📝 Notes

Jul 31, 2025

Meeting Jul 31, 2025 at 22:58 CST

Meeting records Transcript Recording

Summary

Yingjian Wu, Xuanwo Ding, and Renjie Liu discussed the term hierarchy in Iceberg Rust, agreeing that breaking changes are acceptable for alignment with the Iceberg Java specification, with Renjie Liu suggesting a new `term` be added. Yingjian Wu and Renjie Liu also discussed the implementation of a partition-aware writer in Iceberg Rust, confirming its necessity and clarifying its distinct functionality from the file writer. Additionally, Yingjian Wu, Xuanwo Ding, and Kurtis C. Wright discussed Iceberg Rust's integration with Pi Iceberg, with Xuanwo Ding noting that the decision for integration and adoption of Rust-powered components will be left to the Python community, and announced the successful 0.6.0 release of Iceberg Rust.

Details

- **Term Hierarchy in Iceberg Rust** Yingjian Wu raised a concern about aligning the `replace sort order` API in Iceberg Rust with the Iceberg Java specification, specifically regarding the input parameter being a `name` in Rust while Java allows a `term`. They noted that the naming conventions for `term` and `name reference` in Iceberg Rust differ from Iceberg Java, which could cause breaking changes if simply renamed (00:00:00). Xuanwo Ding confirmed that breaking changes are acceptable for alignment with the Java spec, provided they are well-documented and conveyed in change logs (00:01:27). Renjie Liu suggested adding a new `term` instead of renaming existing ones, proposing that Yingjian Wu submit a draft pull request to discuss the approach (00:02:45).
- **Partition-Aware Writer** Yingjian Wu inquired about the implementation of a partition-aware writer in Iceberg Rust, noting current discussions about whether it is necessary or if it would be handled by the repartition node of DataFusion

(00:04:41). Renjie Liu clarified that a partition writer is indeed needed and explained that it would be built upon the existing file writer, differentiating it by its requirement to know which partition a row belongs to, unlike the file writer which abstracts underlying formats (00:06:15). Renjie Liu further detailed two types of partition writers: a "fine" writer that keeps multiple writers open for different partitions and a "cluster" writer that assumes sorted input and only needs one writer at a time (00:09:11).

- **Pi Iceberg and Iceberg Rust Integration** Yingjian Wu asked for guidance on whether to implement features like the "delion vector" in Pi Iceberg or Iceberg Rust, given that Iceberg Rust is expected to have better engine integration and future binding with Pi Iceberg (00:10:40). Xuanwo Ding stated that Iceberg Rust aims to mature and compete with Java, but the decision for integration and adoption of Rust-powered components will be left to the Python community, allowing them to proceed at their own pace (00:12:08). Xuanwo Ding also inquired if there were benchmarks for the performance of Iceberg Rust's transform functionality (00:13:25).
- **0.6.0 Release and Community Engagement** Xuanwo Ding announced the successful 0.6.0 release of Iceberg Rust, highlighting key improvements such as Azure Data Lake support, enhanced valid data table support, and strengthened transaction support . Kurtis C. Wright expressed their intention to contribute more to the community and listened in to understand current pressing issues .

Suggested next steps

- [ ] Yingjian Wu will raise a draft pull request to show the idea of adding a new term.
- [ ] Renjie Liu will create issues for the partition writer.
- [ ] Xuanwo Ding will contact other people about the improvement in transform.

📖 Transcript

Jul 31, 2025

Meeting Jul 31, 2025 at 22:58 CST - Transcript

00:00:00

Renjie Liu: So, so, so what was your uh your your question?
Yingjian Wu: Yeah. So, I'm trying to find the issues I have. I actually asked it in the community. Oh, let me check. It's related to the um I think there is a issue related to the replace sort order act um API.
Renjie Liu: Okay.
Yingjian Wu: So right now I think the input is only a name but iceberg Java allows passing a term. Um so I think that issue was going like trying to fix that. Oh let me try to find that. Uh yes I think it's this one. Yeah. So I think the um so I just link the issue in the doc.
Xuanwo Ding: Okay. Right in
Yingjian Wu: Yeah. So I think right now I think it's I think iceberg ROS doesn't support like um bomb transform and unbomb transform and in iceberg java I think um like I think the naming is also a little bit different in iceberg rs and iceberg java is I think right now the term in iceberg rs is actually the name reference um in iceberg java.

00:01:27

Yingjian Wu: So my goal is basically to kind of implement all the term like the base class of terming iceberg java into iceberg rust but because the naming is already a little bit like confusing and wrong. Um I'm not sure like because if I just simply just rename it the same as what Java has I'm thinking will break the previous users who are using the library. Um so I just want to know what is the path forward here if I want to implement all the like the different term in ISO Java into Rust. Um I see.
Xuanwo Ding: Okay, I think doing some breaking change is okay if that's uh if the intention is to align with our Java spec. So this should be fine and we can do a break uh release.
Yingjian Wu: So even okay so even if I rename the existing like like strct name popstruct

name is okay
Xuanwo Ding: Uh so yeah that should be okay uh as long as we have so we have them documented and uh convey them clearly in our change locks.
Yingjian Wu: sounds good and yep go ahead

00:02:45

Renjie Liu: most uh so you are planning to uh you so you are planning to uh bring the the the term hierarchy into as rust Uh
Yingjian Wu: Yeah.
Renjie Liu: let me see. But do do you need to really uh actually redeem the uh transform to to term? I think you can add a new u more things like that. I I would haven't g s of thought about this but I I uh I think maybe uh adding a new new enough uh say uh term would be would be better. What do you think?
Yingjian Wu: I see. Yeah. Initially I was just thinking like make everything like the same as the iceberg Java. Um so it's like basically the same implementation as iceber Java. So that that was my goal. I haven't thought about like having another in um
Renjie Liu: I think uh currently Asper Ras transform is quite similar to asper as Java. So I I in fact I don't see the necessity to uh to to to to do the rename maybe adding a new new term is a bit I think um uh like um uh like what we what other uh contributors do you can say have a poor request per uh poor you can have uh draft per request to show your idea.

00:04:41

Renjie Liu: Uh so I I would uh maintainer will take a review and see uh and then we can decide the right direction uh and discuss it in the poll request. What do you think?
Yingjian Wu: sounds. Yeah, I think that would be great. Yeah.
Renjie Liu: Okay. Yeah. Thanks.
Yingjian Wu: Yeah. Um actually have another question but other people can go first if they have any.
Xuanwo Ding: seems okay. Go ahead.
Yingjian Wu: Okay. Yeah I think I also asked these questions in one of the issues. Um yeah it's related to the u right path. So I think I have I started some discussions around should we have a Yes.

Xuanwo Ding: Okay. Oh, sorry. Can you add in our box first so that we can keep a record?
Yingjian Wu: Okay. Let me I just link it. Um yeah so I think um in general as um I think we were talking about having a partition aware writer um just want to see what is the because I think there are some different opinions here um so

00:06:15

Xuanwo Ding: Okay. Hi there. Go ahead.
Yingjian Wu: I'm just wondering like what's the path forward here yeah like in the link we have some discussion about partition weird like file writer.
Renjie Liu: Oh, you mean you patina writer partition? Oh, wait.
Xuanwo Ding: Okay. Uh I don't know about the context a bit. Are you refering to that we should write partition clearly like create the directory in our partition writer or something like this.
Yingjian Wu: Oh, um I'm thinking about like the currently I think the file writer in iceberg rust it's not really partition aware so we don't really pass in the um partition Um and then I asked the questions like should we have a like similar to file writer we also have a partition file writer and then I think um some people were saying we don't really need a partition file writer because it will be implemented in the repartion node
Xuanwo Ding: Excellent.
Yingjian Wu: of data fusion but then like Okay.
Renjie Liu: And no, that's incorrect. It's a misunderstanding. I I think uh in fact we we still need need uh the partition writer.

00:07:45

Renjie Liu: But uh uh if you look at the right hierarchy uh it's a you you can see that in Java there is a partition writer right?
Yingjian Wu: Yeah.
Renjie Liu: Uh so we uh we we al I think we also need uh similar uh similar partition writer and it's built on the current uh existing file writer right because fire writer is used to say abstract out on the line format py or iro right and the partition writer is uh it uh it you can see the inter javas in face right uh when you write a row you would need to tell the writer the partition which the which partition the this row belongs to right it's a little

different from fire right we still uh we still need a one yeah no So,
Yingjian Wu: Mhm. I see. And also I haven't looked into this. Does the pocket writer also need to be partition aware? I don't think it's I don't think Oh, it doesn't. Okay, cool. Thank you. Um, Yep.
Renjie Liu: so you can uh you can see Java's partition writer implementation.

00:09:11

Renjie Liu: So essentially there are uh two kinds of partition writer right one is uh find out writer right fin means that it needs to keep in memory uh uh keep uh uh it needs to open several uh writers uh for different partitions. It needs to keep them open at the same time. Right?
Yingjian Wu: Yeah.
Renjie Liu: this uh and uh another is cluster writer run. It assumes that the input are already sorted by the partition. So it only needs to keep one fire writer uh uh for a partition each time, right?
Yingjian Wu: Sounds good. Um yeah. Um so if I want to contribute, do I normally just raise a PR or have have to like create an issue for that? Um
Renjie Liu: Uh let me see. I think we already have issue for uh I think we already uh have issue for I have we already have issued that I just I have uh created task in the in in the uh epic uh uh uh issue but I I didn't create issue let me create some issue for it uh Okay, I will it me uh so I if you want to I can to you if you want to

00:10:40

Yingjian Wu: Okay, sounds good.
Renjie Liu: you are interested in this part I can I can Oh okay thanks your your ID is uh your your ID still You're still right.
Yingjian Wu: Yeah, you can. Yeah, feel free to ask. Yeah, thank you. Let me see.
Xuanwo Ding: CTPI.
Yingjian Wu: Yes, Stevie.
Renjie Liu: Huh? What? Okay. Oh, I don't know why I can't I can send to you.
Xuanwo Ding: Oh, okay. Cool.
Renjie Liu: I just pin you there.
Yingjian Wu: sounds good. Um, I then have another general question. Oh, wait. Oh, yeah.

Okay, I'll just ask since I don't see any other questions. Um, so another question I have is I actually joined the PI iceberg meeting Tuesday and I think there's currently a like how to say people are trying to integrate PI iceberg with iceberg rust um in the future I think. So I'm currently looking at delion vector in pi iceberg but I think because iceberg rust has better integr will have better integration with the engine.

00:12:08

Yingjian Wu: So I'm just like thinking should we implement it in pi iceberg or should we implement it in iceberg rust and then in the future pisper rust can use just have a binding with iceberg rust. So I'm not sure like for iceberg rust community direction um like what's a path forward basically like integration the binding with pi iceberg. Yeah I'm just like a high level thoughts um is okay. Yeah.
Xuanwo Ding: Uh uh yeah bird is much mature than us. So uh so we won't try to just break them and we try to uh like replace head of the fireber and based on our needs. So we will try to uh like uh transform and f IO and uh maybe more things to come. Uh so ideally we want to do to do it at gradually uh pro progressively and uh uh we won't try to force push the fasting community to accept a uh rust powered core uh we want to uh by their own pace based on their own needs and they can decide uh whether or which part they want to use in Python.

00:13:25

Yingjian Wu: Nice.
Xuanwo Ding: Uh so uh so for our perspective we will try our best to make our uh rest implementation that mature and can compete with Java. Uh and uh from the other side we will let let the bite community to uh make the make the decision.
Yingjian Wu: Sounds good. Yeah, because I already see some like integration happening. Um I think the transform is one example.
Xuanwo Ding: Uh by the way, do we have some numbers about our how about how our transform works? Uh is is it much better or uh have we some benchmark for that?
Yingjian Wu: or I haven't I'm not aware of that.
Xuanwo Ding: Okay. Okay. Okay. Okay. Cool. Uh yeah I will contact with other uh pastor people to to say uh what was the improvement we have bring okay
Yingjian Wu: Okay, thank you.

Xuanwo Ding: hi Curtis do you have some question to ask I have seen you in our Okay.
Kurtis C. Wright: Um, no. I was just mostly coming to I'm going to be trying to help out more and so I was just wanted to listen in, see if there's any pressing issues and just be aware of what the where the community's headsp space is at basically. So, pleas Pleasure to meet everyone.
Xuanwo Ding: Cool. Yeah.
Renjie Liu: Okay, welcome.
Xuanwo Ding: Uh, okay. So question is ask and I will use some of time to uh celebrate our uh 0.6.0 zero release and uh uh in this release we uh we did some uh very cool things like uh aure aure data lake and uh uh let me check out and uh yeah we have released the aure data storage support and we also improved our valid data table support and we enhanced our transaction support. Uh so is a big step for our the building. Okay. So if we don't have any questions to go uh we are ending our this meeting. Thank you all for the coming.

Transcription ended after 00:16:38