SDS PODCAST EPISODE 914: DATA LAKES 101 (AND WHY THEY'RE KEY FOR AI MODELS), WITH OZ KATZ

Jon Krohn:	00:00	Welcome to episode number 914 of the SuperDataScience Podcast. I'm your host, Jon Krohn. In today we've got Oz Katz, who's co-founder and CTO of the company lakeFS. lakeFS provides data storage for AI applications. And so we talk about key data related things for AI, like what's the difference between a data warehouse and a data lake? And what are the kinds of things that a modern database needs to be able to do in order to support training and deployment of AI applications? It's a invaluable episode. I think you're going to enjoy it.
	00:39	Oz Katz, welcome to the SuperDataScience podcast. It's great to have you here. Where are you calling in from today?
Oz Katz:	00:45	So I'm in New York and thank you, John for having me, by the way.
Jon Krohn:	00:49	As I'm usually in New York, that is the most likely place that you would find me. But today I'm in San Francisco and I suspect that you make your way out to the Bay Area from time to time as well. Oz?
Oz Katz:	01:01	I sometimes do. Yeah,
Oz Katz:	01:08	Are you there for a conference? Anything happening? That's interesting.
Jon Krohn:	01:12	I'm actually, I'm just on a stopover on my way to Brisbane, Australia.
Oz Katz:	01:17	Oh. Wow.
Oz Katz:	01:18	Okay. That's a
Jon Krohn:	01:18	Journey. Yeah. So that's coming up. After we finish recording, I make my way to the airport and yeah, I'm

going to have a 14 hour flight and yeah, I'm going to lose a day. It's kind of an interesting, I fly on a Wednesday night and then land it seems like much more than 14 hours later because you land on Friday, but it's even trippier. I think when I come back, I gain a day, so I'm in the air. Yeah,

Oz Katz: 01:48 You get that day back somehow, right?

Jon Krohn: 01:50 Yeah, exactly. So I think on the way back, I'm in the air for, again, 14 hours or something like that, but I land before I take off

Oz Katz: 02:00 Somehow. I always feel like I'm not smart enough to understand time zones fully. It just doesn't make sense to me.

Jon Krohn: 02:05 No, I know exactly. Sometimes I think when I'm wrestling with it that I'll, like, I should get a globe out the globes that we, I don't know, I had growing up as a child, the full size globe and just kind spin it and kind spend time thinking about how it's moving. It's probably possible with practice to figure it out. But yeah, it's definitely confusing.

Oz Katz: 02:24 So the trippiest thing about it that I've found so far, there's this thing called the international dateline, and you would expect it to be a straight line going down the ocean. It's definitely not a straight line. Some countries are like, some islands want to be on that side, some want to be on that other side. So there might be islands where it's like the next day, but they're actually further down into the other direction than other

Jon Krohn: 02:51 Places. You

Oz Katz: 02:53 Can just travel across days, one mile apart from each other.

Jon Krohn:	02:56	Yeah, that is trippy. It's all weird. Some countries might want to be kind of more aligned with Asia, I guess, and some more with the Americas, I guess, is the idea. Interesting. I hadn't thought about that. I'll have to look into it. But anyway, we are not here for a lesson on time zones today. We're here to talk about data for ai, and so specifically, I mean, we will eventually get to lakeFS, your company that you co-founded and we're your CTO. And so we will get there at some point in terms of talking about it technically and how it fits in as a solution and why it's so badly needed. But first, let's just talk about the name. What is lakeFS? What does that stand for?
Oz Katz:	03:42	Alright, so I'll preface and I'll say that as the old saying goes, there are two things that are hard in computer science. One is cash invalidation. The other one is naming things. I hope we got cash validation, at least that. So lakeFS stands for Lake as in data lake, which is typically the architecture where you would put lakeFS into FS stands for file system. I come from a systems engineering background. So for me, it makes perfect sense. I don't know that most of our target demographic or the people that get value from efis necessarily understand that.
Jon Krohn:	04:24	So is that named maybe by the tech team, not by the marketing team.
Oz Katz:	04:29	Yeah, that's what happened. We liked it enough. We decided to keep it around. People tend to gravitate towards the name, so I think we probably did okay there.
Jon Krohn:	04:38	Well, it's distinctive. It's distinctive looking because our audio only listeners won't be able to see this right now, but it's lowercase lake, LAKE, and then FS in all caps without a space. So it's kind of like visually, it's a little bit like iOS or Mac os. But yeah, I've never seen a name quite

like it, and that makes it memorable, which is a good thing,

Oz Katz: 05:01 I guess. So yeah,

Jon Krohn: 05:03 Expanding out, you mentioned there how lakeFs fits into a broader category of solutions called Data Lakes. Explain to us what a data lake is. I also hear terms data lakehouse and things like that, which that seems like a marketing gimmick to me, but maybe you can fill me in and give me the technical background on these different kinds of terms, like data lake, Lakehouse, that kind of thing.

Oz Katz: 05:25 Sure, and that's a great question. These are often misused or just used differently by different companies or vendors or people with some interest in the field. I'll leave my definition. It's not necessarily a W S'S definition or snowflake's definition, but I'll try to give mine. So data lake at its the most basic form of it would be just a shared folder, some central location where John, a member of one team and Oz, a member of some other team, can collaborate on top of data. Maybe you're on the marketing team and you have some data coming in from Salesforce or HubSpot and maybe I'm doing inventory for the company and I have something coming from my E-R-P-C-R-M, whatever it is. And we all want to be able to pull that data into one location. That's the lake. All the streams go into that lake,

Jon Krohn: 06:19 All the streams go into the lake. I think that's where it coming from.

Oz Katz: 06:24 I see, I see. And then we have one central location where now we can combine that data to make more out of it, one plus one equals three kind of thing. And once we have everything in one centralized location, all other teams can start using it, communicating with it, building models out

of it, do all the nice things that data lets us do. That's kind of the core

Jon Krohn: 06:48 Idea. Correct me if I'm wrong about this, but I think another feature that I have that I think about when I think about data lakes is that I think it maybe is even implied by what you just said, but it's that the data aren't necessarily very structured. It's kind of like it's a place where all those streams can flow in, and you don't necessarily worry about integrating those all into one kind of

Oz Katz: 07:12 Exactly. Yeah. So it's very much that, right? Maybe if I contrast it with what we typically had before, which is a data warehouse in a data warehouse, everything has to be well organized into tables that have this rigid structure behind them. Typically, there's a team in place that's kind of like the gatekeeper, if you will, because they're the only ones with the expertise that technology is kind of complex. It's also expensive. Whereas with that big shared folder, first of all, you throw your data in, if anyone wants to get access to it and wants to answer some business question, they can do it even if the data is not well formed, or even if it's not optimally designed to meet a specific query. So that's why I like the shared folder analogy. There is some inherent mess with that, but sometimes you want that mess because it allows you to move faster. We'll talk more about that mess later, I imagine.

Jon Krohn: 08:07 Okay, nice. Yeah, I mean maybe now is the time even to start getting into that. So in this era of ai, what are the particular kinds of needs that data scientists, AI engineers, people who are fine tuning, pre-training, running AI models, what are the particular needs that they have in terms of their data solution?

Oz Katz: 08:32 Oh, that's a good one because I think this is the answer I would give you two years ago is probably very different from the answer I'll give you now. So it used to be that the data that we get value from and that we can actually derive business impact out of was your, let's say, tabular data, right? Tables organized into a database. It's all very well structured, but that's not necessarily the case nowadays, especially with advances around AI models. What we can do with them, we can extract value from a lot of different other kinds of data as well. So it could be images, it could be embeddings out of those images, labels attached to them, all different kinds of modalities that are representative of the information we have at the company. So it's a very broad definition and it's no longer just tabular data or one specific kind.

Jon Krohn: 09:31 Wow, that's interesting to hear. And something that occurs to me is that we're in this era now where we have more and more data types that we are throwing into particularly multimodal large language models. They might have vision data, text data, they can be outputting different modalities of data, images, video, also text, and it seems to me like that could be a particularly tricky situation for data management. Is that correct? Am I correct that this is a tricky situation and how are we resolving it?

Oz Katz: 10:09 Yeah, it does make it a lot harder to manage, that's for sure. I think it's not just about the model itself being able to utilize or to output different modalities. It's also that our use cases around those models become more complex. So maybe I have a great model that's doing only text, but I have another model running just before it that I'm using as part of a pipeline that converts an image to text or that converts audio to text, and then the output of that might be converted into an image I'm generating. So it could be a mix of all of these things together. The place where this kind of gets difficult is that even though the

dream or the idea of a data lake is that everything streams into that one centralized location, reality is a bit more dirty than that. If I have images, I might have those images on my just a raw data lake just as image files, but I'm probably also going to have an embedding of those images toward in a vector database, and maybe I have some features that were extracted from those images in a database, maybe labels in some other third party labeling solution.

11:20 Now, instead of having that one source of truth that I was expecting to have, expect, I have multiple sources of lies, they get out of sync very easily. I don't necessarily have access to all the different modalities that were extracted from that one thing. It becomes a lot harder to manage. Right, right. Yeah, tons of

Jon Krohn: 11:39 Complexity. And so what are the kinds of solutions, what are you guys doing at lakeFs to make this new era that we're in with so many different types of data? Object stores, feature stores, the different, obviously data formats that we're talking about is inputs and outputs, different kinds of data pipelines for handling training of these models. Even inference time could potentially be challenging.

Oz Katz: 12:06 So one of the reasons why we built a fast and started working on this problem to begin with was to try to allow humans to better interact with that mess. And this comes up in several different ways. First, it's a very error prone environment. Maybe I'm okay, I looked at that images folder. I see a hundred images, great, I'll start working on my model to build something out of that. But here, John comes along from some other team and says, oh, okay, these images are not great. Let's replace a few of them and let's also add a few others to the mix. And he doesn't know that I'm working on some other thing that depends on those images. So now we have people stepping on each

other's toes, and then our CISO would come along and say, okay, I know that OZ needs access to those files, but he doesn't know that John also does and has a valid reason to do so, and now suddenly you're locked out.

13:00 And of course this multiplies by the amount of modalities that you have and the number of people in the organization. So part of the idea behind Lakes is essentially to create one facade, one way of working with the data across all these different modalities, and now we all speak the same language. So just like Git and GitHub, the same idea that they brought to code, if I want to introduce a change, I'll open a pull request. And John's team, they have to sign off on that change before it gets introduced. I have a very structured workflow to how changes get implemented to bring that same concept, that same notion to the data itself, not just the code, but also the data itself regardless of what modality it is or what type of actual business value represents.

Jon Krohn: 13:49 I gotcha. I gotcha. So when you talk about this unified facade that's like Git, is it literally GI where we type commands similar to the kinds of Git commands where you talked about pull requests there. So I guess we're making commits and that kind of thing, that same kind of paradigm.

Oz Katz: 14:05 So it is very similar in concept. You don't have to type in the commands if you don't want to use ACL I. Although one exists for you. And I know some people that's like they're religious about using their terminal for everything. I'm not one of those people. I like having an IUI. So if S also provides that at, but there's also a bunch of SDKs in place. If you have this daily job that scrapes the internet and brings in external data, you might want that system to run those commits for you. So we'd have a new commit arriving every day at some hour that represents new data coming in along with just like in GitHub where you have

actions and you have those green check boxes that say, okay, this was tested and is validated. You can do the same for that new that you just reduced. These are all images. They're all above a certain, I don't know, sharpness level all at least 500 pixels wide. You can run those checks automatically and then have the system commit and merge those changes for you. If you're a consumer of that data, you know that for sure. If I look at that directory, it's going to contain images that adhere to that level of quality that we set forth.

Jon Krohn: 15:15 Super cool. That sounds very helpful. It sounds like I could make use of that for sure. And so where do you see this all going? I mean, maybe that's something, maybe that's something even that's proprietary and maybe there's parts about that that are confidential roadmap. But to the extent that you can share, where are we going with data for AI models and what kinds of solutions might we need in the future?

Oz Katz: 15:42 So first of all, it's not confidential. We pride ourself of building leg effects out in the open and the core of Lego Fest, the versioning engine at all scales is fully open source as well. Oh really? It's so transparent that you can just go to GitHub and look at the code.

Jon Krohn: 15:57 I see.

Oz Katz: 15:59 So where do we see this going? So first of all, if I look at broader than just like a fest looking at the industry as a whole, one thing we are seeing happening is that everything tends to converge around the object store. It's kind of the source of truth that's emerging. We see S3 probably the most popular object store out there, release support for tabular data using Apache Iceberg recently, and also being a Vector database as well. All of this is from the last few months. So all of these modalities, instead of going to all these different locations, might be

in different technologies and different systems, but they all converge to the object storage, the underlying storage. So that's definitely something we're seeing and also something we're doubling down on. This is your organizational source of truth, and now we let you manage all the different types of data you might want to store there in one pane of glass, as they say. Right? One way to manage all of them

Jon Krohn: 16:59 Supporting vector databases like you mentioned there is, that's definitely something very important for our listeners who aren't aware, probably many of them are. The idea of a Vector database is that you have some numeric representation of the similarity between typically some large number of documents or files, and it allows you to retrieve similar information or some particular type of information very rapidly across millions, billions of documents in fractions of a second. So very powerful technology. Something of course important to be handling with lakeFS or any kind of data solution that people come up with today. What is Apache Iceberg? You mentioned that there as well, and that's one that I'm familiar with.

Oz Katz: 17:50 So Apache Iceberg is an open source project. It's been around about five, six years now. What it does is essentially lets you represent a table, just like a database table on top of an object store. So imagine you have a table on top of something like Amazon's S3 or whatever storage appliance you're using. Apache Iceberg will let you turn that into an actual table that has a schema that you can modify, you can insert, delete and update records on it, and it does so without actually forcing you to use one specific compute engine on top. Maybe you're using Pandas and I'm using AWS's Athena or Snowflake, but the data is still centralized in one place, and regardless of the tool that's consuming it, they all see the same state of the table, the same abstraction. Nice.

Jon Krohn:	18:44	Cool. That does sound important.
Oz Katz:	18:45	Think about the structured data story, those labels or the metadata about those images, they can also get stored back to the object store as an Apache iceberg table.
Jon Krohn:	18:56	Perfect. So just to bring the idea home of how a system like lakeFs works and allows us to manage data in particularly a large organization, it sounds like the bigger the organization gets, the more helpful having a tool like lakeFs in place is. Could you walk us through what it's like? If I'm a listener and I start using lakeFs at my organization, or I go to the open source repo, or maybe I need the commercial version, but for whatever reason I get lakeFs up and running, what is my experience like day-to-day as a user of lakeFs?
Oz Katz:	19:36	So imagine you had something like GitHub, but instead of showing you the code that you're using, you would see the data that you have in your organization. So I can say, okay, maybe I want to experiment. I want to build or optimize an existing model. I'll go ahead, I'll look at the data that I need and I'll branch out of it. I click the big green button on my screen, and now I have for all intents and purposes, my own isolated copy of everything and Fest does this intelligently without actually copying everything to another location. It does keep track of whatever changes you make on your branch. So now I have this entire big data lake that my organization has at my disposal. I can remove half of it to see what happens. I can add new stuff in to improve my results, and when I address that data from my code, I have to specify which version I'm using.
	20:33	So it's no longer just, here's the data lake and let's hope for the best. I'm saying it's this set of data at this version, this specific kind of snapshot frozen in time. What this guarantees is kind of a side effect is that whatever I'm

building now is going to be reproducible later. As long as my code doesn't introduce any variability into it. If it's deterministic, same code, same input data would guarantee the same result. So that's kind of step one. If I'm happy with the change, if I modify the data, introduce new data, remove some stuff, I now can either merge it back so everyone else gets a view of the nice changes I just introduced. Or as we mentioned, I can open a pool request, I can have those tests that automatically run to ensure the quality. I can do all that. And at the end, the idea is that now I'm treating the data not as just this random thing that flows in and out without anyone actually controlling it as something that's actually an asset of the company, which typically it is. And now it's just that you manage it that way. So all the power that GitHub gives you in terms of collaboration and manageability just for the data itself, that's kind of the

Jon Krohn: 21:48 Wow. Yeah, that was crystal clear and I'm really glad that I asked that question because you layered in this level of visualization things like pressing the big green button to get my own branch, and then now I have the whole organization's data lake at my disposal, but I also have the security, the peace of mind that I'm not going to mess up anyone else's workflow or anything else that anyone else is working on. I love it. It makes a lot of sense. Oz, thank you for taking the time today to explain how data are stored for modern AI systems, the kinds of things we need to be prepared for in the future. Before I let you go, I always ask my guests for a book recommendation.

Oz Katz: 22:28 That's a good one. So I'll give one of my favorites. I think the typical target audience for it is not usually data scientists, although I think they can benefit the most from it. It's called Designing Data Intensive Systems by Martin Clapman.

	22:47	It walks you through pretty much everything involved in actually building a data system. Think of how is a database actually, how does it work, what's an index? How does a database represent its actual data on the storage layer? What happens if two people try to write to a database at the same time? How does it manage that? All the basics of a large distributed system, but also a very small scale database gets covered by that book and it's very approachable, even though it's a very complex topic, complex, it's written in a way that's digestible that you can actually read.
Jon Krohn:	23:23	Perfect. And that's an O'Reilly book? I think it is,
Oz Katz:	23:26	Yeah.
Jon Krohn:	23:27	Yeah. I haven't actually read it, but I did buy it at some point and it's on my bookshelf. It ended up being one of those books where I haven't needed to get into the wes myself of designing a data intensive system. Someone else has always done it, but hopefully people working with me have benefited from that book being on the shelf.
Oz Katz:	23:44	And I think I recommend it because I think as a person that's consuming those systems as a customer of those systems, it really helps. Understanding at least one layer below where you are in the stack really helps.
Jon Krohn:	23:57	Perfect. Thanks for the recommendation. And then finally, this has been a really interesting episode. Have loved chatting with you. I'm sure lots of audience members enjoyed learning from you as well. What's the best way to follow you after the episode?
Oz Katz:	24:11	So I would say the Lake of Fest Slack is a great place not just to talk to me, but also with other people kind of in the same boat trying to solve those same challenges. I'm

		also available on LinkedIn and Twitter, so happy to leave links to that below as well.
Jon Krohn:	24:26	Nice. And the Lake Fs Slack. Yeah, that is easy to find. I just Googled it quickly, and so I'll have the link to that in the show notes. Thousands of people have already joined. I can see you right now. Indeed. Yeah. Nice. Alright, Oz, thanks so much for taking the time and catch you again soon. Yeah,
Oz Katz:	24:42	Thanks for having
Jon Krohn:	24:42	Me. What a nice, informative episode in it. Oz Katz covered how Data lakes provide the flexibility to throw in unstructured rivers of data from multiple sources without ridges schemas. Unlike traditional data warehouses that require careful organization upfront, he talked about how modern AI systems work with images and embeddings audio and text all at once, creating complex data management challenges where different modalities tend to get stored across multiple systems that quickly fall out of sync. And he talked about how his company lakeFs applies Git concepts to data management, allowing teams to branch entire data lakes, make isolated changes and merge improvements back through pull requests with automated quality checks. Alright, that's it. I hope you enjoyed today's episode. To be sure not to miss any of our exciting upcoming episodes, subscribe to this podcast if you aren't already. But most importantly, I hope you'll just keep on listening. Until next time, keep on rocking it out there and I'm looking forward to enjoying another round of the Super Data Science Podcast with you very soon.