# EPISODE 1046

[INTRODUCTION]

**[00:00:00] JM:** Data analysts need to collaborate with each other in the same way that software engineers do. They also need a high-quality development environment. These data analysts are not working with programming languages like Java and Python. So they're not using an IDE such as Eclipse. Data analysts predominantly use SQL and the tooling for a data analyst to work with SQL is often a SQL explorer tool, which lacks the kind of collaborative experience that we would expect in an age of Slack and GitHub and all kinds of other high-quality tools.

Rahil Sondhi is the creator of PopSQL, a collaborative SQL explorer. He created PopSQL after several years in the software industry including four years at Instacart. Rahil joins the show to talk about the frictions that data analysts encounter when working with databases and how those frictions led to the design of his company, PopSQL.

[SPONSOR MESSAGE]

**[00:01:01] JM:** I've recently started working with X-Team. X-Team is a company that can help you scale your team with new engineers. X-Team has been helping me out with softwaredaily.com and they have thousands of proven developers in over 50 countries ready to join your team and they can provide an immediate positive impact and lets you get back to focusing on what's most important, which is moving your team forward.

X-Team is able to support a wide range of needs. If you need DevOps, or mobile engineers, or backend architecture, or ecommerce, or frontend development, X-Team can help you with what you need. They've got a full-range of technologists who can help with AWS, and Go lang, and Shopify, and JavaScript, and Java. Whatever your engineering team needs to get to the points of scale that you want to get to, X-Team can help you grow your team. They offer flexible options if you're looking to grow your team efficiently, and their model allows for seamless integration with companies and teams of all sizes. Whether you're a gigantic company like Riot Games, or Coinbase, or Google, or if you're a tiny company like Software Daily. You can get help with the technologies that you need. If you're interested, you can go to x-team.com/sedaily.

That's x-team.com/sedaily to learn about getting some help with your engineering projects from X-Team.

Thank you to X-Team for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

**[00:01:01] JM:** Rahil, welcome to the show.

**[00:02:48] RS:** Thank you.

**[00:02:49] JM:** You're the founder of PopSQL, but before that, you were at Instacart for four years. Did your experience at Instacart inform the idea of PopSQL?

**[00:03:00] RS:** I mean, I've been writing SQL my whole career since like 2012, and I noticed that every time I would do an engineering project, I would want to back up the project with some data. Prepare some success metrics, some counter metrics, and I did this at pretty much every job including Instacart. Instacart was extremely data-driven, is still extremely data-driven. Yeah, for sure, a lot of the motivation from Instacart, and that's actually when I started working on it on the side.

**[00:03:34] JM:** It sounds like there were a few chronic problems that you noticed in the SQL user experience. I think you said something around metrics and something around saving your work. What were those key frustration points that you noticed in your work as somebody working with SQL a lot?

**[00:03:54] RS:** Yeah. I used to use SQL editors like let's say Sequel. It's Mac OS only, MySQL only, and it was great for the time that I was using it, but then I switched jobs and let's say we're no longer using MySQL. We're using Postgres now. I can't use Sequel Pro. At the same time, those editors only work with files on your computer. So if you want to share that SQL with somebody, you end up emailing a file around, or if you want to share the results, you end up exporting to CSV.

**[00:04:29] JM:** I want to understand the different ways in which SQL is used at a typical organization. I know there was a time in the past where there were these dedicated database administrators. I don't think that's really how SQL is used anymore. I think now you have people who are accessing SQL databases because they're doing transactional operations, like people that are building microservices that actually serve a business function. Then you have people who are using SQL as data scientists and they're maybe running these ad hoc queries to understand what's going on in an organization.

Then you have people who are maybe machine learning or data engineering people, and these are the people who are maybe writing OLAP jobs. They're writing these ETL jobs. Can you tell me about the different users of SQL infrastructure?

**[00:05:26] RS:** Yeah. Almost every role these days is writing SQL or wants to be writing SQL, and I think that's the macro trend that supports PopSQL. It's not just for data analysts. It's not just for data scientists. A product manager today wants to be SQL savvy. They want to be able to write SQL and get answers to data themselves without having to bug their analyst or have their request in a Jira queue for an analyst to answer. There's a use case for product managers. There's a use case for people on operations. Imagine you're at Instacart or DoorDash and you need to look up what's going on with some order or what are the – I don't know, like I'd say average wait times in a given region. You can do all that with SQL.

**[00:06:19] JM:** The people who are making OLTP transactional use cases, like the people that are operating with a microservices infrastructure, aren't they in many cases using a document database? Aren't they interacting with a Mongo database in many cases? Then if the company uses SQL infrastructure, the Mongo database is getting ETL'd into like a data warehouse. But at least on the frontend, on the microservices infrastructure, they're typically using like a Mongo database, right? That's not something that they're going to be accessing through a SQL editor.

**[00:07:04] RS:** We actually support Mongo. It's kind of like a Mongo shell. So you could write Mongo commands inside of PopSQL. So you're not writing pure SQL. You're writing Mongo commands. But, yeah, I guess the primary use case is for people writing SQL with Postgres or BigQuery, Snowflake.

**[00:07:24] JM:** For people who are creating queries, so some queries are just run once a day. Maybe you have these regular reporting jobs that get created. Then in other cases you have queries that are standing queries. You have queries that are repeatedly processed over incoming data. How do the use cases for standing queries differ from those of once that are run just once a day?

**[00:07:52] RS:** Yeah. I guess it depends on the author of the query, like if the author is a data engineer, that query might be written in – It's more like a software engineering process. So that might be written in their favorite code editor, let's say VS Code. It's checked into version control and it lives in some data engineering pipeline that they maintain, and that can be run hourly or daily, whatever their interval is.

The other use cases for people wanting to do analytics, so maybe this is once the data is already in the warehouse, somebody in customer success or somebody in operations or the product manager, this is the end user of the work that data engineering has done. The data engineering has spent all this time doing ETL, setting up the warehouse, and now the end user is some stakeholder in the company that's looking for insight, and they're writing their ad hoc queries to get answers.

**[00:07:52] JM:** tell me more about how SQL queries get reused between different areas of an organization. If I work at a thousand-person company, maybe I have – Like Instacart. If I have some query for all the orders that have been made across the company and I want to know how much money was spent in the total of all orders, that might be a query that multiple teams across the organization would want to use. How do those queries typically get shared between different areas or an organization?

**[00:09:31] RS:** Yeah, reusing queries is a topic. It's possible that people can be rewriting each other's queries or rewriting similar queries. That's actually what we want to help prevent. If everybody is using an offline SQL editor on their computer and editing queries on their local file system, then that's the worst possible scenario, right? Then no one can see what other people are writing. But if you have a collaborative environment and people are sharing links to queries and people can build off of each other's queries and people can comment on each other's queries, then all of a sudden you're leveraging what someone else has done in the past, or if you have a good search functionality and you can search for what people have written in the past, that would help too.

Other things that would help are like snippets. I write the same CTE pretty often, like if I want to determine if a given user is a customer, I need to join a few tables to get the answer. I end up copying and pasting the same CTE, and that's just because our data engineering is not that sophisticated at our company. That's an opportunity to perhaps do some data engineering and maybe create some fact table or – I mean, there are lots of solutions to this problem. Another solution could be snippets. Maybe I put that CTE, comment table expression, in a snippet and then other people who want to use the same CTE, they can just grab that snippet.

**[00:11:00] JM:** Okay. Great. We've talked about a few specific usability issues that people have in modern SQL. What are the other usability issues? I mean, we're going to get to eventually what you're building with PopSQL, but I just want to understand the usability problems in SQL management throughout an organization.

**[00:11:22] RS:** There's an issue. Even connecting to the database is difficult. I've heard stories from customers where they join the company and they spend like month just trying to get connected to the database because you have to find the right people to talk to. I don't know. This is probably at a bigger company where it would take a month, but yeah, just getting access to the database can be challenging.

Actually, I saw this problem at Instacart. A lot of people will get excited to try a new SQL solution and then they get presented with a database connection form and it's asking them for like the host and port and username and password and they're like, "Shit! I don't know what this is – Part of my language," but yeah, they say, "I don't know what this is," and they have to go talk to the database administrator and message them on Slack. Yeah, just getting connected to the database. Making that experience smooth can be a challenge.

Other usability problems with SQL, we talked about sharing or like composition of queries, sharing things like CTEs. I guess that's like a general problem with long queries. I guess there's visualization, like what you do after you've got your query. Maybe if you want to do a pivot table or make some kind of basic visualization, a line chart, a bar chart. You end up having to copy your results or let's say export your results to CSV, import into Google Sheets and then your pivot table there or do your bar chart there.

[SPONSOR MESSAGE]

**[00:13:04] JM:** If you're a Software Engineering Daily listener, most of the software that you know about, most of the infrastructure tools that you know about are probably things that are meant for developers. You may not have spent much time learning about business software. Well, when you start a business, you do eventually have to use business software. You have to find tools for finance, and inventory, and HR, and customer management. NetSuite is a suite of business tools that give you a full picture of your business all in one place. You can use it from your mobile phone. You can use it from your desktop computer, and NetSuite will give you the visibility and the control that you need to make decisions about your business.

If you want to take your company from $2 million in revenue to $10 million in revenue, or from $10 million to hundreds of millions of dollars, NetSuite can give you the tools to help charge your growth. Companies like Ring, Hint and Tecovas all use NetSuite. NetSuite is a popular cloud business system. It's trusted by more than 19,000 companies and you can schedule your free product tour right now. You can also get a free guide; Six Ways to Run a More Profitable Business, by going to netsuite.com/sedaily.

If you have started a business recently and you are looking for the right set of tools to manage that business, go to netsuite.com/sedaily and learn more about how NetSuite can help you with that business management process. That's netsuite.com/sedaily.

[INTERVIEW CONTINUED]

**[00:14:55] JM:** This kind of problem led you to building PopSQL, and PopSQL is kind of like a traditional SQL explorer or a database explorer that people might be familiar with. I think of these database explorers as in some ways similar to what a developer wants out of an IDE, out of an interactive development environment, like WebStorm or IntelliJ. You want this system that's going to be support you as a developer. There are different forms of IDEs. Like some people just use Notepad to write their code or they use some slightly sophisticated version of a notepad. Maybe the use of Vim, but there's a wide-range of different tools that people will use to assist them in the development of their code. I don't think we have the same range of tools, of rich tools, when it comes to database usability and database exploration. To me, that seems like one of the problems that you're addressing with PopSQL. How does the experience that you want to create with PopSQL compare to what a developer wants out of an IDE?

**[00:16:17] RS:** Yeah. I think there's a ton of overlap between what people want in a software engineering IDE and a data analysis IDE. For example, good autocomplete. In VS Code, depending if you're using a statically-typed or a dynamically-typed language, you get different set of suggestions. In a SQL editor you kind of want something similar. If I'm connected to Postgres database, can you recommend the Postgres functions? Maybe can you recommend the functions that I use often? It could be generated series or date-trunc extract, and then once I click that suggestion, can you help me fill out the arguments? Can give me hints on what each argument means? What order they need to be in?

I think Google Sheets does this really well. If you start typing a function, it helps you as you’re choosing the function. It gives you a hint about what each argument is. So the software engineering editors have that, and I think the SQL editors should have that too. That's autocomplete for one example. Others can be like limiting and error highlighting. If I make a mistake in my SQL, let's say I’m missing a comma or I've got the order of operations wrong. I always mix up order by N-limit. I put it in the wrong order sometimes. So if the editor can catch this for me and just fix it for me, that would be magic.

**[00:17:46] JM:** Okay. How do people actually use PopSQL in practice? What does the user experience look like?

**[00:17:59] RS:** We have a desktop app and a web apps. It’s cross-platform. If you’re using the desktop app, you can connect to a local database, let’s say on your computer. Really, you can connect to any database that your computer can reach. Even if you are connecting to like your RDS database and you need to be on VPN to access it, then that's a great candidate for the desktop app. As long as your computer is on the VPN, you can SSH into your batch in host, whatever you need to do. As long as your computer can reach the database, PopSQL can reach it as well. If you want to use the web app, then our servers need to be able to connect to your database, and you can do that through an SSH tunnel as well.

Once you're connected – We support all the major databases. So Snowflake, BigQuery, Red Shifts, MySQL, Postgres, even some non-relational databases, Cassandra, Presto. Once you're connected, you’re in an editor-like experience. So you can start writing SQL. You can name your queries just like a Google Doc. You can organize them in folders. You can get like an easy link to share your query and share your analysis with somebody else. You can build charts in the app. Yeah, I think that's a good high-level overview.

**[00:19:10] JM:** How much time have you spent in these different SQL explorer tools that have existed long before PopSQL? Do you take any inspiration or are there any critiques you have of these other SQL explorers that have led to how your designing PopSQL?

**[00:19:34] RS:** Let’s see. Some of them can be pretty clunky. They just feel like a Java application, like they might run slow or start slow. As soon as you start it, you’re presented with an install driver screen. There are some editors where I can use – I was like hard to figure out how to even get to the connection screen, like how do I add my database? How do I like just make a new query and get started?

Yeah. I mean, I’ve never been a big fan of some of the user experiences and the other editors at the speed at which they operate. Yeah, that was the inspiration for why I wanted to build PopSQL. I'm just building it the way – Or we're building it the way we think it should exist.

**[00:20:22] JM:** If I'm working a lot with SQL, I have a wide variety of queries. If I'm focused on SQL all day long, I probably got at least 10 or 15 different queries that I am running for various use cases. How do people working with SQL typically manage these different queries and what kinds of infrastructure have you built to try to make it easier to juggle all these different queries?

**[00:20:51] RS:** Some people have ephemeral queries. They might have a scratchpad query and they're just writing one-off things, like one-off answers that they don't intend to save forever or share with anybody. Just like a one-time thing. Yeah, that's one example, which is an ephemeral query.

There's others where you do really care about the query, and this might be like cohort analysis, and you spend hours writing this and you're going to look at it every week or every month, and you want that to be public within your team, and you want other people to be able to search for it and find it. You would put that in a shared directory in PopSQL. It's kind of like Google Drive. You might have your own private documents and you might have some public documents, and you can organize them in folders in PopSQL. You can do it by teams, sales, finance, customer success. You can do it by topic. For us, we have a folder all about teams. Then we have another folder all about subscriptions and billing. Then within that, there's different queries that are analyzing parts of that topic. Folders is the primary way that people can organize their work. Search is another big part of it. Sometimes you just don't know what folder to look in. So you just search, kind of like a Slack search.

**[00:22:20] JM:** What is the version control experience that we want out of queries? Because like I might have a query for all my users, and then I might slightly alter that query because I want to only get users that have logged in in the last five years. I'm not even considering these other users. Maybe I want this query to be completely updated or maybe I kind of want to fork the query and make a copy of it and make it slightly different. So I have a record of both of them. What do I want out of a version control experience for my different queries?

**[00:23:00] RS:** What we have right now is like built-in version control, revision history. Just like Google Docs, there is another use case where people want to use their git repo. Now, not every company has a git repo for their SQL queries. This is more of I guess advanced use case, and we plan on supporting that pretty soon.

In terms of like forking and changing parameters of your query, like I think you mentioned dates. We have something called query variables. If you had, let's say, select count from users that signed up after a certain date, you can just put in a variable for that date and then you could say – Like you get a form to enter the value for that variable, and that could be a week ago or a month ago or whatever you like. As you're changing the variable value, that does not change the underlying query. You can use a different value every day, but the version control for the query is not being changed.

**[00:24:06] JM:** What do you use to actually build the version control system? Did you build it on top of git or is the actually interfacing with git or did you make the version control system from scratch?

**[00:24:17] RS:** It's not using git. It’s like a Rails plug-in that does like audit tracking , like change tracking. So it’s something. Yeah, it’s not git under the hood.

**[00:24:29] JM:** Got it. Makes sense. Actually want talk a little bit more about the actual engineering of PopSQL itself, but first I want to go through a little bit more of what it actually does. We kind of danced around it a little bit, but it is an application that people actually download or you can access to your browser, and it’s got a bunch of features. So you can actually query your database in it. You can save your queries. You can version control your queries. You can try out different things, and there's also data visualization. So users can make dashboards in PopSQL. Now, there're lots of data visualization tools out there. Why would you want to build data visualization into a SQL editor?

**[00:25:19] RS:** I think it's a common part of the workflow. A lot of times you – If we’re talking about new user signups per day, once we have that in tabular, in a table format, it would be amazing to just quickly see that as a chart and just get a quick visualization. Now, in another SQL editor, you might have to export that to CSV, import it into Google Sheets, and then you've got like a static representation of – Your visualization is static to whenever you ran a query.

But if this is tied to your SQL editor and if it's actually tied to the database, then that link that represents the query and the table results and the visualization is always fresh. So you right it on Monday and a coworker opens it up on Wednesday and the coworker can see a fresh data.

**[00:26:12] JM:** Got it. Now, let's talk through actually building PopSQL, because we've talked through a variety the features. When you were building the first version of PopSQL, what was the infrastructure? Give me an overview of the architecture of PopSQL.

**[00:26:31] RS:** Sure. It's a Rails application on the backend. I chose GraphQL really early on. This is like 2016. I’m very happy with that choice. It’s just like grown in momentum since then. GraphQL has been amazing on the server side and on the client side. Postgres is the database.

On the frontend, it's React an electron. Electron lets us do the desktop app. It’s hosted on AWS. Yeah, that’s the stack.

**[00:27:03] JM:** GraphQL is used to interface between the frontend and the backend, and I guess if you issue a query on that electron frontend, the query hits your backend and then the query actually gets issued to the database. Tell me more about how like the connection between the frontend, like the Electron application or the desktop web application. How are the queries that the user is issuing to the SQL explorer interface? How are those queries getting routed to the actual database that the user is connected to? Does it have to go through your backend or does it go directly to their database?

**[00:27:50] RS:** It depends on what type of connection you have in PopSQL. There's different ways to configure this. Let's say by default you have a private connection and PopSQL, which means you're connecting directly from your computer to your database. If you're connecting to a Postgres database, it could be on your computer. It could be on the cloud. But it's your computer establishing the connection with the database. So all PopSQL server is doing in that scenario. We’re just like hosting the query statement, and we do that just to facilitate collaboration. But the database connection is between you and the database.

For other types of connections, we just ship to Snowflake and Octa. So you can use Octa to connect to Snowflake. Actually, if you want to share your connection with your team and make it easy for everybody on your team to connect so they don't have to bug you for database credentials, then we host the database credentials on our end. We encrypt them with AWS Key Management Service, and then it's our servers talking directly to your database.

In that scenario, when you execute a query in the React application, that query goes to our – That statement goes to our servers. Our servers establish the connection with your database. Execute the query, get the results and then send the results back to the client.

**[00:29:24] JM:** Got it. As you mentioned, you just created the support for Snowflake, and that brings to mind the fact that you have to write the ability to connect to so many different SQL interfaces. I mean, you got Postgres. You got the data warehouses as you just said. Earlier you mentioned that you have the ability to talk to Mongo databases. Is it difficult to write the necessary support to be able to talk to all these different databases?

**[00:29:58] RS:** Yeah. Yeah, it's a challenge especially because we have these two different ways of connecting. So we end up having to write Node.js connectors for the direct connections where it’s your computer talking directly to the database. Then we have to write Ruby connectors for these cloud connections that we call where our servers talk directly to your database. Eventually, we’d like to clean this up and have to just write them once. But for now, we’ve been able to get by.

**[00:30:28] JM:** Tell me more about the engineering difficulties of integrating with all these different database systems.

**[00:30:34] RS:** Yeah. There are different – Even for one database, there is different versions, right? Let’s say Postgres 9 to 12. Those are major updates and those are major changes, and maybe the query that we use to fetch your schema on Postgres 9 might be different than the query we use to fetch your schema in Postgrse 12. Maybe some things are stable, like get me a list of tables. Fine. Maybe that's the same. But maybe get me a list of materialized views. Maybe that changes. Maybe the query for that has changed. Yeah, even within one database, we have to support different versions, and that's a challenge.

Across databases, yeah, we really have to understand each database. How do you get the schema for Postgres versus the schema for Snowflake versus the schema for BigQuery? Then not all databases have the same features. Not everybody has materialized views. Not everybody has triggers. You really have to understand each database. I've only really used MySQL and Postgres through my career. If I have to write some Presto, if I’m writing like the Presto connector, I got to go learn Presto, or whoever is writing it.

**[00:30:34] JM:** What's the hardest engineering problem you've had to solve so far?

**[00:32:01] RS:** I mean, there's things that happen all the time. I think just past weekend, I had to figure out why certain large queries were just not processing. Just debugging that issue requires you to go into so many different systems. First, check the logs. Where is this query going? Fine. Maybe we’re receiving the query, like the execute query request. Are we able to establish the connection to the database? Can we even reach this database? Yeah. Okay, fine. We can reach it. Did the query run successfully against the database? Yeah, it looks like it ran. Okay, what happens after it's finished running? Well, we might've buffered it somewhere. Let's say we buffeted it in memcache.

Basically, I discovered that there's like a max item size parameter in memcache, and this query result was exceeding that parameter, and it was just like silently failing. So the results weren't being returned. I had to go in and change the max item size for memcache, and now like we support large query results. I also improve the user experience of all these. So if you do try and executor a large query, now at leaset we can let you know that, “Hey, this query is too big. Our servers can’t handle that.” Whatever it is, but what we've increase that limit. So you’ve very unlikely to even see that message.

Just debugging these distributed systems, making sure there is really good logging in place so that when somebody writes into support and gives you some vague explanation of what's going on just from their perspective, all they did was run in query and it never completed. That's all the information you have. Now, you have to go to debug a ton of systems and trace everything and see what went wrong, and you learn from these experiences. I’d become a much more a defensive engineer over time. I just assume everything is in her break. I'm going to have to debug everything at some point. That support request is going to come in and I’m going to have to trace everything. So the logging better be good. The error reporting better be good. Yeah, that's been a big lesson for me.

**[00:34:17] JM:** That example of having to debug problems that emerge when a customer issues a query and it just doesn’t work, and they don't know why. You’re going to have to dig in to the logs. Does that mean they you're logging everything that the users are doing, or do the users have to send you in their logs in that context?

**[00:34:44] RS:** Well, in this example, it was one of the cloud connections where our servers reached, like our servers are talking directly to your database. Some companies do this to make onboarding easier, right? The first person that signs up for PopSQL at your company sets up the connection, let's say, the Red Shift connection. Now, the next 10 people that sign up, they’re not greeted with this daunting connection form. They can just use the connection that the first person set up.

[SPONSOR MESSAGE]

**[00:35:20] JM:** DNS allows users to navigate to your web endpoints, and whether they're there hitting your app from their mobile phone or accessing your website from their browser, DNS is critical infrastructure for any piece of software.

F5 Cloud Services builds fast, reliable load-balancing and DNS services. For more than 20 years, F5 has been building load-balancing infrastructure, and today, F5 Cloud Services provides global DNS infrastructure for lightning fast access around the world. If you're looking for a scalable, high-quality DNS provider, visit f5.com/sedaily and get a free trial of F5 Cloud Services. Geolocation-based routing, health checking, DDoS protection and the stability and reliability of F5 Networks. Go to f5.com/sedaily for a free trial of F5 Cloud Services; fast, scalable DNS and load-balancing infrastructure. That's f5.com/sedaily.

[INTERVIEW CONTINUED]

**[00:36:30] JM:** What other kinds of infrastructure have put in place to help you debug problems that occur in the end-user situation?

**[00:36:39] RS:** Yeah, this is an ongoing project. Actually, right now, one of the biggest issues in this category is schema troubleshooting. People connect to their database. Maybe they're able to run their first query, but for some reason, their schema is not loading. There are a variety of reasons why that could happen. Maybe the tables are not showing up. Maybe some materialized views are not showing up. There's like a whole matrix of possibilities here. What databases are connected to what user you're using? What permissions does your user have? Does your user even have access to execute queries or to see the information schema or something?

What we're putting in place is some troubleshooting, some like self-diagnostics. For example, if you go to the Slack mobile app and you go to their settings menu and you go to help, they have a button to troubleshoot push notifications. You can bet that a ton of people have probably written into Slack support and said, “Hey, my push notifications don't work.” Then Slack engineering and support have to think of a solution, and they're like, “Okay, we need to put diagnostics in here to help people troubleshoot this issue themselves and to give Slack support better information.”

Same with us, we get a lot of people talking about the schema issues. We need to put in these diagnostics, and the diagnostic for us would be can your computer successfully reach the database? If you have an SSH tunnel, can you reach the SSH host? After that, does your user have the right permissions to execute the queries that we are executing to fetch the schema? To fetch all the tables, the columns the views, the triggers. We can show you the queries that we’re running to do that and we can show you the output.

All this information will help the user potentially fix their own problem. Maybe it's a permissions problem. Maybe they're not on VPN. Maybe they have the wrong port. This is all part of the user experiences. Just making these difficult technical problems like connecting to a database, making that really smooth for the end-user. When they need help, making it easy for our team to jump in and help.

**[00:38:59] JM:** We haven't talked too much about the actual collaboration experience, and I think one of the main focuses of you building PopSQL was this collaboration? Can you explain in more detail how engineers want to collaborate in a SQL editor like this?

**[00:39:22] RS:** The primary user for PopSQL is like a data analyst, and of course there's a ton of secondary roles. Like I mentioned, like product managers, people in operations, people in customer success, software engineers, data engineers. I guess it is a lot of people.

The way they collaborate, it starts off with a question. Someone has a question. Is our login form broken? If I want to assert that's true, then I'm going to write a SQL query and show you that like user sign-ups are down. It should be easy for me to get a link for that query and to share that with you, and so you can see the query I used it to get the answer. You can see the results. If there's a chart, you can see that. We think that that sharing experience, that collaboration experience, should be very seamless.

**[00:40:13] JM:** If you add those collaboration features, does it change how data teams actually work together?

**[00:40:20] RS:** I mean, if there will if there was not an easy way to collaborate like that, then we're going back many years. We're going back like 10 years when there were only offline SQL editors, and this is the pain that I was experiencing back then, is I was using one of those offline SQL editors and I was answering a lot of data questions and I had a folder on my desktop called SQL queries and I was literally exporting the query results to CSV and either sending the CSV over Hipchat back then, or email, or putting it in Google Drive, or importing it into a spreadsheet – Sorry, Google Sheet. That's just the static representation of the query and of the results at that time.

Now, if that coworker that you're sharing it with, if they want to look at updated data or if they want to change the query, change the date range like we were talking about before, they need to get that SQL, put it in their editor, they're going to be faced with this crazy connection form. It's just like a crazy back-and-forth that doesn't need to exist. It should be seamless.

**[00:41:28] JM:** What are the other collaboration features that you want to build in the near future? What are the problems in collaborating over SQL that are still preeminent?

**[00:41:39] RS:** We're pretty bullish on notebooks. Notebooks is a hot topic right now. Obviously, Jupyter is huge, but it's I don't think notebooks really exist for data analysts and just people that write SQL as the primary language. Jupyter is more for, let's' say, Python. I know there's SQL kernels in Jupyter, but I don't know. The user experience is not where we think it can be. So it goes back to that story of like I have a question. I answer it with SQL. I share the answer with somebody. They can see the SQL and the results. It's normally more than just one question and one query and one result. It's like a series of questions and a series of results. Sometimes I want to show the query. Sometimes I want to show the results.

Like on one page, I want to show the tabular results and I want to show a chart, and then I might want to maybe sample some of the rows. This ends up looking like a notebook, like a story, a narrative, and maybe I want some paragraphs explaining my methodology or explaining the conclusion. Yeah, we're pretty bullish on notebooks and we think the storytelling needs to be improved for SQL.

**[00:43:01] JM:** I'd love to talk a little bit about your general product development and especially how things have changed since the COVID 19 virus has become such a part of everybody's lives. Tell me where you're at product development-wise and what changes have been made to the company and your productivity since this virus has affected everybody?

**[00:43:29] RS:** We've been remote since the beginning, since about September 2019. Honestly, my day hasn't changed that much. The biggest differences I can't go to Phil's every day. Actually, they're open through mobile now, but yeah, we've all been remote since the beginning. It hasn't been a change for us. We're online on Slack. Some people are in different time zones. We've got like Pacific and Eastern. We Zoom a lot. Honestly, it hasn't been a big change for us.

**[00:44:02] JM:** Yeah, you're not alone. I've had a similar experience myself. Have you talked to any of your former colleagues from Instacart about how things are going there?

**[00:44:12] RS:** It's busy. It's certainly busy. I mean, they've had a lot of press recently, hiring 300,000 shoppers. It's hard to get Instacart delivery slot. It's hard to get – For the other grocery delivery companies, it's hard to get those delivery slots too. Yeah, it's a crazy time, and then they're doing a great service to the continent. We really need that service right.

**[00:44:35] JM:** Totally agree. Has as your personal productivity changed since the virus? Do you do feel like undercurrent of unease at all?

**[00:44:45] RS:** Not really. I mean, I'm watching the news a lot more I was never really into politics, but now I find myself watching the press briefings every day from the president and New York's governor. Yeah, I'm concerned. I'm concerned about like the whole world. But in terms of my productivity, it hasn't really been a change. I'm still working out the working out. I'm still working out at home.

**[00:45:13] JM:** Yeah. I guess you were not a gym person before.

**[00:45:17] RS:** Yeah. I was working out. I had a trainer before, but now the gym is closed and obviously I can meet the trainer. I just started doing it at home. Basically, I'm saying I didn't let the virus stop me from training.

**[00:45:30] JM:** Totally. what would you be working on if not PopSQL?

**[00:45:34] RS:** I'm really not sure. I don't think about that much. I've always wanted to be an entrepreneur. I started my first company when I was like 13-years-old selling Counterstrike game servers. I knew what I wanted from that young age, and coming to Silicon Valley was me trying to get the experience I knew I would need to start a company. Become a better engineer. See how companies like Instacart scale. Basically be a student in Silicon Valley, and then when I'm ready with my idea and the skills I think I'll need, then I can start. That's where I am now. Yeah, I don't think about other things.

**[00:46:15] JM:** Awesome. Rahil, thank you for coming on the show. It's been really great talking to you. Do you have anything to add about the future of PopSQL and where you're going with the company?

**[00:46:24] RS:** Yeah. I mean, we have a very ambitious roadmap ahead of us. We're trying to make the best editor for teams, best SQL editor for teams, and that starts by creating an amazing experience for an individual user and then making the collaboration story amazing. Yeah, we're just getting started and we're excited for what we have in the roadmap.

**[00:46:47] JM:** Awesome. Rahil, thanks for coming on the show. It's been great talking to you.

**[00:46:50] RS:** Thanks for having me, Jeff.

[END OF INTERVIEW]

**[00:47:00] JM:** When you need to focus on building software, you don't want to get bogged down by your database. MongoDB is an intuitive, flexible document database that lets you get to building. MongoDB's document model is a natural way to represent data so that you can focus on what matters. MongoDB Atlas is the best way to use MongoDB from the company that creates MongoDB. It's a global cloud database service that gives you all the developer productivity of MongoDB plus the added simplicity of a fully-managed database service. You can get started for free with MongoDB Atlas by going to mongodb.com/atlas.

Thank you to Mongo, DB, and you can get started for free by going to mongodb.com/atlas.

[END]