset of it, or at least have a window into what else is in the organization, that presents useful features for the "data platform."

It reminds me of what you're saying here around like, "Okay, look. We've got these disparate tools, but if we are consolidating around SQL, we can implicitly consolidate around data virtualized SQL."

[0:17:28.1] SK: Correct. I mean, that's what I alluded to earlier on is that SQL is forcing that marriage of the different disparate data stores, or database systems. Instead of in the era we lived two decades ago where we used – where we all ended up building general-purpose, one size fit all type of databases, we have come to realization, the industry has come to realization that that's not possible anymore.

There are going to be new database tools that are going to be created and those that create these new database engines and tools and continue to use the SQL as the unified and consolidate around the SQL as a language, then by virtue of that, they get immediately, they get access to this humongous ecosystem that has already been created around SQL. That means you can continue to use what you used to before. That's the power of it.

Also, I would add that traditionally, I mean, like past database systems were evolving, you will also see that okay, I would have one great database for running my transaction system and then I would run with the ETL tools, or copy the data, or replicate the data at some other system, and that could be your enterprise data warehouse, it could be a datamart, and then you would have a lot of one – each datamart per business unit, and now you have to run your complex ETL logic data pipelines.

That is the fuel I believe is almost obsolete. In the world that we live in, where the data volume is so huge, it doesn't make practical sense, or our physics comes in a way of being able to replicate this – replicate the data, the velocity of the volume and the volume of data. You have to figure out ways in which we can do distributed data processing, in-line with where the data is, not ship the data from one – or replicate the data from one system to another.

[0:19:37.2] JM: There are definitely going to be people in the audience right now who are thinking, "Okay, this idea of virtual data is hard for me to comprehend, because if I'm –" Let's say I'm a top-level business user. I want to create a dashboard, or I just want to make a query for a data set. As a business user, I just know that data is being created, but in actuality, it's being streamed through Kafka, it's hitting perhaps at an OLTP database, it's getting dumped into a data lake with an ETL job. You've got these different temporal situations in each of these different materialized views along the data pipeline.

As a business user, I want to have insight into that, but maybe I also just want to get my data now. The notion of virtualized data is the idea that I should be able to have the virtual view into some idea of what is going on in my data, but I don't have access to all the data on the fly, at least immediately. Maybe I make a query, that query – the latency might depend on what parts of the data pipeline have to be queried to actually satisfy that query. I'm just trying to get a better understanding of what your view is for how this actually gets implemented.

[0:20:59.4] SK: Well, let me see. Suppose, say your Kafka is your real-time feed about the data is getting in. Now if I want to run a SQL query and I want to run – I want to create a dashboard, or what are those real-time events, as a user, I can't wait for that data to persist in some common database, like some tens of milliseconds later.

Now with the power of SQL, I will be able to query directly against that Kafka pipe. If I want to join that against any other data source that is living in another database, I should be able to run my sequel query that would span data sitting in Kafka and as well as another relational store, or perhaps maybe even another NoSQL data store that supports the SQL language, right?

As you want to build more real-time systems and that the latency matters, and you want to get fast insights as the data is getting ingested. Data virtualization and being able to process the data in-line, without having to wait for the copy of that data to land in a specific data source and then be able to query is no longer tolerable, right?

That's why building data platform is becomes extremely important, and building that data platform in an open way is even more important. That's what we and Microsoft are actually helping to build these systems of tomorrow. The data platform is not about a specific database engine anymore. It's not about a specific database product. It is really about how do you create a data platform acknowledging that data will be in different places for different reasons. It would perhaps be in data lake, for example, to store data longer term, and you want to do that in cheap storage.

Or it could be that you want your data be able to query on Kafka, or it would be available on Hadoop systems, or it could be on relational systems. You should be able to run, or create applications that regardless of the boundaries of that it is storing, create a unified data platform that allows you to support rich querying and also support real-time latencies that these applications, especially IoT type of applications demand.

[0:23:29.2] JM: Okay. Well, that's a really nice view, a nice perspective, a nice vision for the future. In order to get there, we will need to figure out some degree of instrumentation for these different data platforms. If I want to present this end-business user who's making a dashboard with the virtual data keys to the kingdom, then I'm going to need to instrument my Kafka cluster, I'm going to need to instrument my Cassandra, my PostgreSQL, all these different things are going to need some instrumentation to be able to communicate the virtualized data to the end-user, or maybe not. I mean, tell me about the instrumentation that needs to go into allowing this business user to have access to virtual data.

[0:24:13.9] SK: It would require a rich metadata framework, a rich metadata door that has the intelligence, that has the information on where to go find the data that is required to serve its particular purpose.

I'd also say that much of this is currently in the labs, where we are curating, where we are even figuring it out, in terms of what type of solutions make sense. Really here, it is first figuring out the what part of what the customer pain points and the businesses are trying to enable it for their users, in terms of applications that they need, in order to drive their business outcomes. The technology and the how piece has to quickly adapt and has to quickly evolve, so that we can really achieve what I'm talking about.

I wouldn't say that we are there yet, and I think, this is where I think key database research and investments need to happen in solving real-world problems. Definitely an area which by as a virtue of the limitation and the physics that comes in the way and an ever-increasing volume of data and the applications that people need to support going into the future, these are the – this is one of the areas in which how we would solve the needs of the future applications.

[0:25:42.2] JM: Okay. If we talk about the top-down perspective, the business user application perspective, what are the choke points today? I mean, you said we're not there yet. What are the kinds of business applications you're seeing people wanting to build today that they cannot quite build due to our immature data infrastructure?

[0:26:03.3] SK: You see, I mean, I think the core fundamental problem is that because of this organic evolution, yes, we all talk about, yes, there's a lot of data and so on and forth. What really businesses face, the number one problem they face is that this data is all over the place. The traditional and/or the older approaches, where vendors used to go and pitch enterprise data warehouse and centralize and bring all the data from all the disparate sources and put it into one gigantic big database was inefficient, or not really serving the needs. These happen to be multi-year project. Just to bring all data into one place, let alone, enabling applications.

That approach, like we know as the industry taught us is not working anymore. Now these days, nobody talks about enterprise data warehouse, or bringing all the data into a common repository to go run your analytical, or application workloads. Therefore, I mean, the first primary challenge as I said, to summarize, is really there's all this data, but now how do I how get all this data in one place? Number two, once I get data – once I get access, or once I get my hands to the data, how can I trust that the data is correct, or the data is accurate? What type of data smoothening, or data validity, or validity chart that needs to happen?

Then goes on to the question about how do we govern the data? What is the single source of truth? Because there might be different line of businesses within an enterprise might have a view of data that may not match, or that might be inconsistent. Having the ability to also then know what is the single source of truth, etc., that all – these are these are the common data challenges that businesses face today. Let alone, once you have done this is when you can really start monetizing the value of data to drive outcomes.

That is where the pendulum has to shift, because that's what the businesses need is not just data. They need to be able to monetize the value of that data by driving successful business outcomes. That's the problem, that's the landscape that customers face today.

The other dimension now is that this data could be stored on-premises, it could be on the cloud, it could be somewhere else, and it could also be spread across a lot of different clouds and data centers, which adds yet another degree of complexity. These customers have to go figure out what is the right approach, in terms of how they're going to solve this problem of data and being able to monetize the value from the data.

If you talk to the businesses, these are the challenges that they wrestle with. Clearly, there is not a single silver bullet today in industry that exists that says, that we can solve all of this problem. The industry is evolving and the technology is evolving fast, in terms of hybrid and all such good things, but I think there's still a lot of opportunity ahead in terms of really getting to the core of solving the problems that businesses face today in terms of bringing the value from the data industry.

[SPONSOR MESSAGE]

[0:29:37.5] JM: For almost 20 years, techmeme.com has been a common site that people in Silicon Valley go to for news. You might have heard of Techmeme. You might go there multiple times a day to catch up on what you missed. Mark Zuckerberg has reportedly been somebody who reads Techmeme on a regular basis.

Did you know that Techmeme has a podcast? It's called the Techmeme Ride Home and it's similar to Techmeme as a written content source. Techmeme catches you up on the latest news in the world of tech and the Techmem Ride Home Podcast does that every day at 5 p.m. Eastern in short 15 to 20-minute episodes.

The Techmeme Ride Home podcast tells you what happened in tech while you head home after a long day. It makes a great compliment to Software Engineering Daily, because most of our shows are pretty deep in the weeds, not really timely. We are pretty late to cover a lot of different topics. If you're looking for a source of tech news that is up-to-date, search your podcast app for Techmeme Ride Home, or just a ride home and subscribe to the Techmeme Ride Home podcast.

[INTERVIEW CONTINUED]

[0:31:09.2] JM: You're a Principal PM at Microsoft. I think of the Principal PM role, especially with somebody of your experience, probably you're spending a lot of time walking around, going to these different data teams. I mean, you've got the PostgreSQL team, the managed Kafka team, or event hub. Yeah, so you've got all these different teams that you need to communicate these different ideas around where data systems should be headed. Can you tell me a little bit about your work, like how do you manage your time, or what does a Principal PM and a cloud provider, where you really want to communicate your vision to all these different teams, what do you do?

[0:31:52.0] SK: A good portion of my time actually is understanding business problems and not be technology-driven. Although yes, technology is always very cool to work with. If it is not grounded in terms of what are the really the true business problems that our customers are trying to solve, then you ultimately end up solving problems that nobody else cares.

A good chunk of the time is also then as you, as to your point, is talking to all the different teams within Azure, and creating that unified database platform, enabling us to build on that vision of what I was referring to as a common database platform. It's not about that platform is a relational, or non-relational. It is a common database platform and making the appropriate integrations, and the adapters to integrate with Spark engines, or data warehousing, or NoSQL, or search, or pub/sub, or Kafka. Focusing on enabling the end-to-end scenario.

That's what the product management function typically focuses on is how do we bring all these disparate pieces of technologies and solutions and complete the scenario that adds business value to our customers.

[0:33:19.5] JM: Okay. Well, let's focus specifically on the database side of things, because in the world of databases, we have this term called NewSQL. NewSQL from my point of view, means something like your database is capable of doing both OLTP and OLAP queries. OLTP being your business logic. If I'm calling a ride-sharing service, like Uber, an OLTP database is handling my session and my user account and so on. This might be something like Mongo. An OLAP database is this thing for analytical processing, like rolling up all of the transactions that have taken place across a given time window. This is going to hit more entries. This might be ideally modeled as a columnar storage system.

We have a tension between the document to the relational model and the columnar storage model. That tension is trying to be addressed by some of these different NewSQL databases. Tell me how you see the space of NewSQL.

[0:34:32.1] SK: Yeah. I mean, it's a good terminology that's been introduced to call NewSQL. Essentially as we know, it's really the transition from NoSQL, to not only SQL to NewSQL, right? These are all the different names given, where it starts from like I hate SQL, to I might have to use SQL. Okay, now fully to accept knowledge in that yes, we eventually need SQL. Hence, the NewSQL, which is really trying to address the massive scale out, distributed scale out, your distributed databases, like Spanner from Google, or CosmosDB from our own offering, the examples of technologies that our products are available to go capture those extreme workload scenarios, where you want the power of SQL, you want very high availability, but you can sometimes sacrifice the consistency of the data, right?

I mean, at the end of the day, we are all bound by the same cap theorem that we all learned graduating from schools. In which says that you can leave how two and not all three. With some relaxed constraints, yes it enables classes of workloads that needs to go distribute across regions globally and scale very, very efficiently.

There is definitely – I mean, needs to be like through our own customers, like database. I mean, we see huge growth and adoption of that database and these type of solutions that are required for like say, IoT scenarios, or real-time applications workload scenarios. At the end of the day, I see that there is a market for everything, right? Because one of the biggest acknowledgment I think the industry has had is that there's no one fit for – one fit for all. No one database that fits all scenarios. There will be a need and a market for these different types of systems.
I think that's actually a good thing, which is really going to be the enabler of the type of applications that otherwise you would never think off. I really see the NewSQL is growing, but relational database market still continues to dominate overall database industry. Time will tell, but on my perspective, so long as products and technology solve business problems, I'm all-in. That's how I feel it should be.

[0:37:00.3] JM: Well, final question. Help me assess these NewSQL databases while we're on the subject, because when I started this podcast three and a half years ago, I thought it was complicated enough to try to assess Mongo versus Cassandra versus MySQL. Now I'm like, “Okay, I've got CosmosDB, I've got TieDB, I've got MemSQL, I've got just PostgreSQL, like the whole site is data vision for PostgreSQL thing. What are the axes of trade-offs that we're evaluating today? I mean, it's not even cap theorem anymore. Or maybe it is in some regards, but it feels more a developer experience, or what is it? What are we trading off against?

[0:37:52.3] SK: Well, it's trading off between performance, scalability, availability and the cost at which you can achieve one of the three, or two of the three, or all the three, right? That is where it becomes important, right? Where do you park some of these technologies and how do you put – just put it into its right use? We find that there are application workloads that are not necessarily driven by extreme performance, but they require skill. They want to operate within a particular envelope of the total cost of ownership.

Some are driven by the ecosystem. For example, with PostgreSQL, then its extensibility framework. Many developers are there for many more and that developers are therefore running behind PostgreSQL. That is the way that we are trying to capture and make sure that we are enabling our customers, the best of what PostgreSQL has to offer, alongside core differentiators, in terms of the scalability that can be achieved, or built on top of those PostgreSQL databases.

That was the key rationale behind Microsoft acquiring site as data earlier this year. We will now be able to open up new possibilities on PostgreSQL that was not possible before. That really is about if you want to broaden what your transactional database systems, or in real-time events and you want to run analytics at the same time on the same database without having to copy the data elsewhere, now you'll be able to do both and you can take advantage of the distributed processing, distributed database processing, so you can run a lot more efficiently.
Then for some IoT scenarios, you would find you would want to be able to run and ingest in microseconds, milliseconds. That's where technologies and solutions like CosmosDB come into play. I see that, I mean, that's the way developers and also businesses are going to do the trade-off between what they need and the application they're creating and what is the price point that they think about, that they can build their applications with.

Performance is primarily a driver of the business outcome that they want to deliver. Then they would pick a certain type of technology, even maybe including in-memory technologies. If scalability, or the volume of the data is important, then they would focus on the relational databases and then be able to do the hybrid, the transaction analytical processing on the same database. They can do that much more richly through relational databases today, than using NewSQL databases, or other stores.

That's how economics – I mean, at the end of the day, it's all about economics and application that they're trying to create, and a value that application will have to the business will drive the technology choices and the trade-offs they will make in terms of whether they want to go for performance, scalability, availability, one of the two, or all of the above.

[0:41:11.1] JM: Sunil Kamath, thanks for coming on the show. It's been fun talking to you.

[0:41:13.6] SK: Same here, Jeff. Thank you so much.

[END OF INTERVIEW]

[0:41:19.0] JM: This podcast is brought to you by wix.com. Build your website quickly with Wix. Wix code unites design features with advanced code capabilities, so you can build data-driven websites and professional web apps very quickly.

You can store and manage unlimited data. You can create hundreds of dynamic pages, you can add repeating layouts, make custom forms, call external APIs and take full control of your site's functionality using Wix code APIs and your own JavaScript. You don't need HTML or CSS.

With Wix code's built-in database and IDE, you've got one-click deployment that instantly updates all the content on your site. Everything is SEO-friendly. What about security and hosting and maintenance? Wix has you covered, so you can spend more time focusing on yourself and your clients.

If you're not a developer, it's not a problem. There is plenty that you can do without writing a line of code, although of course, if you are a developer then you can do much more. You can explore all the resources on the Wix code site to learn more about web development wherever you are in your developer career. You can discover video tutorials, articles, code snippets, API references and a lively forum where you can get advanced tips from Wix code experts.

Check it out for yourself at wix.com/sed. That's wix.com/sed. You can get 10% off your premium plan while developing a website quickly for the web. To get that 10% off the premium plan and support Software Engineering Daily, go to wix.com/sed and see what you could do with Wix code today.

[END]