6 Nov 2024

Agenda

- [David] DuckDB
- [David] General Updates
    - Validator
    - Consumer Testing
- [David] Core merging
- [Weston] WriteRel

Attendees

- Victor Barua
- Ashvin Agrawal
- Ian Cook
- David Sisson
- Weston Pace
- Jacques Nadeau

Notes

- [Jacques] DuckDB
    - The DuckDB folks are not planning on doing future work on the Substrait DuckDB extension. The repo is not easy to contribute to, so we're planning on pulling it into the substrait org.
    - [Jacques] The focus from Sundeck is on consuming Substrait.
    - [David] The DuckDB folks have cleaned up all their outstanding PR and merged them in.
    - [Victor] To confirm as well, the DuckDB folks are open to having us take the repo over.
    - [Ian] The DuckDB Substrait work was sponsored by Voltron Data. There was some behind the scenes work and tickets that we can port to the repo.
    - [Jacques] Aim is to resolve this this week. We would be transferring the repo to substrait, and then we would propose the PRs we have open against the new repo.
- [David] General Updates: Validator
    - There's been a lot of work, Wendell has made all the validator tests passed and it now includes all the latest Substrait stuff. They are working on cutting a release.
- [David] Consumer Testing
    - Also having CI issues. Ingo has a proposal about how to fix it.
    - [Weston] Splitting tests into CI test to make the testing infra works, and these tests are the actual consumer tests
- [David] Core merging

    - We've changed the policy of merging into core to hard-require 2 approvals. The expectations around when things should be merged into the core spec are not necessarily clear/known and are not enforced by CI.
    - [David] It would be nice to enforce the rules consistently.
    - [Victor] I have now required 2 approvals, and only maintainers/admins (i.e. SMC) can merge.
        - Followup work ticket: https://github.com/substrait-io/substrait/issues/732

Action Items

-

23 Oct 2024

Agenda

- [David] Updating the SMC Membership Guidelines
- [David] New committers
- [David] Plan anchors
- [Victor] DataFusion Seattle Field Report

Attendees

- Victor Barua
- Ashvin Agrawal
- Weston Pace
- David Sisson

Notes

- [David] SMC Updates
    - [David] I am now working at Sundeck, which means we have two SMC members there now (David, Jacques).
    - …
    - [David] Would it make sense to limit votes to 1 per org?
    - [Victor] Potentially something to look out for, but I trust David and Jacques overall and you haven't misused your power.
    - [Weston] (paraphrasing) It is nice to encode things in a way that doesn't rely on trust.
- [David] New committers
    - [Summary] There are 4+ committers that have unanimous votes.
- [David] Plan anchors
    - …
- [Victor] DataFusion Seattle Field Report
    - …

Action Items

- (Weston) Invite new committers

9 Oct 2024

Agenda

- [David] Partial Validator Update PR
    - Do we hold up releases until the validator is consistent with a recent Substrait?
- [David] What steps do we need to take in order to establish a “Substrait protocol”?

Attendees

- David Sisson
- Ashvin Agrawal
- Richard Tia
- Victor Barua
- Jacques Nadeau

Notes

- [David] Partial Validator Update PR
        - There is a PR out to update the validator.
    - [Victor] What’s the concern with releasing the validator with partial validations?
    - [David] It checks things up to like version 0.30.0 of the spec, but not things added since then.
    - [Victor] Are there any issues with only having partial validations? Does updating the spec to 0.57.1 indicate that it validates the spec to that version as well.
    - …
    - [Decision] We can merge and release. Partial validations are helpful, and we are only indicating that we support the *protos* up to that version.
- [David] What steps do we need to take in order to establish a “Substrait protocol”?
    - Substrait Protocol === Arrow Flight + Substrait
    - [David] One thing that would be useful would be to have some backend that does this. Experimenting with this might be the first step.
    - …
    - [Jacques] Is the goal to have an example similar to the Spark example so folks can see how it works?
    - [David] At some point it would be nice to have an example of a Substrait protocol where you can send a request to an endpoint and it runs.
    - …
    - [David] For the way folks would want to use it, I don’t think an in-memory engine (e.g DataFusion, DuckDb) is an interesting example. Want something that requires more scaling / multiple workers.
- Impact of Voltron Layoffs

        - Which repos are still under Voltron
            - Spark Repo
        - Is there any licensing we would need to deal with if we fork?
            - No, most of them are Apache licensed
            - [Jacques] There's a difference between saying we can use code, and you can own code.
            - [Jacques] If we wanted to contribute Substrait to Apache, we would need to be able to show / clear provenance.
            - [Jacques] The bar I would like to have in Substrait is less high. Effectively, no controversial forks / only friendly forks.
                - Example: The duckb substrait integration may become homeless.
- Order of Operations for Type Expression
    - https://github.com/substrait-io/substrait/issues/721
    -

Action Items

- Move ownership of YouTube account to Substrait org.
- Make forks of various Substrait related repos in Voltron.

25 Sept 2024

Agenda

- [Victor] SetRel Definition Updates
    - https://github.com/substrait-io/substrait/pull/708
- [Ingo] `-contrib` repositories (#707 and discussion on mailing list)
- [Ingo] Changes to `AggregateRel` (#706)
- [Victor] Table Functions

Attendees

- Victor Rea-Barua
- Ashvin Agrawal
- Ingo Muller
- David Sisson
- Joel Lubinitsky
- Weston Pace
- Jacques Nadeau

Notes

- [Victor] SetRel Definition Updates
    - [Victor] Existing not ALL become DISTINCT
    - [Victor] Should we match SQL in the spec for INTERSECT ALL and EXCEPT ALL?

        - [Jacques] There is a trick in that the SQL rules only apply to 2 operators, whereas can have multiple operators.
        - [Weston] The SQL definition is natural to extend.
        - …
        - [Decision] We will update INTERSECT ALL and EXCEPT ALL to match the SQL spec.
- [Ingo] `-contrib` repositories (#707 and discussion on mailing list)
    - People agree that the latest version of the PR looks good.
    - Next step: call for a vote on the mailing list.
- [Ingo] Changes to `AggregateRel` (#706)
    - The aggregate relation defines grouping sets. The same expression in different groupings sets should only be emitted once, but it's hard to define equality for expressions (at the protobuf level).
    - Proposal: limit grouping expressions to only be references.
    - People agree that the PR looks good.
    - Jacques wants to look at the commit message again.
- [Victor] Table Functions
    - Would it make sense to have table functions be similar to functions extensions?
    - [Jacques] I think there are broadly three types of table functions:
        - Table functions that have known output schemas. (i.e. range)
        - Table functions where input types can clarify output schemas. (i.e. unnest)
        - Table functions where only values define schemas.
            - Imagine a table function that is: run this query in Postgres. Input is a text string.
    - [Ingo] Some table functions also consume sets / relations.
    - [Ingo] There are named functions and embedded/embeddable (?) functions.
    - [Jacques] This may not be one thing or two. (Table functions that consume literal arguments vs table functions that consume relations).
- (Discussion on function testing as well as producer and consumer testing)
    - https://github.com/substrait-io/substrait/pull/680
    - https://github.com/substrait-io/consumer-testing
- [Victor] What motivating the changes for the define sideband optimization hints PR
    - [David] This is a more broadly applicable solution to https://github.com/substrait-io/substrait/pull/695

11 Sept 2024

Agenda

- [Victor] updating consumer-testing TCPH tests
- [Jacques] DuckDB Extension
- [Jacques] Parameterized Types
- [Victor] Promoting Extension Rels

- [David] PRs needing action
    - Name hint PR needs second review
    - Duplicate Eliminated Joins
    - JSON format
    - Test file format stalled?

Attendees

- Victor Rea-Barua
- Ashvin Agrawal
- Richard Tia
- David Sisson
- Weston Pace
- Jacques Nadeau

Notes

- [Victor] updating consumer-testing TCPH tests
    - [Victor] What do we need to do to regenerate them?
    - [Richard] There are some manual changes required after regeneration.
    - …
    - [Jacques] Could we submodule them into DataFusion? Another option would be to make release artifacts that's just those plans. A common problem is systems copying files over and then they become out of date.
    - [David] We don't really have a test system for producers. We do some limited testing for results, but we don't test generated plans across systems.
- [Jacques] DuckDB Extension
    - [Jacques] The DuckDB substrait extensions is currently in the DuckDB project. However they don't really take patches from external sources. Should we fork this to make it more open? I haven't talked to the DuckDB guys.
    - …
    - [David] We wouldn't make progress as fast as they are. Some things have been overnight, some things have taken longer.
    - [Weston] If we were to make a fork could we ask them to contribute to that?
    - [David] Their Substrait extensions are part of their release process. They do have a community extension program.
    - …
- [Jacques] Parameterized Types
    - [Jacques] Three Types of Types
        - Concrete Type: use in function declarations.
        - Parameterized Types: These are types that show up in function definitions. Things like `decimal<P,S>`
        - Type Programs / Type Expressions: expressions that return type expression
    - …

28 Aug 2024

Agenda

- [Victor] Substrait Contrib Organization
- Test File Format continued
- [David] Meeting Videos
- [David] substrait-validator is stale

Attendees

- Victor Rea-Barua
- Ashvin Agrawal
- David Sisson
- Richard Tia
- Weston Pace
- Ian Cook
- Jacques

Notes

- [Victor] Substrait Contrib Organization
    - Making a new organization for contributions.
    - [David] Main concern with having it in the core repo is reviews. If we don't have someone in charge of reviews then we won't get to it.
    - [Victor] I'm partial to having things in substrait-contrib to allow the people who are currently working on it to self-review.
    - … [conversation about what should and should not go in substrait-contrib]
    - [Jacques] [Specifically about substrait-mlir] This is something that could eventually live in core, but it needs a lot more resources. Development would be very dependent on Ingo for now.
    - …
    - [Jacques] General theme that I'm hearing. We would like to support the project more, it would be good to have some association, but we can't necessarily help with it directly.
    - [Jacques] Other question, does it make sense to have another org, or should we prefix / postfix the repo.
    - [Jacques] One argument against a separate org is discoverability.
    - [Weston] We could make a carve out in the guidelines for it. They should also agree to follow the code of conduct.
- Test File Format continued
    - [Jacques] Curious about the use-case for referencing tests.
    - [David] It's useful to be able to identify tests in the test suite that you want to ignore/skip.
    - [Jacques] The idea of defining explicit names for every tests seems painful.
    - [David] We don't necessarily need it per test, but groups would be useful.

    - [Victor] Another approach I've seen to this is hashing the test.
    - …
    - [Jacques] In the current (BFT) format, how do we reference tests?
    - [Richard] All the tests are named.
    - [Weston] It's also based on options.
    - …
- [David] Meeting Videos
- [David] substrait-validator is stale
    - [David] The validator is still useful to run to see if plans make sense. It doesn't have support for window functions.
    - [Jacques] Has anyone had a chance to look it it or work on it? I looked at it a while ago and it was fairly complicated (Rust aside).
    - [Weston] Part of it was the validator accepts invalid input which complicates it.
    - …
    - [Victor] One thing that could be interesting is wiring the validator into Datafusion's test suite.

14 Aug 2024

Agenda

- [David] BFT is now a part of Substrait
- [David] Test File Format
- ~~[David] Meeting Videos~~

Attendees

- Victor Rea-Barua
- Jacques Nadeau
- Richard Tia
- Asvhin Agrawal
- David Sisson

Notes

- [David] BFT is now a part of Substrait
    - [Jacques] Plan is to set all current Substrait contributors as contributors for the BFT.
    - The BFT website is not currently working
- [David] Test File Format
    - [Jacques] (paraphrasing) The initial experiment used a YAML format. Was hard to work with.
    - [Victor] I'm a fan. Reminds me of the sqllogictest format. Will make it easy to verify behaviors.
    - …

    - [Victor] One advantage of having the tests in the core repo is that we can add the tests in the same PR as the spec change. If we split the tests into their own rep (i.e. BFT) then we have to make two PRs.
    - [David] One potential issue is folks adding tests to the core spec, merging them in, and then breaking when they are pulled into the test framework.
        - [Victor] We already have a variant of this issue when we make extension changes that break in downstreams like substrait-java. IMO this is a CI problem.
    - …
    - [Jacques] Longer term, one of the things I'm thinking about is what is necessary for contributing to a function. I think ideally the requirement for contributing a function should be valid YAML and test cases.

Action Item

- Get the BFT website back up and running.
- Add CI check for YAML extensions being parsable.
- Add CI check for test cases executing correctly.

Jul 31, 2024

Agenda

- [Jacques] Function URIs and laziness 😛
- [Jacques] BFT Contribution Status
- [Jacques] IntervalCompound and precision
- [Jacques] substrait-cpp text format
- [Jacques] Substrait Compute Services
- [David] Are there any pending PRs needing pushing?

Attendees

- Victor Rea-Barua
- Jacques
- Weston P.
- Ingo Müller
- Richard Tia
- Ashvin Agrawal
- Ian Cook
- David Sisson

Notes

- [Jacques] Function URIs and laziness 😛
    - These descriptions seem like an alternative to options.
    - Perhaps just packaged versions of uris and options for dialects?

    - Implicit binding fails in the face of options though
    - Dialects are similar to server class such as a storage server, do we just need to have something with a list of URIs and options?
    - Q: should URIs include options in them?
    -
- Having a compute engine service allows more composability:
    - Ability to run two services with an A/B comparison
    - Performance measurement
    - Router/gateway to switch backends based on query complexity
    - (a lot of these we are working on for Spark Connect as part of the gateway work)
    -
    -

17 Jul 2024

Agenda

- [Ingo] Substrait dialect in MLIR
- [David] More PR roundup
    - feat: add arithmetic function with decimal type
        - (could use a second look)
    - feat: add CSV (text) file support
- [Victor] Intervals!
    - https://github.com/substrait-io/substrait/issues/664
    - https://github.com/substrait-io/substrait/pull/665

Attendees

Igno Muller, David Sisson, Weston Pace, Jacques Pienaar, Kevin Languasco, Victor Barua

Notes

- [Ingo] Substrait dialect in MLIR
    - Ingo is giving a presentation with slides.

3 Jul 2024

Agenda

- [David] PR roundup
  - feat: allow naming/aliasing relations
  - feat: add null input handling options for any_value
  - feat: define SetRel output nullability derivation
  - feat: add "initcap" function
- [Jacques] BFT contrib
- [Jacques] BFT Snowflake
- [Jacques] substrait-go Snowflake function implementations
- [Jacques] Tactical: CASE_INSENSITIVE_ASCII
- [Weston] Count: clarify null handling
- [David] More PR roundup
  - feat: add CSV (text) file support
- [David] News
  - Just released ibis-substrait v4.0.0 ! Supporting Ibis >= 9.x only and built against the new-and-improved upstream IR!

Attendees

- Jacques, Joel, Richard, David Sisson, Ingo Muller, Weston P, Victor Barua

Notes

- feat: allow naming/aliasing relations
  - [Victor] Not for this PR, but should we have hints at all. Couldn't they all be optimizations? Theoretically all hints could be extensions. (This was a discussion prompt
- feat: add null input handling options for any_value
  - [Weston] Skipping nulls is potentially more broadly applicable.
- feat: define SetRel output nullability derivation
  - …
  - [Jacques] Sets of problems
    - We weren't specifying the output.
    - There's a danger that we would pick the record type of the first input which is wrong?
    - Is it possible to have a less restrictive type? In practice what do systems do?
  - [Jacques] Intersect we could say: pick the first input or pick the most restrictive of inputs.
  - [Victor] If we allow choice I think that is fine. My issue with using the least-restrictive option for everything is that it limits producers to the lowest common denominator consumer (i.e. those that can't derive tight bounds).

    - [Jacques] We could potentially want options on the relation itself to distinguish these behaviours.
    - [Weston] This came back a while ago when talking about capabilities. There are some implementations that don't quite handle all edge cases.
    - …
    - [Jaques] For this particular PR I think we have two choices:
        - Move forward w/ the logical reasoning behind this.
        - Look through what systems/ecosystems and see what they do.
    - [Victor] This has come up in practice for us because Calcite and Substrait disagree on this, and I'm trying to improve this in both Substrait and Calcite.
    - …
    - [Joel] <mentioned casting as a way for consumers to force types>
        - The discussion pointed out that you can cast a nullable type to a non-null type which would allow consumers to deal with the most restrictive type.
- [Jacques] BFT contrib
    - Folks are generally in favour.
    - Potentially might want to change the name for something more descriptive.
- [Jacques] BFT Snowflake
    - [Jacques] Planning on contributing a Snowflake dialect to BFT. My assumption is that when you contribute new code the CI is running the testers. Potentially an issue running Snowflake in CI, thinking about doing this manually.
    - [Richard] I think that works. There are already some tests that need to be run manually.
    - [Jacques] The work that we're doing around this is for a Snowflake emulator.
- [Jacques] substrait-go Snowflake function implementations
    - [Jacques] Emulator we're working on is in Go. DuckDB is the kernel. Wrapping/enhancing it to look like Snowflake. Aim is to have no false positives. If something runs on the emulator we want to guarantee it runs on Snowflake.
    - [Jacques] We are thinking of contributing function *implementations* in Go. Ideally we would like to be able to implements this once and use in any language, we talked about WASM in the past <discussion about other portable languages> but we couldn't find anything that worked well.
    - [Weston] There has been a lot of ideas around universal function repositories. Spark has similar things were it has a lot of UDFs in a similar way. Having them in substrait-go could drive usage of Substrait in Go.
    - [Weston] These could also live in Arrow as kernels.
    - [Jacques] We're actually not focused on having this be vectorized / performant. Our focus is on emulation.
    - [Jacques] This could potentially also be in BFT?
    - [Victor] I could see this being kind of nice for building toy interpreters in Go using Substrait.
    - …

19 Jun 2024

Agenda

- [Weston] Alias relation PR
- [David] Gluten is looking to uplevel into Spark
    - Ideally the Substrait fork issue is fixed before this happens
- [David] Gluten code PR
    - Do we want scala in substrait-java?

Notes

- We should create a playlist of videos in the new Substrait meetings YT channel:
    - https://www.youtube.com/watch?v=YXnoBUQP258
- Alias relation PR
    - Do we want table alias somewhere?
    - How do we want to do it? Metadata advanced extension section perhaps?
    - OK for round trip information, Datafusion can fix self-join problems without a change to Substrait
    - Is this a key-map of metadata or is this a dedicated alias place?
    - Do the alias names need to be unique across the file?
    - Singular vs repeated advanced extension fields

5 Jun 2024

Agenda

- [Weston / Ingo] Can ReadRel::base_schema be a subset of the columns in the file?
- [Joel] State of message types such as type_expressions and function?
- [Richard] BFT donation to substrait project
- [David] Blog PR

Notes

- [Ashvin] Should we move the recordings to YT?
    - David to make Substrait account
- [Weston] Seems like ReadRel could add additional fields no problem
    - But after discussion, not as easy as that.
    - Field ids are useful for some systems
- [Joel]
    - Weston: Related to https://github.com/substrait-io/substrait/issues/193 perhaps?
    - Connecting these protos to YAML might help keep them in line
    - https://github.com/joellubi/bonobo
- Output schemas

8 May 2024

Agenda

- [Victor] Quick MATCH_RECOGNIZE update
- [David] Better error messages update
  Substrait: Adding Location to Error Messages
- [David] !~~ operator/function
- [Joel] Prescriptions for how simple extensions should be implemented/supported

Attendees

- Victor Barua
- David Sisson
- Weston P
- Richard Tia
- Joel Lubinitsky
- Ashvin Agrawal
- Ingo Müller

Notes

- [Victor] Quick MATCH_RECOGNIZE update
  - Just sketching out some stuff in public. Integrating with Calcite to catch issues and edge-cases.
  - Will make a more formal proposal when ready.
- Better error messages update
  - [David] Wrote up a doc around better error handling
    Substrait: Adding Location to Error Messages
  - [David] One thing that's up in the air is how to represent the locations.
  - …
  - [David] One advantage of a parsed location is that it could potentially provide more support for IDEs.
  - [Jacques] In the past found it useful to keep everything as structured as possible until the very end. The pattern typically was catching errors as they occurred and add more context as you unrolled. You can then do things like only show the last two layers of context. There are errors that are more directed towards users, and some that are more internal.
  - [Ingo] The comment that I made on this was that I know how this is done in MLIR, there are couple of classes of location like line number, and some like fused locations which are merged together, and some like call site locations.
  - [David] This one now has the capability to handle history. When locations are merged together it gets a bit more complicated.
- !~~ operator/function
  - [David] NOT LIKE operator. Postgres has it.
  - [Jaques] Other systems have it but not as special operators.

    - [David] DuckDB was interested in a dedicated NOT LIKE operator. It seems silly when we can just use NOT (LIKE …).
    - [Jacques] There could maybe be an issue around null handling.
    - [Victor] According to Postgres they are equivalent https://www.postgresql.org/docs/current/functions-matching.html
    - …
    - [Jacques] We are okay with having things that are duplicative if they have some inherent value. For example x IN (...) vs x = … OR x = … OR …
- Prescriptions for how simple extensions should be implemented/supported
    - [Joel] There have been a couple of discussions around how to handle URIs in extensions. I was thinking it would be useful to have some docs around common patterns for this. For example, it's clear that you need to namespace the definitions of functions with URI, name and signature.
    - …
    - [Joel] Ibis substrait runs into issue because it generates the mappings from the extension files, and multiple repeat definitions can override each other. Are there any prescriptions on whether this can be a pattern to follow?
    - …
    - [Jacques] There was one item which was if I want to build a substrait compatible thing what do I need. Another item was if I want to map functions from extensions how should that work.
    - [Jacques] For the first item we've talked about having groups of functionality that systems could support. It's a multi-dimensional problem because for different use cases you might have different capabilities. It's funny because I was recently watching a discussion on Parquet about this in which they were frustrated by a similar thing. We don't have a definition right now. I don't have a strong opinion about not being Substrait compatible.
    - …
    - [David] DuckDB is getting closer to supporting everything. Acero is a close 2nd. I was trying to figure out some of these groups after our discussion last time.
    - [Weston] SARGABLE could be on potential set of operations. … To your point Jacques there's a risk of making it harder for consumers, but would accelerate the growth of consumers. …
    - [David] Systems might want to know if you can handle subqueries or not subqueries.
    - [Joel] I think it's even a lower-level than what I'm thinking about. For example, addition is a simple capability. It makes sense to use it, but there's nothing telling producers and consumers to use them.
    - [Weston] You're talking about the case where the producer doesn't specify the url or there's a mismatch between the producer and the consumer.
    - [Joel] The concern of the producer is I want to be understood, and the producer is I want it to be clear.
    - [Victor] To echo your point, there's nothing in the spec about using the core extensions explicitly. Folks are free to make there own equivalent extension sets

and require those. It's a happy accident / evolutionary pressure that systems re-use the core extensions to be understood.
    - …
    - [Jaques] To the earlier point about sets of functionality, it also helps consumers because it gives them milestones to work towards. Possible dimensions: Function Sets, Relation Sets, Data Type Set. My main feeling is we shouldn't overthink this.
    - …
    - [Richard, via chat] This was generated from the consumer-testing repo (the results haven't been updated in a few months though): https://substrait-function-compatibility.streamlit.app/
    - …
    - [Weston] Capabilities could be something like a system returning a boolean field for each relation type to indicate where they support.
    - …
    - [Joel] I might have a use case. A query engine might want to know if its talking to dumb storage vs smart storage to know what it can push down.
    - …
    - <Discussing topics brought up in https://github.com/substrait-io/substrait/issues/634>
    -

24 Apr 2024

Agenda

- [David] Better error messages
- [David] Substrait consumers – survey of overall compliance
- [Victor] Splitting Isthmus into library and CLI components
    - context: https://github.com/substrait-io/substrait-java/issues/248
- [Victor] Combined Interval Type
    - skip
- [Victor] duplicate function definition issue
    - https://github.com/substrait-io/substrait/pull/631
- [Victor] dynamic fallback for unmapped functions in substrait-java

Attendees

- David Sisson
- Weston Pace
- Richard Tia
- Victor Rea-Barua
- Ashvin Agrawal
- Joel Lubinitsky

Notes

- [Better Error Messages]
    - Observation: All of the common stuff for relations goes in RelCommon but in Expression it goes in the Expression message. If we had put the RelCommon stuff in Relation we wouldn't have to worry about forgetting to include RelCommon in each Rel type.
    - [David] A lot of backends have been returning different error message like "# of fields is not right" or "this expression is not supported in this join". However, these errors don't indicate *where* in the plan this error is happening. Something I've noticed from Spark is that they have (arbitrary) plan ids which can be used to refer to relations. This can be useful for things like determining which relation produced which fields.
    - [David] Having plan ids on each relation would be helpful. Potentially useful would also be expression ids.
    - [Ashvin] JSON and XML do this by showing the path to the field.
    - [David] That would work, but the issue is that you have to keep track of the path.
    - [Weston] Would having a standard path language help consumers with this?
    - [David] There's something like this in protobuf when decoding a message *without* the definitions, where you can specify parts of the message based on accesses.
    - [Victor] The two options are effectively 1) add ids to various nodes which is easier for consumers to deal with in that they can just associated errors / information with ids 2) define a path language and implement support for it in the libraries, which would be more work for consumers potentially.
    - …
    - [Weston] I could go either way.
    - [David] It could be worth experimenting to see how much work tracking the path would be.
    - …
    - [David] With adding an extra field, there is some bloat and overhead as well. The other overhead I'm worried about is how hard it is to implement support in the consumers.
    - [Weston] Could also potentially be more work for producers which may need to keep track of ids to ensure uniqueness.
    - …
    - [Victor] Would the ids be required?
    - [David] No, they're for debugging purposes only.
    - [Weston] What if we stated that the ID was the depth-first position of the relation in the plan? There would be no explicit labels. Tools dumping plans for debugging would generate ids on the way out.
    - [Weston] Another option would be to make the ids optional.
    - [Victor] I think we're all in agreement that better error messages would be good.
    - [David] The question then is more how we want to approach this.
- [Substrait Consumers]

    - [David] I've been trying to figure out which consumers have full support for Substrait, but so far it's zero. DuckDB doesn't have subquery support. DataFusion doesn't have subquery support and also other small issues. Acero doesn't support subqueries. Velox is the next one to look into.
    - [Weston] It's typically unusual for consumers to support subqueries. They are usually re-written by the planner. When you say no consumer has full-support for Substrait, my reaction is that no consumer will have full support for Substrait. Most systems will pick some subset.
    - …
    - [Weston] Some systems have planners that eliminate subqueries. DuckDB potentially could flatten subqueries in Substrait plans, but this possibly happens *before* substrait conversion in the normal planner.
    - [Weston] Acero for example has hand-flattened TPCH queries.
    - …
    - [Weston] Maybe it's time to start putting names on our compatibility levels.
    - …
    - [Joel] I think that leaning into partial support for Substrait would be useful, as it reduces the barrier for implementations. Some of the challenges here are negotiations of capabilities for systems. How can you find out ahead of time what capabilities a system has.
    - …
    - [Victor] Support for substrait also goes beyond just which relation and which functions, but also are you supporting all the options associated with the function definitions.
- [Splitting Isthmus into library and CLI components]
    - [Victor] Mostly giving a heads-up.
    - [David] This change is also potentially needed as well because the CLI is pulling in a concrete logging implementation.
    - Victor Barua
- [duplicate function definition issue]
    - context: https://github.com/substrait-io/substrait/pull/631
    - [Victor] Should we pull out the most recently added function, or keep the newer one for consistency.
    - [Weston] Potentially only Acero cares about this.
- [dynamic fallback for unmapped functions in substrait-java]
    - TLDR: we have a system in place that allows us to handle functions that don't / aren't mapped to Calcite functions by generating a function object in Calcite base on the substrait definition (which has all the argument and return type information needed). This is useful in cases were there is not currently a Substrait to Calcite mapping but we would just like it to work, or we have a user-defined function that we just want to pass through Calcite to one of our underlying systems.
    - [General Discussion] This seems like something that could be useful.

10 Apr 2024

Agenda

- [Victor] Untyped Null Literals
- [Victor] JSON Type
- [David] Delimiter joins

Attendees

- David Sisson
- Victor Rea-Barua
- Asvhin Agrawal
- Gil Forsyth
- Richard
- Weston P
- Joel Lubinitsky
- Jacques

Notes

- [Untyped Null Literals]
    - …
    - [Jacques] Untyped nulls are in SQL, you can do comparisons.
    - [Victor] Our issue is that our consumer is better positioned to handle untyped nulls than our producer.
    - [Weston] One thing that Substrait provides is that every node of our expression tree has a type.
    - [Jacques] Even if you had an untyped null in the function, you know the type from the function.
    - …
    - [Jacques] If you have a untyped null in the plan, then it propagates through the plan. You would have a whole subtree that is untyped.
    - [Weston] It might be nice to see concrete places where humans put a null into a plan. What would that use case look like?
    - …
    - [Victor] One example of this is “SELECT null”. This null doesn’t technically have a type associated with it. This is a valid query, but not necessarily a useful one.
    - [Joel] Does Substrait have a hierarchy of types?
    - [Jacques] No
    - [Joel] This feels like it would be something like a bottom type.
    - …
    - [David] Is null even a type?
    - [Weston] In some cases it is a type. One example of this is parsing a CSV where the whole column is null.

    - [Jacques] In some cases null is just used to indicate the value of a type.
    - …
    - [Jacques] Going back to your coalesce example, even if we had a null type I would argue that the call `coalesce(int, int, nullType)` is not a valid call.
    - [Jacques] Maybe we can do something like you can have a null type but you can only use it where you would use `any`.
- [JSON Type]
    - [Jacques] For JSON I would argue that we should do what Snowflake does. Go look at variant.
    - [Victor] Context, we're looking at adding various JSON functions into our system, and some of these functions along with a type could make sense to live in the upstream.
    - [Jacques] Do systems even have JSON types?
    - [Victor] Postgres is one system that has it (JSON, JSONB)
    - [Joel] BigQuery has a JSON type.
    - [Weston] One question: There is adding it as a type, and then adding support for unstructured data. So for example something like `metadata.width` where `metadata` is a JSON type.
    - [David] The alternative to doing this would be adding a JSON as a type and then using functions.
    - [Jacques] …
    - [Weston] Why does Substrait need to know anything about types? 1, it defines a valid set of values. 2, it defines what functions are allowed on it. 3, what is the representation.
    - [David] A lot of people don't bother looking at the JSON values / validating it.
    - …
    - [Jacques] One of the things that we're trying to do. There are different systems that have names for different things that are the same thing. We have some systems that have a json types, some systems use variant. Should we have representations for all of these, or can we collapse them down.
    - …
    - [Weston] Let's say we have json_extract_path(json) -> json which is supported by Postgres and DuckDB, and then we have something like json_extract_path(varchar) -> varchar for (i.e SqlServer). Are you saying we should have a higher-level type to encapsulate all of these.
    - [Jacques] The question I'm asking is if there is a higher-level concept for extracting fields out of semi-structured data. Another example of this is BSON (from Mongo) which has more types than JSON.
    - [Jacques] There might be different names for these functions across these different types with the same functionality.
    - [Weston] We get requests for this as well, somewhat of this is storage-engine focused. A lot of users have no schemas. Things like Mongo which are truly schemaless, versus a lot of systems will use advanced schema evolution for this (i.e. the first time you insert a column you add a type for it). For truly schemaless,

    - one way to look at this, if you have a truly unstructured column and you want to extract a field is that a function that extracts something like a varchar or should it return an unknown type?
    - [Jacques] In Snowflake, you can traverse and if you ask for a width field (which is like json_extract_path) you would get back another variant value, which you can cast or trycast. It has a type, but it's something you have to interrogate.
    - [Victor] This is a lot of food for thought for me to digest.
    - [Jacques] There's a project called Malloy. It's various things, one of which is a semantic layer. One of the ideas behind it is that you shouldn't have normalized large data. Instead you store all of your data in a protobuf structure and then you do large scans and perform local aggregates. …
- [Data Council Live Report]
    - [Weston] I was at Data Council and I heard a bunch of people mention Substrait. I asked around for why people weren't using it. Some people were looking for an optimizer. Some people were interested in solving dialect issues. It was positive to me to get it mentioned, but mostly in a "we see a lot of possibility".
    - [Joel] The most interaction that people were having with Substrait was via Data Fusion. I have a thought of something that could be useful. One thing that I've worked with that has been tricky is ODBC. By adapting Flight SQL and Substrait together you could try and solve the ODBC problem forever.
    - [Jacques] It's actually part of the spec.
    - [David] DuckDB is one place that tries to support it, but right now it throws an error due to encoding issues.
    - …
    - [Joel] <brought up the concept of middleware as a space for Substrait, as something that exists *between* producers and consumers>
- [Delimited Joins]
    - What is a delimited join?
    - [Gil] https://btw-2015.informatik.uni-hamburg.de/res/proceedings/Hauptband/Wiss/Neumann-Unnesting_Arbitrary_Querie.pdf

27 Mar 2024

Agenda

-

Attendees

- Victor Rea-Barua
- Ashvin Agrawal

- Guillaume Massé
- Ingo Müller
- David Sisson
- Jacques
- David Sisson
- Ian Cook
- Gil Forsyth

Notes

- [David] There are a number of stalled PRs in the Substrait repo, we should be looking to close those out.
- [David] The Spark Substrait Gateway is making great progress, already handles most SparkConnect operations except joins (and all SQL using the DuckDB dialect)
    - show_string required a subplan with 11 relations to implement
    - Needed a builder, will be moving that into substrait-python
        - Keeping track of the used functions is one advantage of the builder
        - Reducing some of the boilerplate was also useful
    - Will be moving intro voltrondata's organization before it finds its way into ours

13 Mar 2024

Agenda

- [Victor] Support for nanoseconds in Time and IntervalDayToSecond
- [Jacques] Golang status update and known usage

Attendees

- Victor Rea-Barua
- Asvhin Agrawal
- David Sisson
- Weston P
- Jacques
- Joel Lubinitsky

Notes

- [Victor] Support for nanoseconds in Time and IntervalDayToSecond
    - [Victor] Arbitrary precision time. Yay or nay?
    - [Overall] Yay
    - [Victor] Adding nanoseconds to IntervalDayToString
    - [David] Has anyone else added it?
    - [Jacques] These are conceptually weird because some systems allow you to go from Months to Days but others don't.

    - …
    - Snowflake support this.
    - [Jacques] For a system that supports multiple precisions, something that's painful is adding intervals with different precisions. If you take an interval of precision 9 and add it to a timestamp of precision 6, what's the precision of the output? Substrait may not have opinions about this, but we want to make it clear what the adding precision rules would be.
    - [Jacques] For consistency reasons, we could have a precision interval, backed by the IntervalDayToSecond literal.
    - [Victor/Summarizing] Add an IntervalDayToSecondWithPrecision backed by a literal with fields for the various precisions (i.e. days, hours, seconds, millis, micro, nano, …).
    - [Joel] Asking for clarification about earlier point about Substrait not having opinions about adding intervals.
    - [Jacques] …
    - [Jacques] How does this work for timestamp with precision now?
    - [Weston] …
    - [Weston] As long as we don't put anything in the YAML, we don't have any opinions. But as soon as we put them in the YAML we're staking out an opinion.
    - [Jacques/paraphrase] If we define the functions adding interval and timestamp to only allow the same precision, we can avoid some ambiguity.
    - [Joel] I'm asking because there is a desire to have consistency between engines.
    - [Jacques] There's a distinction in my head between clear definition and preferred definition. It's possible that some people want to upcast when adding a precision 9 interval to a precision 6 interval, but others may want rounding. But as to whether there is one that's more blessed by Substrait…
    - [Joel] …
    - [Weston] My preference would be that systems reject functions they don't implement if they don't match the behavior as specified.
    - [Jacques] Engines may consume plans in a lossy manner in some cases. Example of an engine that takes adds float but returns double.
    - [Victor] IMO that engine should reject add:fp32_fp32
    - [Jacques] …
    - …
    - [Weston] <talking about BFT>
    - …

28 Feb 2024

Agenda

- [Victor] Set Operations
    - Clarify language around Primary and Multiset Operations: https://substrait.io/relations/logical_relations/#set-operation-types

    - Missing support for `EXCEPT DISTINCT` and `INTERSECT DISTINCT`
        - mysql
            - INTERSECT: https://dev.mysql.com/doc/refman/8.0/en/intersect.html
            - EXCEPT: https://dev.mysql.com/doc/refman/8.0/en/except.html
        - Postgres: https://www.postgresql.org/docs/current/queries-union.html
- [David] spark-substrait-gateway – is this something we want under the substrait group?
    - File formats – Spark and Substrait have a disparate set of formats:
      parquet, orc, text, json, csv, avro for Spark
      parquet, orc, dwrf, arrow, extensions for Substrait
      Should we add more standard formats?
- [Victor] Wildcard type matching. `any` vs `any1`, `any2`, `any3`
- [Victor] Quick Temperature Check: Remove Jabel from substrait-java
- [David] Any thoughts on the semantic layer question on Slack this week? cube.dev

Attendees

- Victor Rea-Barua
- David Sisson
- Weston Pace
- Jacques
- Joel Lubinitsky

Notes

- Set Op Discussion
    - [Victor] The difference between primary and multiset operations is a little unclear, though reading it just now it makes more sense.
    - …
    - [David] Runtime examples could be useful here.
- Why is there no minus distinct or intersection distinct?
    - [Jacques] [paraphrase] In practice you can perform minus and intersection and then apply a distinct after. Union Distinct is common enough that we included it.
    - [Victor] Wouldn't it make sense to include a distinct version of these operators for optimization/performance (to avoid performing the set op and then the distinct as separate stages).
    - [Jacques] That would be a physical operation.
- Spark Substrait Gateway
    - [David] You can connect to it with PySpark and it emits Substrait. You can use this instead of Spark, to plug in Substrait consumers.
    - [Weston] What if I don't want to use GRPC?
    - [David] If you had the spark plan and want to convert it to Substrait directly you could just use the library.
    - [Jacques] I assume you have to swallow the JVM?
    - [David] This actually doesn't have any JVM. It's just Python.

    - [Victor] To summarize this is two things:
        - A Python library to convert Spark plans to Substrait plans.
        - A service that wraps the library.
    - [Ashvin] Once we add optimization how is this different from Gluten?
    - [David] Gluten is much lower level, it emits multiple Substrait plans. This is much higher level and emits a single Substrait plan.
    - [David] Should we move this into the Substrait Org?
    - [Jacques] Question: We have three different ways of converting from Spark to Substrait
        - This
        - The gluten project
        - Another project from some folks who came to meetings prior.
    - [Jacques] What would be good would be to clarify what each of these would be for.
    - [Joel] I second that point. I've been trying to understand the gaps between consumers and producers and had to do my investigations. It would be good to have a summary for the producers and consumers for Substrait.
    - [Jacques] The site right now is more about people working on Substrait, not people using Substrait.
    - **Decision**: Makes sense to bring it in.
        - [Weston] I can do reviews.
- [David] Separate formats support for Spark and Substrait
    - [David] Should I just go and add more support for local files?
    - [Victor] What does it mean to add support?
    - [David] Add an option to the oneof for file_format in LocalFiles.
    - [Weston] Quoting Jacques (from before) the original intent was to have that be an extension point instead of explicit options.
    - [David] I will see about adding CSV support.
- [Victor] Wildcard type matching
    - Question: do all `any` have to have the same type?
    - [Jacques] No. They do not need to be consistent.
    - [Victor] Write Clarification for This
- [Victor] Removing Jabel
    - [Victor] <Explaining problems with Jabel>
    - [Jacques] To clarify what Jabel does. It doesn't modify the bytecode, it only applies to syntax features. There is apparently a flag to enable a flag to compile new features to old flags. It effectively is something like: if there is a feature that could be compiled to Java 8, then compile it down.
    - [Jacques] The flipside of this question is to drop Java 8 support?
    - <Chatting about why Spark use Java>
- [David] semantic layer question in Slack
    - [Jacques] … I feel like semantic layers are still very early. There's lot of tools to specify them, but no generic specification. Another way to think about it, there

could be a tool that looks at Substrait plan and determines whether it's a good plan (i.e. your joining to tables but not using the field)
    - [Joel] are these similar to views?
    - [Jacques] Semantics layers are basically the cube definition without the data. Defining that "hey you can roll this up by the categories". A view is a semantic layer, but a semantic layer can do more than a view.
- [Victor] Can you express recursive CTEs?
    - [David] You can reference another relation by number (ReferenceRel).
    - [Jacques] I don't think we can support this as is. The result of the query is fed back into itself and terminates.

14 Feb 2024

Agenda

- [Victor] Purpose/usage of struct type in user-defined types
    - Slack Thread
    - Docs
- [Victor] Governance Question. Clarification of approvals required for *library* modifications
    - https://substrait.io/governance/#substrait-voting-process

Notes

- Purpose/usage of struct type in user-defined types
    - [Weston] The structure can be used to define literals.
    - [Victor] So the literal has to conform to that struct.
    - [Weston] I'm thinking of systems that don't have knowledge would only …
    - [Victor] The language around moves and copies is confusing because I would expect it to be possible to move and copy a protobuf Any around without restrictions.
    - [Weston] Right now we don't make any assumptions about the representation of the data.
    - [Victor] If there is a struct defined, do literals need to conform to that struct?
    - …
    - [Victor] So it sounds like if there is a struct defined, instead of sending a literal as protobuf ANY it could be sent as a message of the struct.
    - [Weston] When it comes to user-defined types, the thing that Substrait cares about are what functions are defined on those structs, and what are the semantics.
    - [Weston] If there are struct representations of the UDT, they could be nested.
    - [Jacques] The inspiration for this is akin to the logical and physical pattern in Parquet. There are several types in Parquet that are composite types. The way the Big Query? paper defined struct there was no way to define nullability, so in

Parquet there is a physical layer around the struct. So for example, there is a int96 time in Impala that not all systems could work with it.
    - [Jaques] Use case, if you come up with a really useful type, you could use it in systems that can't handle.
    - [Weston] As an example, if we have two systems that can communicate Arrow with an extension type. If the Substrait extension type information has enough to tell you about the internal storage type then we could use that in both systems even if they don't know about the type.
    - [Victor] If we declare a type representation is that binding for literal representation.
    - [Victor] Question around the wording of
        - The structure field is optional. If not specified, the type class is considered to be fully opaque. This implies that a systems without built-in support for the type cannot manipulate values in any way, including moving and cloning. This may be useful for exotic, context-sensitive types, such as raw pointers or identifiers that cannot be cloned.
    - [Action Item] Victor to make PR.
- Governance Question. Clarification of approvals required for *library* modifications
    - [Action Item] Victor to make PR to add language around library modifications specifically.
    - [Weston] Can you review yourself, or should we require another reviewer.
    - [Jacques] We advertise as review then commit.
    - [Weston] This could potentially matter for folks starting new projects within Substrait.
- Weston at Iceberg Meeting
    - [Weston] The Iceberg view philosophy, we have lots of different backends which speak SQL variants. They prefer not to emit *different* dialects. They have been exploring Substrait and Coral.
    - …
    - [Weston] Most people had legacy stuff in Hive that they are migrating to Trino. The usecase for views was to get consistency between Trino and Hive.
    - …
    - [Jacques] Was there active work around this?
    - [Weston] The sense I got was that people were enjoying debating this as a solution to the problem, but not actually working on it.

31 Jan 2024

Agenda

- [Weston] Parameterized timestamp type
- [Jacques] BFT
- [David] Text format
- [Richard] Blogs/Posts

Notes

- Re: Parameterized timestamp type
    - There were asks on Slack for nanosecond timestamp or, even better, a parameterized timestamp.
    - Richard created a PR (mentioned in call that nanosecond timestamp is important for voltron as well, not just responding to slack)
    - Question about the range
        - If engines are storing timestamps as 8-byte nanoseconds then they can't meet the range we document today
    - [Jacques] Do we need to specify the range? What's the downside if we do not?
        - [Weston] Functions will be more "vague". You won't know for certain that a plan running on one engine will run on another.
        - [Weston] Pragmatically, it might not matter. Engines aren't respecting this range today. We don't document what happens when the range is exceeded.
- Substrait blog
    - Looking into Medium as a potential way to author blog
    - SMC will need to make sure that the shared publication doesn't turn into a single vendor blog

17 Jan 2024

Agenda

- TBD

Notes

- TBD

3 Jan 2024

Agenda

- [Weston] Null awareness discussion continued
    - https://groups.google.com/g/substrait/c/AWFHB8Tvb2M/m/h1eAji-FAAAJ
    -

Notes

- Null Awareness

    - [Weston] I have sent out an email about this, and have done some more investigation since then to confirm engine behaviors. Turns out there are some strange corner cases when you try to decorrelate subqueries, and also some weirdness around sets.
        - Decorrelating Subqueries: Use Posgres as ground truth the behavior is different between NOT IN and WHERE NOT EXISTS. Velox introduced a flag to differentiate between these two behaviors. As best as I can tell not other engine uses a flag like this. DuckDB use a mark join instead of an ant-join. DataFusion emits wrong results in some cases. Spark gives you the right answer, but they use a hash join in one case but for the other (WHERE NOT EXIST) they use a sort-merge join.
        - Set Difference: You can use an anti-join to implement SET DIFFERENCE. Use an anti-join to give me all the the results that are in table A but not in table B. DataFusion and Acero have a nulls_equals_null flag which is only used in the anti-join.
    - [Jacques] The way we handled in Dremio is we had a … our hash joining would be able to accept expression what where either EQUAL or IS NOT DISTINCT FROM
        - [Weston] IS NOT DISTINCT from solves it for the SET DIFFERENCE, not the decorrelated subqueries. This also only works for logical plans as well.
        - [Jacques] In Dremio we allowed limited expressions in physical joins (i.e. equals, IS NOT DISTINCT FROM, etc)
    - [Jacques] Could we solve this with some like (left, right, null_aware_flag)
        - [Weston] That solves the IS NOT DISTINCT FROM problem but not the decorrelated subquery problem.

20 Dec 2023

Agenda

- [David] Items close to finished:
  https://github.com/substrait-io/substrait/pull/572
  https://github.com/substrait-io/substrait/pull/580
- https://cwiki.apache.org/confluence/display/INCUBATOR/GlutenProposal
-

Notes

- Null aware discussion
    - https://github.com/substrait-io/substrait/issues/325
    - https://github.com/substrait-io/substrait/pull/467
    -
-

6 Dec 2023

Agenda

- Spark update? -
- Extended expressions for Substrait Java (review, comments, questions?)
- Context functions, any new thoughts?
- Join Output Types
- Release process
- Victor's very specify gripe about Calcite Aggregations not allowing anything other than references in AggregateCalls

Attendees

- Victor Barua
- Dane Pitkin
- David Sisson
- David Susanibar
- Gil Forsythe
- Jacques Nadeau

Notes

- Spark update
    - [Jacques] I feel like there was good momentum around spark, but I haven't heard any updates. Not much progresss. Gluten good progress on Gluten but they are working mostly independent of Substrait core. Upstream the logical plan stuff has stalled. Jacques will try to reach out to people engaged in the logical plan upstreaming previously.
- Extended Expressions
    - [David Susanibar] [paraphrase] This will be useful as part of arrow.
    - [Victor] I'm planning on looking at these PRs today.
    - [Jacques] It would be good to have some kind of blog post or documentation around this.
- Join Output Types
    - [Victor] Context: Joins currently specify the output record shape based on the concatenation of the left record and right record. Depending on the join type, the nullability of the records can change.
    - [Gil] This comes up in Ibis land because people ask why a column is nullable when the table isn't. Nullability is a property of the relation.
    - [Jaques] Aggregations can also affect nullability. Calcite tracks nullabiilty. Does Ibis track nullability?
    - [Gil] Ibis treats everything as nullable after the table scan.
    - [Victor] This comes up in Calcite because Calcite disagrees with Substrait on the types of columns.
- Release Process

    - [Jacques] I'm so thankful that we don't talk about releases.
- Victor's very specify gripe about Calcite Aggregations not allowing anything other than references in AggregateCalls
    - [Victor] This causes problems when trying to represent functions like stringagg(<string>, "literal")
    - [Victor] If you have a reference to an input, and the input column is a constant, does that satisfy an an argument being constant.
    - [Jacques] One approach to this would be to generate functions for all each invocation.
    - [Victor] Another approach is to curry the constant argument into the Calcite function and then hid it from Calcite.

Nov 22, 2023

Agenda

- [David] Sharing functionality between producer and consumer (recent Slack discussion)
    - Are there any mechanisms that we prefer over others?
    - Are there any we'd like to discourage?
    - Seems like a good FAQ addition.
    - [Jacques] The intention is that the extension URIs are canonical. With capabilities, the idea was that you would share the set of things that you support.
    - [David] From what I'm seeing, it looks like most of what they want is there. One thing that may be missing is that there's no way to specify options. It's possible a producer could send an option a consumer doesn't support.
    - [Victor] From the docs, a consumer is expected to fail if it does not support any of the options it receives.
    - [Victor] If a producer wants an options that a consumer doesn't support, would it make sense to fail at the consumer when the plan is sent or the producer when it tries to generate the plan? The producer can generate a much nicer error message.
    - [Jacques] Another way to approach this would be with an exception reporting mechanism.
- [Victor] What does it mean for an argument to be a constant?
    - [Jacques] A constant is a value that can be known at plan time.
    - [Victor] So effectively literals?
    - [Jacques] Also subtrees that can be folded into constants.
- [David] PRs waiting on action
    - #265 - feat: support for simple extensions dependencies
    - #520 - docs: add example for field_references
    - #521 - docs: clarify map key behavior
    - #523 - docs: elucidate obscure discussion points
    - #536 - feat: add extra option for on domain errors in log functions
    - #546 - docs: expand the ExpandRel documentation

    - *#548 - feat: add current_date and current_timestamp functions*
        - [Jacques, paraphrasing] These may not be functions we want to have in the spec. They behave very weirdly. For example if current_timestamp is evaluated in a single system multiple times it returns the same value, but if you call current_timestamp in a federated system the different systems will generate different timestamps. In practice, these behave more like macros.
        - [Jacques]These are more like variables that you evaluate once in a plan.
    - *#572 - feat: add PhysicalJoinType*
    - #578 - docs: faq section + post-join filter question

Nov 8, 2023

Agenda

- [David] CrossRel – should this be CrossJoinRel for consistency?
    - If so, do we want to take the hit for a breaking change?
- [Dane] Physical Join operators in Isthmus
    - How?
    - FYI: HJ and NLJ are added to Substrait Java core. Merge Join has a PR open.
- [David] “Prepared Substrait” plans discussion from Slack
    - https://substrait.slack.com/archives/C02D7CTQXHD/p1698387191278629
- [David] Going from a query plan to Substrait and back (https://github.com/substrait-io/substrait/issues/571)
- [David] No PR of the week
    - All PRs are newer than one year, but would be nice to make progress on the 3 month old ones. Perhaps everyone takes one PR and pushes it to completion?

Notes

- Re: CrossRel
    - David to check with Pedro @DuckDB for CrossJoin (is it logical/physical and is it still used).
- Re: Physical Join ops in Isthmus
    - [Jacques]: Might be some old Dremio code that does this
    - Is the input physical/logical?
    - Lots of phases of optimization
- Re: Prepared Substrait
    - [Jacques]: May have had a long discussion previously
    - [Victor] One use case for this is to cache query plans. We see a lot of templated queries which only have constants changing, so being able to cache the plan and sub in the variables would be nice.
    - Generally seems like a good idea. Whether they should be typed or not is not necessarily clear (seems related to issue #515).
    - …

    - [Jacques] I'm interested in this because it's a very common pattern in SQL to use prepared statements for which this would be useful.
- Re: SQL to Substrait and back
    - Weston to reach out to Datafusion
    - Do we need a FAQ?
        - PostJoin filter (Victor: Will handle this)
        - Names vs Numbers
        - How do you pick from two sides of a join?
        -

Attendees

- Ashvin Agrawal
- Dane Pitkin
- David Sisson
- Jacques Nadeau
- Richard Tia
- Victor Barua
- Weston Pace

Oct 25, 2023

Agenda

- [David]: Pending PRs now have specific assignees where return feedback is required to progress
- [David]: PR of the week: Add arg_min and arg_max and Add least and greatest

Oct 11, 2023

Agenda

- [David] feat: add NestedLoopJoinRel definition #561 - Do we want to be duplicating the enum JoinType in every join relation? If not, what's the path forward to replacing the existing JoinType copies?
- [David] PR of the week: chore: improve coalesce nullability handling #340

Notes

- Re: NestedLoopJoinRel
    - [David]: Do we want to replicate the JoinType enum for every one of these joins that we have?
    - [Weston]: A similar question came up when looking at the physical hash equi join. The situation we wanted to avoid was having context-dependent fields (e.g. these two operators share a message but these fields only make sense when using

operator X). However, I think pulling out the JoinType by itself might not be such a bad idea.
    - [Dane]: Are there any join types that don't make sense in all of the relations?
    - [David]: There might be, but that is something we could document.
    - [Weston]: Technically, that is something we wanted to avoid. I don't know if any such cases really exist for JoinType though? What do we gain by pulling this out?
    - [David]: Looking at the current status there is one physical join types that aren't in the logical type (right anti) and one join type in the logical joins (single) that isn't in the physical joins.
    - [Weston]: I think it's better for a physical relation to share the logical relation's join type and not handle all the cases than it is to go the other direction.
    - [David]: One way to handle the left/right anti thing would be to have a separate left/right property on the physical rels and reuse the join type. However, this would be backwards incompatible.
- Re: coalesce nullability handling
    - [David]: There was discussion on this in the past. Some general hope we can wrap this up. However, no one actually followed up on this. However, it looks like it's ready to go.
    - [Weston]: So my memory, from the last discussion on this, was that we preferred making a change to the miniature programs that calculate return types (e.g. that used in decimal) and not change the complexity / syntax of the return type parsing that exists today.
    - [Weston]: However, this means, as an action item, we need to document the syntax of these miniature programs that we support today (e.g. the syntax of what is in decimal's return)

Sep 27, 2023

Agenda

- [David]: Verify that this new time works for all.
- [Gil]: compound type signatures for variadic functions

  A function add that takes two int32s gets the signature `add:i32_i32`

  Coalesce is variadic, with a minimum number of arguments of 2
  Should the signature be `coalesce:any1_any1` b/c min(values) == 2
  Or should it be `coalesce:{'_'.join(["any1"] * len(args)}`

  In discussion with Weston on Slack, it seems like it was the former (no. of minimum args).

What do we do about the kleene boolean or which has a minimum number of args of 0 (and defines behavior for getting no arguments) – does it have a type signature at all?

- [David]: Any update on Gluten/Spark efforts?
- [David] Q: Why do we only have a definition for the query plan and not a query protocol? Seems like everyone using Substrait ends up packaging it differently which means anyone wanting to use Substrait needs to support N-different communication methods
- [David] Cross platform compatibility

  Currently Substrait describes what to do with the data but given the options we've put in the definition it doesn't mean that any two systems agree. We can't just point at any Substrait conformant backend and expect it to just work. We need some way of getting on the same page. That could be capability information and a query transformation library and it could mean mandating a specific behavior and forcing backends to adapt (or at least their Substrait adaptors to adapt). How do we want to position Substrait? Are these concerns out of scope? And if so, where should those capabilities end up?
- [Dane] ExtendedExpression support/tooling
  - Adding EE support to Substrait java and Isthmus
  - Arrow can now accept EE blobs as project/filter operations in some implementations
- [David] PR of the week: [Point to IEEE 754 instead of Wikipedia #449]

Notes

- Re: new time
  - [David] Does this time work for everyone?
  - [Ashvin] It works for me but this is a popular time for meetings. However, I think it will be ok.
  - No objections, let's proceed
- Re: Compound type signatures for variadic functions
  - [Gil] I discussed this briefly on Slack. There is no guidance on the site for how to handle type signatures for function implementations that are variadic.
  - [Gil] From the discussion on Slack with Weston I think we landed on using the minimum number of arguments. In other words, if your functions was foo(bool…) then use foo:bool regardless of how many arguments are being used.
  - [Gil] An alternative would be to use the number of arguments you are actually using. For example, if you have foo(x, y, z) then use foo:bool_bool_bool.
  - [David] What does C++ name mangling do for variadics? Can we use that as precedent?
  - [Gil] As a peculiar aside, I noticed that there is or_kleene which could potentially have 0 arguments
  - [Jacques] I don't understand why we would have that? What does an or with zero arguments return?

    - [Gil] Null?
    - [Jacques] What about T or F? I think it would need to be well defined.
    - [Gil] Actually, looking at Arrow, it appears the current implementation returns false
    - [Jacques] Is this something we really want?
    - [Weston] Maybe survey existing engines and see if Arrow is the only one that has something like this.
    - [Jacques] Can we return to the original question. I believe that the intention was that the extension functions section of the plan comes from the YAML. So if there is one implementation in the YAML then there should be one encoded version of the name.
- Re: Gluten / spark efforts, any update?
    - No
- Re: Why do we have a standard for a plan but not a protocol?
    - [David] It seems that everyone is needing to come up with a similar and duplicate way of communicating the plan itself.
    - [Jacques] Well, there are different ways you might want to communicate a plan. You might want to communicate the plan to get a query result or maybe to optimize the query or maybe to validate the query. Which case are you thinking of?
    - [David] Yes, but a protocol could be extensible too. For example, if I just want to point at some arbitrary backend and have my query work then you would need a compatible protocol. You'll also need prior knowledge of which backend you are communicating with to know what protocol to use to communicate with it.
    - [Weston] Isn't this ADBC or Flight SQL?
    - [Jacques] Yes, they support Substrait
    - [David] Do we need to pick one of those as the official mechanism for substrait queries?
    - [Weston] I don't think Substrait cares what users are using. Users care. But they aren't using substrait directly. They start with ADBC or Flight SQL and then use substrait as their mechanism of choice.
    - [Jacques] I think we want a default. There is a lot of complexity here that we are very familiar with but users are not. We ran into this problem with Arrow too. So I think we want some kind of default for users to start with. Then, if people want advanced cases that don't fit the default, that is fine, but at least this way users have something easier to start with. Does this address the problem?
    - [David] It helps but I would like it to be stronger than a recommendation. The only way we will get interoperability will be to have common behavior. If I have to convert to text to use duckdb and I have to convert to json to use some other engine then we have a problem.
    - [Jacques] I think this is a question of persona. Most common personas (I am a company X trying to solve Y) aren't seeing the benefit without this common layer. However, is the problem that these personas aren't active in the community? Is this a cause and effect thing because we don't yet have people that need this?

    - [Weston] Maybe we define something like a “substrait compatible query engine” which is really substrait + adbc
- Re: cross platform compatibility
    - [David] That brings us to the next question. We don’t yet have any capability querying in the protocol. We don’t have consistent enough functionality.
    - [Weston] For the capability queries I wonder if we need to add some kind of capability query APIs into ADBC?
    - [Weston] For consistency, I think it is risky to mandate any particular behavior because consumers aren’t going to want to change.
    - [David] Yeah, but I think there is some core behavior that we need to define. An interesting example case is https://github.com/facebookincubator/velox/discussions/6597 which did solve things by having the two implementations adopt the same behavior.
    - [Jacques] Yes, I think we want to get to consistent behavior, but it is going to be in many steps. For example, polyfills can patch libraries to add missing behaviors. That’s many layers from where we are today, but that’s how I see this playing out eventually.
    - [Jacques] Nearer term, if I want to switch from platform A to platform B, then I have to accept the behavior may be different. One interesting anecdote. There was once an engine that didn’t support errors on overflow, but customers really needed it. So eventually a compatibility layer was added, but it was 2x as slow and no customers turned it on.
    - [Jacques] One of the things that comes up as we are talking is “where should substrait be headed”? I think there is a small subset of personas we are currently targeting (people that are building databases) and maybe we want to get to the place where we target broader personas (people building client applications or wanting to query multiple backends) and then I think we focus more on this type of feature.
- Re: Extended expression support
    - [Dane] Does anyone know of any ecosystem work being done around extended expressions? I’ve noticed there is some work on Java being done here. But I noticed that other implementations aren’t supporting that yet.
    - [Dane] I do plan to eventually get this in substrait-java and sql but I’m curious who else is using it?
    - [Ian] As context, I have also been pushing more on the voltron data team to provide more support here. Many of the problems we are encountering are just moving expressions back and forth between different libraries and this is an opportunity for substrait that we wouldn’t want to miss.
    - [Weston] I am adding support for this in rust, this week, in particular I hope to support converting from extended expressions to datafusion’s logical plan.
    - [Ian] That’s good, I have a little demo showing how this works in pyarrow/python but it’s not very useful yet until others adopt support for it.
    - [Jacques] Originally there was some excitement for this in storage providers. Has there been any traction there?

    - [Ian] Not that I know of, but it seems like a huge opportunity for these systems, e.g. S3 select could support this.
    - [Weston] Maybe one way we could promote this is to have a python "expressions" library that uses substrait under the hood but doesn't even mention this to users. It's just a single library that any python user can use for programmatic expression building and then they can convert to polars, datafusion, pyarrow, pandas, etc.
    - [Dane] The quickest way to that might be either sql glot or ibis.
- Re: PR of the week
    - [David] This one seems pretty straightforward. It's a modification to an existing function. We are just pointing to IEEE instead of wikipedia.
    - [Jacques] Approved

Sep 13, 2023

Agenda

- [David] PR of the week: Code of Conduct (#339)
- [Jacques] Meeting time
- [David] Website reviews take too long

Notes

- PR of the Week
    - [Victor] Took a look. The only thing that was missing was a way to contact the SMC.
    - [Jacques] I can take a look. Possibly make an email or somesuch.
    - [Victor] What do security@substrait.io and harrasment@substrait.io go to?
        - Both of these emails are unable to be delivered.
    - …
    - [Jacques] The committer list is not very well populated right now.

Action Items

- Make security@substrait.io, harrasment@substrait.io should be real.
    - Document who receives these emails.
        - security can go to committers
        - harassment should be limited to SMC
- Meeting Time
    - [Jacques] Proposal to move this back an hour.
        - Current time was picked as it worked for folks in India.
    - [Victor] Current attendance is now folks in the US. I think it would make sense to push this back a bit and then reconsider time if folks from other time zones wish to join.

    - [Jacques] I've seen other projects which move the meetings every other meeting time to accommodate for this.
- Website
    - [David] My goal this quarter is to push out the website update.
    - [Jacques] Are you have difficulties getting PMC reviews?
    - [David] The requirements for two PMC reviews for all spec changes, even non-semantic ones, is a bit onerous.
    - [David] Can we reduce the requirement for the non semantic spec changes?
    - [Jacques] Let's make a proposal for that.
    - [Jacques] Additionally, we could expand the SMC. The main concern before was having too many folks from Voltron.

Aug 30, 2023

Agenda

- [David] WindowRel and ExpandRel are in as of last week. Are the Gluten folks now unblocked and progressing?
- [David] PR of the week: The specification page – what do we want it to look like? #490 vs #547

  [Weston] What's keeping us from a 1.0 release?
- [Richard] Any more PR feedback for https://github.com/substrait-io/substrait/pull/534?
- [Victor] substrait-java build failures
    - https://github.com/substrait-io/substrait-java/actions/runs/6004797518/job/16316860418?pr=170
    - [Victor] I will take a look

Notes

- [David] WindowRel and ExpandRel are in as of last week. Are the Gluten folks now unblocked and progressing?
    - [Victor] The spec needs to be pulled into substrate-java. There is a PR out for that. This will add the protos, but there would still be work remaining to update the POJOs.
    - [David] Meta has expressed interest in Substrait, but they are waiting on two things. Physical plan support, and 0 vs 1 based indexing support in functions
    - [Weston] Meta is also interested in a way to flag distributions. For example, they want to be able to specify isPartial in a SortRel. They had a component that looks at the logical plan and can add this flag, they have a separate component that uses this flag. [Relevant PR]
    - [Victor] Isn't this what enhancements are for?
    - [Weston] Stepping back, the question to ask is who is looking to convert Substrait plans to Velox (or vice-versa).

    - [Ashvin] There is some exploration we're doing in this space around Spark. The Spark non-distributed physical plan can be converted to Velox.
    - [Weston] Velox has implemented thread parallelism via … [lost track of discussion]
    - [Weston] Coming from Spark single-stage logical plan, something is going to have to do the smarts to split that over multiple files.
    - [Weston] I don't think we're missing a physical plan for Spark, I think we're missing a planner component. Just adding an isPartial flag is insufficient, as when translating from Spark to other system we'll still need a planner to handle this.
- 0-or-1 based indexing for functions like substrait
    - [Weston] Original problem, Presto and Spark have different indexing which Meta ran into when trying to standardize behaviour. Challenge is that we don't have a catalog of all of these differences, there's a lot of weirdness in this and other behaviour (overlow, NaNs).
    - [Weston] There are some substrait functions that have 4 different options with 4 different behaviours.
    - [Victor] Shouldn't this be the responsibility of the consumter / producers. If you're writing the translation you should know the behaviour of your system and be able to map it correctly.
    - [Weston] You're assuming that the folks writing the adapters are knowledgeable in the systems. Now something like 0-1 based indexing is fairly fixable, but overflow behaviour and other such options are hard to fix.
    - [Weston] Just because the spec is opinionated doesn't mean you have to propagate those opinions into your system.
- [David] PR of the week: The specification page – what do we want it to look like? #490 vs #547
    - [David] I've been making a summary of PRs in each repo weekly. I'm also thinking we should try and tackle one PR a week to work through it.
    - [David] This week, what's missing / out-of-date in the specification page
    - Page in question: https://substrait.io/spec/specification/
    - [Victor] Something that I think is missing before 1.0 would be the ability to allow references to functions and types from other extension files.
    - [Victor] Is 1.0 something that we could make breaking changes?
    - [David] According to our docs we would remove all deprecated fields.
    - [Action Item] David and Weston to collaborate on this.

Aug 16, 2023

Agenda

- [Victor] Moving away from Runtime Exceptions in substrait-java.
- [David] Increasing PR velocity.

    - Many PRs look like they're just missing PMC approval:
        - https://github.com/substrait-io/substrait/pull/339
    - I volunteer to get aggressive with labels – but will that be enough to get things moving?
- [David] Any thoughts on the revised home page that moves a lot of content to a separate page?
    - Separately, what do we think of adding a bit more jazz to the home page? (Take a look at the Apache Spark design for inspiration.)
- [Matthijs] I can't attend the meeting today, but can someone please take a look at the failing release workflow?
- [Victor] non-deterministic build issues in substrait-java
    - Is it possible to run tests on a different version of java than we build the application?
- [Victor] Window Function details.
    - Window Functions (in Postgres) allow for usage of types like intervals as offsets in Window Functions

Notes

- [Victor] Substrait Java throws a lot of runtime exceptions which is a little odd for a library. If we change to checked exceptions then it could be more friendly / explicit
    - [Weston] Major Java contributors aren't here. Shoot a message on the mailing list about this to get their opinion.
- Increasing PR Velocity
    - [Weston] It can be hard to tell which PRs are ready for re-review. I've created an "awaiting PMC approval"
    - [Dane] Do all PRs require PMC approval?
    - [Weston] Spec changes require PMC approval, for libraries it just requires committer approver.
- Window Rel PR
    - [Weston] This need a rebase.
    - [Victor] Do we need to wait for the original contributor to do this, or could we do it on their behalf?
    - [Weston] If you have their permission it should be okay to do it on their behalf.
- Release Workflow Issue
    - [Weston] API key appears to have expired.
- Build issues in Java
    - [Victor] Can we drop jabel in tests? It seems to generate some rather difficult to understand failures. I updated the substrait proto ref and it caused some errors.
    - [Jacques] Jabel's purpose is to tell the compiler to allow futuristic code as long as it can still compile down to Java 8 but it shouldn't actually be modifying the bytecode.
    - [Jacques] This pretty much boils down to whether we want to support Spark or not which still runs on and supports Java 8. Java 8 is very old and I think it would

be rather cumbersome to make the code itself fully compatible with Java 8 without Jabel.
    - [Victor] Do we think just not running the tests with Jabel is a reasonable solution?
    - [Jacques] Yes
- Window Function Details
    - [TLDR] Interval support in offsets seems to vary between engines. It could be worth having an extension point to allow users to use arbitrary types for offsets. We [Victor] should investigate how common interval support is.

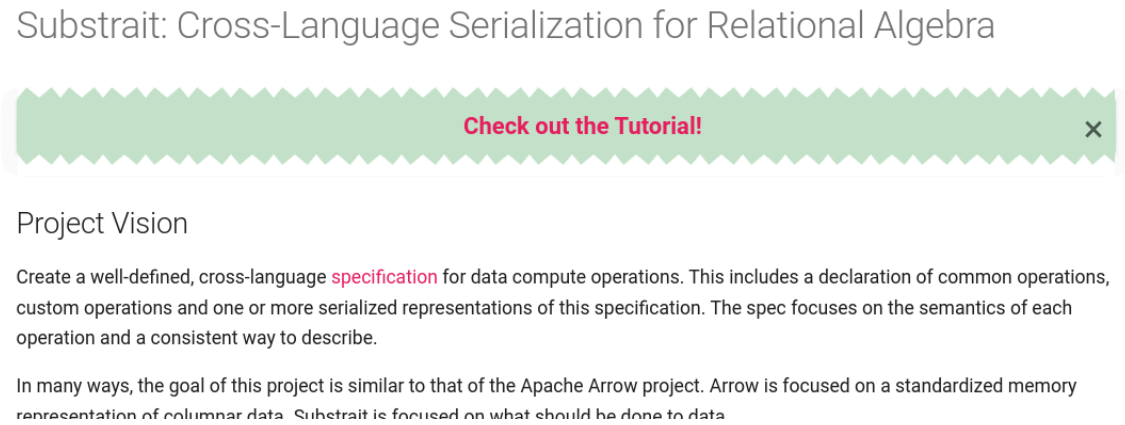
Attendees

- Ashvin Agrawal
- Victor Barua
- ~~David Sisson~~ (lost connection and can't rejoin)
- Ian Cook
- Matan Peleg
- Sri Nadukudy
- Dane Pitkin

Aug 2, 2023

Agenda

- [Weston] Do we want more sophisticated syntax for expressing nullability shorthand (|? And &?)? https://github.com/substrait-io/substrait/pull/247
- [David] How much Javascript do we want on the Substrait home page? I've proposed adding a call to action for visiting the tutorial here: https://github.com/substrait-io/substrait/pull/526



- [David] Any thoughts on absl::Status and absl::StatusOr<X> use in substrait-cpp? It's going to be a while until C++23's std::excepted implementation is available. We could create some other similar class to hide it in the meantime but absl is already an explicit dependency of protobuf. https://github.com/substrait-io/substrait-cpp/pull/80

- [Matan] Updates regarding Spark after meeting the folks from Intel.
- [Victor] Validation. Where does it happen?

Links

-

Attendees

- Ashvin Agrawal
- Victor Barua
- Matan Peleg
- Richard Tia
- Vibhatha Abeykoon
- Weston Pace
- Gil Forsyth
- Ian Cook
- Krystian Sakowski

Notes

- Re: Special Nullability Syntax
    - [Weston] There are some functions where the nullability of the output type can be known from the input types but we can't express it with our current syntax.
    - [Weston] For example, coalesce. We know the output is nullable if and only if all the inputs are nullable. E.g. If there are any non-nullable inputs then the output is non-nullable.
    - [Weston] There are roughly three proposed solutions
        - Just approximate with what we have today (e.g. say the output of coalesce is nullable)
        - Introduce new shorthand (|? and &?)
        - Introduce a mechanism for the "calculate the return type with a small inline program" to also be able to calculate nullability
    - [Jacques] What if you ended up with a scenario where some of the inputs (but not all of them) determine the nullability of the output type? Would these approaches be able to handle that?
    - [Weston] That situation would not be handled by the first two approaches (we would need to approximate) but presumably could be handled by the third
    - [David] I am interested in whatever we can do to make sure that the YAML explains the output type as much as possible so things are fully defined. For example, <any> is ambiguous and requires users to explicitly declare the output type.
    - [Jacques] I think we're ok with <any> because it doesn't mean the output can be any type. It means it needs to be the same type as the "any" input.
    - [Victor] Is this the same as the nullability option
    - Discussion about whether this would be the same

    - [Weston] I think you're right, and this would be valid
    - [Jacques] Yes, and maybe coalesce isn't enough justification for this, but I still think it would be good to have new functions in the output derivation syntax at_least_one_nullable and at_least_one_not_nullable.
    - [Jacques] However, I agree a new nullability option is simpler and might be good enough for coalesce and we can handle the more complex cases later.
    - [Weston] So if nullability is set to mirror do we include a '?' in the return type or not?
    - [Jacques] I think we don't, and it should be a validation error, but I doubt we check for that today.
- Re: Call to action for tutorial and/or javascript in the site
    - [David] I have a PR pending to add a big box at the top of the site. However, this depends on JS and isn't fully builtin to mkdocs. I'd also like to use some custom JS to add our meeting calendar to the site.
    - [Jacques] My initial preference is to rely on the framework as much as possible. So, and this isn't a strong opinion, but if it doesn't exist in mkdocs then I would say we don't do it. Mainly because this seems like a classic case for code rot and maintainability issue.
    - [Jacques] Did you first check to see if mkdocs has this capability?
    - [David] The dialog box itself is mkdocs with custom styling. The JS is used to ensure the user doesn't see the tutorial on future visits to the site.
    - [Victor] Meta question here, do we really want the call to action to be dismissed on future visits?
    - [Jacques] Or, we could simplify the homepage so that the call to action is more prominent by removing some of the other content.
    - [Gil] This is the approach we took on Ibis, our front page is now only the getting started guide and a simple statement.
    - [Jacques] On a similar thought, I think we have a few too many top-level navs as well. 5 or less is my rule of thumb. We have talked about moving all the spec stuff into a single top level nav as well that would be good. I think this call to action is pragmatic but maybe the approach should be wider.
    - [Jacques] We are getting a bit off topic here. I think the original question was "should we use JS?" and my answer is still "try not to as much as possible, only as a last resort"
    - [Weston] Maybe the calendar is a better example of this case. There is no good mkdocs plugin to display a calendar on the page. Would this be a better justification?
    - [Victor] Can't we put a link to the calendar?
    - [Weston] Yes, but I think David's goal was the embed the calendar itself (as a widget)
    - [Jacques] We should see what's the most popular way of doing this and, if it requires JS, then that is ok.
    - [David] I think this is good input. I think I have what I need to proceed.
- Re: abseil

    - [David] abseil is a pretty large dependency. It is something our users will already need to some degree. However, status & status-or are only placeholders until we move to the newest version of C++. Do we want to expose abseil in our public API which pretty much cements it as a dependency even if we could switch to some builtin C++ thing?
    - [Weston] I think we can. We can still move to builtin C++ in C++20 but I doubt that protobuf is ever going to move away from abseil.
- Re: Spark
    - [Matan] I got in touch with the gluten team. They are willing to do the upstream contribution. They need attention on the two PRs (expand relation and window relation) and once those are in place they can do their contribution. Once that is in place then we can put in the extensions we have added on top of that.
    - [Weston] I think the expand relation is ready
    - [Jacques] I think the markdown is not sufficient yet. The proto description has a lot of information but there is not enough in the markdown or actual spec content.
    - [Weston] I will make a pass at updating this content this week and ping you.
    - [Weston] Regarding the window relation, <explanation of the PR>
    - [Weston] I think the main open question here is whether we want to duplicate a lot of the fields. However, I do think we ended up with consensus that trying to share the fields would be a bad idea.
    - [Jacques] Actually, there was another open question that you had on the PR weston. Why do we need both a “hash window rel” and a “streaming window rel”. I think “hash” is probably not the best name, but there are cases where a window rel implementation does not need to keep all of the results in memory (e.g. if count is the window function)
    - [Weston] That makes sense.
    - [Jacques] Are we confident these are the only two things that the Intel team needs? I seem to remember some other PRs that we declined.
    - [Matan] Yes, they have said that these are the only two PRs that are blocking them from upstreaming this work.
- Re: Validator
    - [Victor] “Validation” got mentioned a few times. How do we handle validation when we have several different applications that all might have a slightly different interpretation of things.
    - [Jacques] I think the goal was that the “validator” serves this purpose. It should have bindings in different languages so that they can use them.
    - [David] Is this the approach we want to take for other things? For example, the text format is probably something we want to use in different places. Do we want to follow the same approach with it?
    - [Jacques] Yes…but a word of caution with C++. If you want something to be a reusable tool then make sure you do as much as you can to make sure you can import that tool without importing any unnecessary dependencies (and also avoid dependencies in general).

Jul 19, 2023

Agenda

- [Gil] Relax / remove function signature compound names (https://substrait.io/extensions/#function-signature-compound-names)
- [Weston] https://github.com/substrait-io/substrait/pull/518 is probably ready to merge, needs one more PMC approval (adds exchange rel to rel's one-of)
- [Carlo] Question about Coral/Transport

Links

-

Attendees

-

Notes

- Re: Function signature names
  - [Gil] In the spec and the documentation there is a mini-DSL for function signatures (e.g. add:i32_i32) with separate notation for nullability (?) and variadics (any).
  - [Gil] It's a pain on the producer's side to produce. I can see how it could be useful to have this information front-and-center but it is redundant.
  - [Gil] This became an issue because someone took a plan from Ibis (which doesn't write this) and tried to convert it with substrait-java (which expected this)
  - [Gil] Given that the known consumers (Acero, DuckDb, and Velox) do not use this information, do we really need it?
  - [David] If you have two different add kernels (one which takes int and one which takes float) then how do you distinguish between the two?
  - [Jacques] I haven't spent a lot thinking about this in a while. I agree it seems excessive in the obvious cases. I think though, that there are edge cases, where it is required. And it is difficult to recall those off the top of my head.
  - [Weston] There are a few things I recall. A function could have multiple ambiguous kernels. For example, you might have a foo:any,i32 kernel and a foo:i32,i32 kernel. You would need the fully qualified name to resolve which kernel to choose.
  - [Weston] Another reason was that it becomes easier for middleware components (e.g. validator, planner) to process the plan because they don't need to look up a kernel library to deal with expressions.
  - [Weston] Lastly, it could be easier in the future to implement a substrait-first consumer because you don't have to worry about kernel binding / resolution. You can simply make a big map from fully qualified function name to kernel.
  - [Victor] Another advantage could be that the consumer might create an invalid plan (e.g. add_i8_i16) and having this type information the consumer could reject

it instead of implicitly doing something (and leading the different results between different consumers)
    - [Jacques] There may be cases where this information isn't needed. However, since this is machine-to-machine communication, I think it is generally not too hard to be explicit here and by being explicit we are better situated for potential future use cases. For example, I think Weston's last point (easier for substrait-first implementation) is significant. We saw things like this in Arrow. Initially, people are using the technology as an add-on but later, as it grows in popularity, it may be the first-class approach.
    - [David] Do we need to have the information in the name? Could we put it in an easier to parse location.
    - [Jacques] The string should never be parsed.
    - [David] Then how do we do function lookup
    - [Weston] If you already have an existing system with a two-level lookup then I think you at least need to split it.
    - [Jacques] Is the problem here that existing implementations just aren't validating the plans? How many engines allow implicit casts today?
    - [Aldrin] Wouldn't it be good to let engines be more flexible, in case an engine did want to allow float/int operations for some reason?
    - [Jacques] More flexible is ok. An engine can still allow float/int operations, it just needs to be explicit about it.
    - [Weston] Well, we can do that validation without the fully qualified type strings today
    - [Jacques] True, I'm not arguing that the type strings are needed, I'm just wondering how important this happens to be to people today. Are consumers even validating these plans?
    - [Weston] Fair point, we don't reject plans in Acero today. Even if those plans do involve implicit casts. I see that as a "garbage in / garbage out" case. If I get an invalid plan then it's probably not terrible that I give valid results.
    - [Weston] A bigger concern, for me, is that different engines might give different results to two valid plans because they are ignoring the options.
    - [Jacques] I think, from a user's perspective, these two things look very much the same.
    - [Jacques] Circling back, if the whole community wants to remove this, then I think we should consider it. I think what we need to do then is find the corner cases and make sure we have other ways to address them. However, I think this is valuable information, especially for future engines that will be substrait-native.
- Re: Coral/Transport
    - [Carlo] Coral and Transport are two different technologies which appear to be the same thing as Substrait. I'm wondering how these are related to Substrait?
    - [Jacques] I don't think they're quite the same thing. Coral translates between Trino & Hive and uses Calcite. Transport is an API definition for making portable functions. The main challenge is that it is entirely tied to the JVM.

    - [Jacques] I think, having transport embedded functions (wasm embedded functions, …) makes a lot of sense. But that is how I see transport.
    - [Jacques] Since Coral can go to Calcite and Substrait can go to Calcite. I think we can already go from Coral to Substrait and back.
    - [Carlo] I think the value here is that these technologies have already done a lot of the heavy work of translating functions between different dialects (e.g. trino / spark)
    - [Jacques] Yes, but Coral is very much implementation based. So I can do the translation, but only in the context of the JVM. I think there are two things. Can we give people a toolkit to use Coral easily is one and can we extract the knowledge from Coral into Substrait is another.
    - [Weston] One thing we are working on, that I hope to release soon, is that we are starting to catalog a lot of these differences. However, we have not looked much into how we translate from one dialect to the other.
    - [Carlo] Yes, and I think this is one thing that these systems have done well, and there is a lot of labor involved. Either we end up repeating all of that work or we find a way to use that information somehow.

Jul 5, 2023

Agenda

- None!

Links

- Latest Text Format PR

Attendees

- Ashvin Agrawal
- David Sisson
- Matan Peleg
- Victor Barua
- Jacques Nadeau
- Gil Forsyth

Notes

- Spark progress
    - Spark mostly uses Substrait at the low level, with custom plans per system
    - Also uses older version of Substrait
    - Top level/mid level/low level plans – are they all Substrait?
- Unbound relations/expressions
    - PyArrow compute expressions is the main motivation

    - But could be applied to Spark use case
    - Not currently a focus for anyone
- Text format is nearing completion of its second milestone
    - Supports all TPCH queries not containing subqueries
    - Is a bit more verbose than previously, with explicit types in functions and field references (includes the schema)
        - Will be addressed later on to make it simpler
-

Jun 21, 2023

Agenda

- [Weston] PRs needing a second SMC approval
    - Add ordering to aggregate measures (e.g. ARRAY_AGG(<c> ORDER BY <C2>))
        - https://github.com/substrait-io/substrait/pull/498
- [Weston] Expand vs. Unnest
    - Sent email to the mailing list. Interested in hearing opinions on whether we need both or just one and, if just one, which one to prefer?

Links

Attendees

- Ashvin Agrawal
- David Sisson
- Gil Forsyth
- Ian Cook
- Jacques
- Matan Peleg
- Richard Tia
- Weston Pace

Notes

- [Weston] Needs a second PMC approval for the PR above. Would allow per-measure ordering in aggregate outputs
    - Jacques: I thought we already solved this
    - Weston: there is an aggregate field “sorts”
    - This may already be solved, Weston will look into it further
    - David: need more docs to make this easier to find
- [Weston] PR written by the Gluten folks to add EXPAND to Substrait b/c it’s a logical op in Spark – Weston discovered that engines have EXPAND, others use UNNEST in place of EXPAND. EXPAND can be used for pivot_longer and ROLLUP operations.
  EXPAND functions as an n-way projection
  DuckDB, Datafusion, Postgres use UNNEST

Spark and …. Use EXPAND

We can pick one and try to centralize, or we can adopt both.

More details here:
https://docs.google.com/document/d/1WS9SAfyT-xQa2Nqq5KXdjeRXrNCAei6bZHa398wqXmA/edit?usp=sharing

- Jacques: feels a lot like the in vs. or thing – `a in [1, 2, 3]` vs `a == 1 or a == 2 or a == 3` – it is non-trivial to get from one of these to the other. Possible, but non-trivial. Subqueries are similar, they can collapse into a common primitive but it's hard to explode back out. Inclination is to apply the same rule here in that the pain to go from UNNEST to EXPAND and back (while possible) is sufficient reason to adopt both. Forcing the edge to convert from one to the other would be burdensome.

  If there's sufficient use of both patterns, then it seems fine to adopt both.
- Weston: these aren't scalar functions but they look like functions. Should they pretend to be scalar functions or should we introduce a new type of function / relation
  - Jacques: the moment you have a different number of input -> output rows, that's a relation, not a (scalar function) "I'm a SQL guy". It's weird if a function can change the cardinality of the output of the input
  - David: does it matter that we put stuff under scalar function from a protobuf standpoint?
  - Weston: it gets confusing when you have trees of expressions
    Select 1 + max(x + 1) → three relations
  - Jacques, when you define measures in an aggregate function, the root must be an aggregate function, it cannot be a scalar function. You can express sum(x + 1) in an aggregate, but the 1+ on the outside needs to be a project afterwards
  - Weston: both scalar and window functions are part of expression trees, but it's important for the engine to be able to distinguish between them. The engine can figure this out from the plan, but it's helpful to know up-front that you will _need_ to handle both in a given plan.
  - Jacques: it is a useful distinction even if it's an arbitrary distinction

- [Weston] When do we add relations? What are the criteria if someone wants to make a change to the core proto files?
  - Multiple engine support
  - Translate into substrait terminology
  - Don't go it alone (you'll be more likely to succeed with community involvement + feedback)
  - [jacques]: is this unique to relations? Or is it the most prominent place this comes up? We don't spend a lot of time talking about what "add" means because there's a general consensus.

    - Jacques: is the broader issue here that reviews aren't responsive / speedy enough?
        - Probably part of the issue, but other problems are that sometimes prospective contributors don't want to continue driving something and it requires more effort from the maintainers to get it over the finish line.
    - Can we set up more structure around the state of a PR:
        - Needs more approvers vs.
        - This is in a draft state and needs lots of work
    - Everyone add labels and notes to everything, we can clean up whatever mess we make later
- [Matan] Spark Talk
  Working to switch to Substrait completely, only working in the logical part. Not sure where we stand re: Gluten folks.
    - [Jacques]: let's just post a patch and get it merged
    - [Weston]: we can make an effort post-merge to get the updated stuff merged back into Gluten
- [Matan] Spark Talk 2
  Expression extension – done internally but would be nice to contribute it back (and make a smaller contribution). Might need some help to make sure this is properly extensible / generic. (AdvancedExtension in Expression)
    - [Jacques]: One of the weaknesses of the current model is that it's not an array, it's a single advanced extension, so it's awkward to combine standard and advanced extensions, so it might be good to have them as an array of extensions

Jun 7, 2023

Agenda

- [Gil] Application of governance rules to language-specific substrait repos
- [Gil] Make your substrait org membership public: https://github.com/orgs/substrait-io/people
- [Follow up action items] Substrait-Spark extraction conversation

Links

- Original PR discussing substrait/spark/gluten: https://github.com/substrait-io/substrait-java/pull/90
- IP clearance template from ASF: https://incubator.apache.org/ip-clearance/ip-clearance-template.html

Attendees

- Ashvin Agrawal
- Martan Peleg
- Victor Barua

- Ian Cook
- Gil Forsyth
- David Sisson
- Weston Pace
- Richard Tia

Notes

- Re: Governance rules on language-specific repos
    - [Gil] We have the new substrait-python package and repo. I wanted to confirm that we don't require two SMC votes for a commit right? Do we just need a committer to approve a PR? As committer do I have free reign?
    - [Weston]
    - [Victor] The governance doc does have a section regarding code changes and it just says there needs to be a +1 vote from a committer and an approved PR counts as a vote
        - https://substrait.io/governance/#votes-on-code-modification
    - …
    - [Weston] Current interpretation is approval by a *different* commiter.
- Re: Substrait org membership
    - [Gil] There is a PR that has merged that now automates fetching the membership information from Github to create the list of committers and SMC members. However, this workflow doesn't have permission to read the organization's members and so it can only see members that have publicly exposed their membership with the organization (this is a github setting that is a little tricky to configure).
- Re: Substrate / Spark extraction conversation
    - [Ashvin] Our team is very interested in this and I was reaching out to see if there were any updates.
    - [Ian] I can get in touch with some of the people at Intel that I have spoken with in the past about this. I think what would be helpful is a bit more context about what our goals our. We've had previous discussions about this and I want to make sure I set the context for this new conversation.
    - [Ashvin] I think I was introduced to this idea last time so I didn't know the context either. I do know that we are going to be wanting this ourselves. I've noticed that gluten's forks are more up-to-date with substrate than gluten itself. I'm hoping that by bringing it closer into Substrait that it will stay more up-to-date.
    - [Matan] Last time we said we should find the authors and get some approval / cooperation. I tried to reach around via Slack and find out who to ask but it doesn't appear that anyone is active on Substriat Slack.
    - [Ian] We might be talking about two things here. First, is that we want to update Substrait with relations that are able to represent concepts from Spark. This seems reasonable that it is something we want in Substrait

    - [Ian] The other thing is a tool that actually converts from Spark plans to Substrait plans. It's not as clear to me that this needs to belong as part of Substrait.
    - [Victor] I could imagine this being like Isthmus / Calcite. The core library doesn't care about Spark but there could be a library that connects Spark with Substrait-java.
    - [Ian] E.g. if there were a separate substrait-spark repo then perhaps that could work?
    - [Victor] The project that is in gluten is relying on a very old version of Substrait
    - [Ashvin] Yes, I think this will often be a problem. Backends will use the version of substrait-java they need and then not update if things are working.
    - [Weston] It sounds like we agree that the thing we want is a library that converts from spark to substrait?
    - [Others] Nods
    - [Weston] So would this need to be a separate repository? Or can it be a separate jar in the substrait-java repo?
    - [Victor] Keeping it in the same repo will make it easier to keep it in lockstep with substrait-java changes
    - [Weston] I agree that is probably easier. Every repo has a maintenance cost too.
    - [Victor] I like the idea of moving this into the substrait because now we can put spark-specific extensions here and get consensus on them before trying to promote them into the main substrait repo as new operators.
    - [Matan] I don't know much about spark physical plans and don't know if this approach will work here. However, we don't need it and it does not seem that gluten is using it today. What they call substrait-spark is using logical plans.
    - [Victor] I do think we can probably solve issues encountered with physical plans by using custom extensions and making those available as part of the substrait-spark module.

May 24th 2023 8AM PDT

Agenda

- [Victor] Case (in)sensitivity for names
    - Does this include function arguments/options?
- [David] Why does AggregateRel have repeated Groupings of repeated Expressions? One level of repeated seems sufficient.

Links

- Recording

Attendees

- Ashvin Agrawal
- Martan Peleg
- Victor Barua

- David Sisson
- Jacques Nadeau
- Weston Pace
- Richard Tia

Notes

- Re: Case sensitivity
    - [Victor] I was looking at substrait-java and noticed that names could be different case and I didn't know how that would be handled
    - [Victor] Mattjis mentioned that the validator is case insensitive, which makes sense, but I don't think it's clear
    - [Victor] Is there any reason we can think of where we want case sensitivity? Would ADD ever be different than add?
    - [Jacques] I think that is one consideration. Another is that we want to allow consumers to be as dumb as possible.
    - [David] Does the consumer really care about the function name?
    - [Victor] It does in substrait-java at least
    - [Weston] Same in acero
    - [Victor] substrait-validator protos mention https://github.com/substrait-io/substrait-validator/blob/3c934798c609a45b886fa805c694b5ad983acac3/proto/substrait/validator/simple_extensions.proto#L38-L39 case-insensitivity.
    - [Victor] There are a few questions?
        - How do we want to do these comparison?
        - Do we want to restrict this at the extension level to make it easier on consumers?
        - Do we want to limit the character set for names?
    - [Weston] My simple preference would be to restrict the character set to lower-case ascii characters
    - [Jacques] I think I'd prefer allowing any UTF-8 but stating that comparison will be simple binary matching. My experience has been that working with case insensitive stuff ends up leading to more complications.
    - [David] There is value still in the validator doing case-insensitive matching, it can detect if you're using two different cases
    - [Victor] Or the validator could just validate the case, that might be even better?
    - [Jacques] Let's start with "does anyone support or want ADD to match add?"
    - <silence>
    - [Jacques] Ok, how about we just stick with "plain binary matching?"
    - [Gil] That seems the best approach, given we aren't experts in languages
    - [Gil] We might also want to update the rules for option matching so we can have consistent matching everywhere.
    - [Jacques] That sounds reasonable, Victor, do you want to put together a PR?
    - [Victor] Sounds reasonable
    - [David] Do we want to recommend (not require) that everything be lowercase?

    - [Jacques] I think that's fine
    - [Victor] What about also recommending snake_case?
    - [Jacques] Maybe lets move this to the ticket now
- Re: Aggregate relations
    - [David] I was looking at aggregate relations and I noticed that we seem to have one more level of nesting than required.
    - [David] Most plans I've looked at have one group with lots of expressions
    - [Weston] I was confused by this myself originally. I have since learned that this is for an SQL concept called "grouping sets" which allow you to specify multiple sets of keys in a single aggregate pass
    - [Weston] For example, one might want the average height grouped by (country, state) and grouped by (country) and grouped by (). They could then specify all three to be calculated in a single pass
    - [David] Wouldn't the reader need to know which output row belonged to each grouping set?
    - [Weston] Yes, that is why, if there is more than one grouping set, then the aggregate node has an extra output column
    - [David] Why isn't this documented better?
    - [Weston] It's documented on the site, but without examples it is pretty confusing, and there is nothing in the protobuf
    - [Jacques] This is related to an old question too of "which is canonical source for docs? Protobuf or site?"
- Re: Annotations
    - [Jacques] Some of this conversation, and the recent ddl discussion on slack, reminds me that we need some kind of annotations for plans
    - [Victor] Couldn't that just be an advanced extension?
    - [Jacques] Maybe, but I don't think advanced extensions are present at enough places in the plan. For example, what if I want to annotate a data type (e.g. aligned vs not aligned data)
    - [Ashvin] I could also have the query (or maybe the query hash) that generated the plan as an annotation on the plan itself.
    - [Jacques] Yes, I think that would be fine
    - [Matan] This is something we need for expressions. We actually added this internally. In spark there are names at every point in the plan and so we had to add this in.
    - [Matan] Aliases are another thing we needed. We break the plan up into pieces and so we often need to know names at different points.
    - [Jacques] I'd push on whether this is absolutely needed or not? Names should be arbitrary if you can use positional arguments.
    - [Matan] I think it depends on the level of the products.
    - [Ashvin] Would these annotations be in addition to hints? Or are hints annotations?

    - [Weston] Same question, as we add these annotations, can we try and avoid situations where there are multiple “generic extension points” in the same message?
    - [Jacques] I think, in spirit, I agree with this. However, I’m not entirely sure that’s exactly true. Having an annotation rel makes it clear that a rel can be ignored
    - [Weston] These annotations are relations and not just decorations on relations?
    - [Jacques] Using expressions for example, you could have an annotation expression which has one expression as input, has the same data type as input, and can be ignored by systems that don’t care about it.
    - Some discussion about possible implementation
    - [Jacques] Let’s look at a concrete case. I have a plan, a portion of that plan is a subtree that has a particular name, maybe it’s a view. You could put this annotation on the rel at the top of the subtree but that wouldn’t be as clear.
    - [Jacques] An alias is like that too I think.
    - [Ashvin] I’m not sure I’d mix debugging and annotating a subtree in the same context. I wouldn’t use this debug information at runtime, it’s just for debugging. But view definitions are something that I would always expect to be present in the plan.
    - [Jacques] You still might want the debugging information at runtime. For example, you might want to, in an error, tell the user which line of the source query it came from.
    - [Jacques] I think we’re aligned in spirit. Let’s not build multiple systems that do the same thing. Let’s not go too crazy on the addition of extensions just for the sake of making things easier.
- Re mapping spark:
    - [Weston] Matan, are you interested in mapping spark logical plans or physical plans?
    - [Matan] At the moment we are sending the logical plan. I know that gluten deals with physical plans but we haven’t look at that as much.
    - [Matan] I noticed there was some discussion of extracting the substrait-spark component from gluten. I think that would be a good idea. I’m not sure who the Gluten people are here.
    - [Jacques] I don’t think any Gluten maintainers are here. I believe they were checking to see if anyone wanted to do this. It would be nice to see an effort from a Gluten maintainer to formally start this process just so its clear that Gluten is wanting to contribute this work to Substrait.
    - [Matan] We can help, though we are only interested in the logical part. We’ve found most of what we need. I’ve noticed some discussion of the missing points, like expand, and names, but I think it is a good starting point.
    - [Matan] We have been using ibis-substrait in our testing and noticed some things there that are not quite exactly the same which would be good to add to Spark.
    - [Matan] But I think we can lead this effort for the logical spark operators.

    - [Jacques] I think that's good. You can go ahead and create a PR, we don't need the PR itself created by Gluten, if they are willing to just comment on the PR that they approve it or give their blessing.
    - [Jacques] I'm curious to see what things you think are missing with regards to naming. I was thinking positional indices are sufficient.
    - [Matan] Sometimes we break the plan into pieces and move things around. Then Spark needs to go and find the correct column.
    - [Weston] I think we also run into trouble with names too when, for example, we run project we generate a name for the new column. Users could be relying on that name but it's not in the schema so it gets weird.
    - [Matan] We also see this in join. You could have the same name in different parts of your query but maybe they refer to a different column at different points.

May 10th 2023 8AM PDT

Agenda

- [Weston] Should we remove IEEE rounding options?

Links

-

Attendees

- David Sisson
- Victor Baura
- Ian Cook
- Jesus Rodriguez
- Weston Pace
- Jacques Nadeau
- Richard Tia
- Sanjiban Sengupta

Notes

- Re: Rounding Options:
    - …
- Dealing with slowdown of PRs
    - What is the core issue? Documentation?
- Ian to solve all issues!!

April 26th 2023 8AM PDT

Agenda

- [Victor] User Defined Type Extensions
    - Can/should you be able to reference types from different files?
- [Weston] Sophisticated syntax for return types & nullability, is it worth it?
    - https://github.com/substrait-io/substrait/pull/340
- [Victor] DerivationExpression & ParametrizedType, what are they for?

Links

- https://github.com/substrait-io/substrait/blob/main/extensions/functions_arithmetic_decimal.yaml#L17
- https://substrait.io/expressions/scalar_functions/#return-type-expressions
-

Attendees

- Victor Barua
- David Sisson
- Weston Pace
- Jesus Rodriguez
- Richard Tia
- Ian Cook
- Jacques Nadeau
- Sri Nadukudy

Notes

- Re: User defined type extension
    - [Victor] User defined types can be defined in YAML files. However, in the places where we reference those types, we only specify the name. It isn't clear how we would reference a type in a different file.
    - [Jacques] This makes a lot of sense and I think we should come up with something. There was some discussion on this in the past but I don't recall exactly what we came up with.
    - [Victor] I think we will also need external references if we want to start recording fallback expressions (expressions that describe how a function can be decomposed) so that we can reference other functions in other files
    - [Victor]
    - [Jaques] How does Arrow handle custom types.
        - [Weston/Paraphrase] They are defined internally, not available as YAML file. Arrow consumer knows how to handle them, but it's out of (Substrait) band.
    - …

    - [Victor] One complication this will introduce (allowing references to other function/types in extensions) is that extension parsing will now need to check for the validity of references.
- Re: Sophisticated nullability in return type
    - [Weston] There is a PR in place to discuss the return type of coalesce. The return type is nullable only if every input is nullable. Do we want to add special syntax to represent this concept (and a similar concept that returns nullable if any input is nullable)?
    - [Jacques] We have a special language for coming up with complex return type scenarios. It's how we calculate the return type for decimals. We should use that instead.
    - [Weston] That makes sense
- Re: DerivationExpression & ParameterizedType:
    - [Victor] This conversation reminded me of these two classes that I saw in the code and I didn't fully understand. Is this what they are for?
    - [Jacques] Yes.
    - <Explanation of how decimal type derivation works today and how these classes are involved>
    -

April 12th 2023 8AM PDT

Agenda

- [David] Basic binary to text plan converter final PR is out for review
    - Sample Converted Text Plans or you can run planconverter yourself
    - Looking for thoughts on the format
    - Will likely have some minor tweaks as the work on the text to binary parser work is still ongoing
    -
- [Dane] PySubstrait proposal
    - Should we move forward with it?
        - Overall, leaning yes
    - If so, do we want to bootstrap with the existing repo?
        - We will start a repo named "substrait-python", can open PRs against it
    - Naming? Substrait vs PySubstrait vs …
- [Victor] SingularOrList and IN
- [Victor] Parameterized type signatures for collection types.

Links

-

Attendees

- Victor Baura
- Ashvin Agrawal
- David Sisson
- Richard Tia
- Dane Pitkin
- Sri Nadukudy
- Weston Pace
- Jacques Nadeau
- Gil Forsyth
- Matthijs Broddel
- Ian Cook

Notes

- Re: Text Format
- Re: PySubstrait
    - [Dane] I've proposed a pysubstrait on the ML. There has been some discussion there. Interested in any other feedback.
    - [Jacques] I haven't seen it but I am very supportive of it. I know there was some initial thought we could just use the protobuf but I do think there are some useful utilities here.
    - [Victor] Coming from Java I can say that working with protobuf has been painful. Even just having a hierarchy of generated code and it was very useful.
    - [Dane] I do think one long term thing we could do is build out this foundation in PySubstrait and, once it's ready, maybe remove some of the redundant bits in Ibis in favor of PySubstrait
    - [Jacques] I think we should generally expect this. There will always be people that want to work directly with the format
    - [Weston] We had something similar with C++/Acero. We worked with the proto directly. I think that is more common when someone already has a product and they are adding Substrait as "another format". But when someone is getting started with a new project then having PySubstrait is very helpful.
    - [Dane] So what are the next steps for getting this going?
    - [Jacques] I can go create a repo. What should we call it?
    - [Gil] Let's prefer substrait over pysubstrait
    - [Ian] Agreed
    - [Jacques] Repo created, so feel free to create PRs
    - [Weston] I think the second piece you will need is a committer willing to oversee and get the initial PRs merged in. I think Gil has volunteered on the mailing list
- Re: SingularOrList
    - [Victor] I was pretty confused by this message. It appears to be "in" (as in, "x in y"). Why is this a dedicated AST node instead of a function?

    - [Jacques] It could be a function. However, it is a very common pattern to have an equality between an item and a list of literals.
    - [Jacques] If there isn't a standalone construct you also run the risk of users thinking they need to do "a == b || a == c || a == d" instead of "a in [b, c, d]" because they didn't realize the function existed. There are rewrites and optimizations out there to apply this optimization too so having the pattern "properly recognized" is helpful here.
    - [Victor] I agree, and I do think this is going to be useful for what I am doing. I think it does complicate the optimizer a little.
    - [Jacques] You can always still create the list of or expressions if you want. E.g. you don't need to use subqueries if you want to decorrelate them all yourself.
    - [Jacques] Multi-or-list is a similar pattern as well and one that isn't so easily mappable to a function (you'd need constant arrays as an argument type which is doable but more complex)
    - [Victor] I think this is good. We probably want to add more comments around what they are used for and how they are useful.
- Re: Type signatures
    - [Victor] Why don't we include type parameters in the short names?
    - [Jacques] The goal was to keep it simple. We want to make these plans as easy to consume as possible. It is a sort of constraint. We had earlier passes on more complicated things but it every time it ended up being way too hard to work with programmatically.
    - [Victor] I think that's ok, I can still find the actual types in the YAML if I need
    - [Jacques] Yes, one thing to remember is that the short name is effectively a lossy transformation. It's not meant to be parsed back to its original form and more just meant as a unique anchor.
- Re: Web site
    - [Jacques] At a minimum I want to fix up the "current status" section of the site. However, we've also talked before about making the site more approachable for users (and not just maintainers). For example, "Is this good for my use case? How do I get started?"
    - [Victor] I think, getting started, Isthmus was really helpful and, now, I've found ibis-substrait to be very helpful
    - [Jacques] Do we want to put those example text format queries on the site somewhere?
    - [Dane] I've been ramping up recently and I have been missing a bunch of "cookbook" examples.
    - [Jacques] If you could write down all the things that weren't immediately obvious, these could form the seeds for an initial set of recipes, and I think that would be very helpful to people. E.g. until the singular-or-list convo came up on slack recently it never even occurred to me it wasn't obvious.
    - [Ian] It's easier to come up with common questions and answers. I think it's harder to integrate those into a meaningful web page that is well formatted. One

thing I heard from a colleague at a conference is that he was still getting very common arrow questions even though arrow has been around 7 years
    - [Jacques] I think we recently had a query editor tool demo'd, maybe we can get that linked into the site
    - [Weston] I was delayed getting it into a public repo but that's done now and I'll make sure to get a PR up linking that in.
- Re: Physical plans
    - [Weston] There's been challenges as we figure out how to represent physical plan (e.g. spark physical plans) in Substrait. Curious if there are opinions on that.
    - [Jacques] I think it will always be a challenge that we have "I need these four fields, how do we get them?" Which is ok but we aren't getting concrete definitions of these fields which I think is essential.
    - [Jacques] E.g. I find isPartial to be the most confusing discussion at the moment because I don't understand still the meaning of the field. Somehow we need to get to the heart of "what is the behavior that is occurring" without using impl-specific terminology.
    - [Ashvin] I can add some SQL context maybe. We have an optimizer, etc, and scheduler. The scheduler uses this info "is it partial" or not to split a plan into multiple plans and materialize it.

March 29th 2023 8AM PDT

https://meet.google.com/pwj-bvuj-fwc

Agenda

- Discuss proposed new committers
- [Victor][substrait-java] PR w/ support for Advanced Extensions, Table Extension and Extension Rels
- [Richard] Demo support matrix generated from integration tests

Links

- https://github.com/substrait-io/substrait-java/pull/135

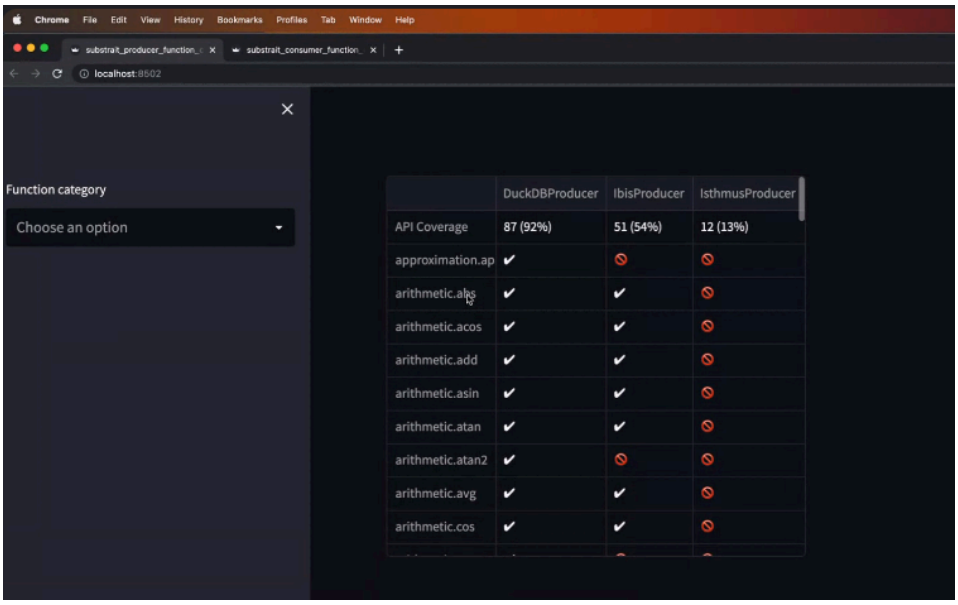
Attendees

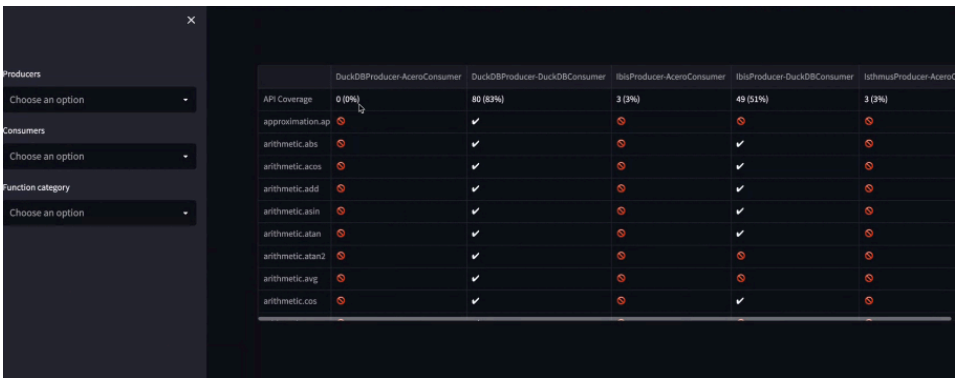
- David Sisson
- Richard Tia
- Vibhatha Abeykoon
- Gil Forsyth
- Victor Barua
- Weston Pace

- Matthijs Brobbel
- Sanjiban Sengupta
- Jacques Nadeau
- Ian Cook

Notes

- Re: advanced extensions, etc. in Java
    - [Victor] PR available that makes it possible to use protobuf any in Java in the various places that need it.
    - [Jacques] I will look at this, I have been a bit under water at the moment
- Re: state of the strait
    - [Jacques] Thanks for posting this. I think it's valuable to have feedback. I'd encourage others to chime in and mention places they think things are doing well and places we can improve.
- Re: Lambdas (function and plan transformations)
    - [Jacques] I am curious if any updates have been done here. I know at one point Weston was starting to look at this
    - [Weston] I am still in the "starting to look at this" stage.
    - [Jacques] Did James ever share the work done here with looking at the language from cockroachdb?
    - [Weston] No, I don't think so, though I did review it.
    - [Jacques] Ok, I think that would be good to do. I'll work on it.
    - [Victor] What exactly is the lambda being discussed here?
    - [Jacques] The idea is to be able to express that one function / plan can be represented by another (maybe simpler) plan or set of functions.
        - For example, decomposing window functions into window rel or decomposing an aggregate rel with grouping sets into an expand rel. It's similar to "rules" in Calcite
    - [Victor] Does this mean we won't include composite operators and just expose them as lambdas / extensions?
    - [Jacques] I think the idea is that it is still useful to have the composite operators defined.
    - [Victor] So we push the transformation down into the consumer?
    - [Weston] Yes. But not necessarily the consumer. It could be a third party tool that asks a consumer what it is capable of. The consumer could answer with some known composite operators or it could answer with extensions and transformations that can be used to create those extensions. The third party tool could then take this knowledge and translate a query.
    - [Weston] I haven't heard "composite operators". Is that a standard term for this sort of thing?
    - [Jacques] I have heard "composite" operators in cases where "scan -> filter -> project" gets joined into a "scanfilterproject". However, I think, in many cases, it is more decomposition (e.g. aggregate decomposing into expand + aggregate).
- Re: Support matrix & integration tests

    - [Richard] We have been going through the extension functions and trying to see which producers can produce plans with those functions. We've been starting with SQL queries (or Ibis expressions). Once we create these Substrait plans we are then going through the consumers and figuring out which consumers are able to consume these plans and get results.
    - [Richard] This is still a work in progress so don't be surprised if some consumers/producers have low support.
        - 

        - 

    - [Richard] Still a lot of holes to work through but we are working on some of them. For example, Acero doesn't consume plans from DuckDb because it doesn't yet support projection in the read rel.
    - [Richard] This is currently an app that runs on streamlit. Eventually, once the numbers get a bit more solid, we'd like to embed this maybe in the main Substrait site
    - [Victor] You might be able to improve the Isthmus numbers if you update. I recently added support for many of these functions.
- Re: Extension functions:
    - [Weston] We started by implementing a basic set of common functions in the core repo, per the scope described here: https://github.com/substrait-io/substrait/issues/307

    - [Weston] Efforts to add additional core extension functions are raising some questions about what our scope/charter for core functions is, i.e. what the criteria are for when a function should be added to the core YAML in the substrait repo vs. defined in custom extensions outside of the substrait repo
    - [Ian] The scope described in #307 was intended to serve as a starting point. It was not intended to be exhaustive. There is at least another 20% more functions that seem like obvious good additions based on previously discussed criteria (e.g. it exists in at least 2 widely used systems) before we will really encounter questions about appropriate scope
    - [Victor] The way the functions are organized into files / sections is also something we should give more consideration to

March 15th 2023 8AM PDT

Agenda

- [Sanjiban] Substrait-Fiddle: Tool to code or upload a substrait plan, validate it and generate a plot

Links

- https://github.com/sanjibansg/substrait-fiddle
- https://substrait-fiddle.com/
- https://github.com/PRQL/prql/issues/738
-

Attendees

- Matt Topol
- David Sisson
- James Taylor
- Ian Cook
- Ashvin Agrawal
- Vibhatha Abeykoon
- Sanjiban Sengupta
- Weston Pace
- Sri Nadukudy
- Gil Forsyth

Notes

- Re Substrait Fiddle Demo:
    - [Sanjiban] We have been working on a Substrait fiddle which lets you enter a query in JSON or SQL and visualize and validate the plan. For now the functionality is limited but we have a lot more planned for the future.
    - [Sanjiban] <Demo of tool, see link above>

    - [Sanjiban/Weston] Long term goals include the ability to share plans, and potentially even run against data. We would also like to break the visualization into its own component.
    - [Sanjiban/Weston] We would welcome contributions and help. We will be sharing in an email once we get the repo location finalized so that people can file issues.
- Re PRQL:
    - [Ian] I want to share a project I came across called PRQL. It is attempting to create an sql-like language that aims to be engine agnostic. So there is some commonality with Substrait goals.
    - [Ian] There is an issue in their github to add support for serializing to Substrait: https://github.com/PRQL/prql/issues/738 I think this is an interesting opportunity for us. I'm curious if anyone has heard of PRQL before or has any thoughts (see links).
    - [James] Do we know what the motivation or origination of PRQL is?
    - [Ian] I do not. I think the primary motivation is frustration with SQL but I don't know what the background of the originators is. Similar to human readable Substrait, I think there is ability to add support for something like this and get the ability to quickly convert between formats they understand so they don't need to understand the details of Substrait itself.
    - [Ian] The PRQL project does not attempt to solve the problem of standardizing functions / options / etc in the same way that Substrait does. So I think that speaks to the magnitude of that challenge and is a good place where Substrait can help more to provide a backend agnostic experience.
- Re: General activity
    - [Weston] There has been some activity in a few repos over the past few weeks. There is a physical operator for window function evaluation shaping up. There is some activity in go & java. If anyone can take a look at those Java PRs it would be appreciated.
    - [Matt] I can speak a bit to the go work. I'm working on the ability for arrow-go to compute and evaluate expressions using the extended expression message.
    - [Matt] Also, we have some outside interest from a datadog developer. I was looking for some contribution guidelines for the project and couldn't find them.
    - [Matt] I did have a question regarding programmatic expression building. I wasn't sure what others were doing in regards to validation. Should I evaluate expressions and functions as they are added? Or should I defer that validation to some downstream component.
    - [Weston] I think it's ok to validate if you recognize the function but you do need to leave it open to the idea that you might never recognize the function. For example, UDFs that are pre-registered with the consumer might be invisible to the go library.
    - [Weston] What kinds of things were you looking for in a contribution guidelines document?
    - [Matt] Perhaps some mention of the CLA. I'm not exactly sure either.

    - [Weston] Maybe we can create a basic guidelines document that just says “contributions welcome” and links to the governance doc and talks a bit about the CLA.
- Re: Semi/Anti join
    - [Weston] Note that there is a PR currently available which is proposing a backwards-incompatible change to the join relation. It moves anti/semi join out of the logical join operator and into the physical hash-join operator.
    - [Ian] How does one logically express an anti/semi join then?
    - [Weston] You would typically use a subquery. So either `SELECT … WHERE NOT EXISTS …` or `SELECT … WHERE my_table.id IN (SELECT …)`. These subqueries can often (but not always) be optimized into an anti/semi join but that is a physical optimization.
    - [Weston] I did do a brief survey of some common SQL dialects. Postgres, SQL server, and MySQL all do not allow you to specify a semi/anti join. The one exception was spark SQL but I think spark SQL’s plans tend to be a little more physical in general.
    - [Ian] Impala is another dialect that allows you to specify a semi/anti join explicitly.

March 1st 2023 8AM PDT

Agenda

- [Weston] Null handling in anti-join
- [Victor] PR Reviews
    - Are there specific people to tag for each of the libraries?
- [Victor] Project Roadmap

Links

-

Attendees

- Ashvin Agrawal
- Richard Tia
- David Sisson
- Dane Pitkin
- Victor Baura
- Gil Forsyth
- Weston Pace

Notes

- [Weston] Slides on what anti-join is. Two different uses of anti-join (with and without null being included).

    - Many ways to implement this including choosing one choice and using rewrite rules to convert one form to the other.
    - [Ian]
- Re: PR reviews
    - [Victor] What is the process for moving forward a PR. If I create a PR then who should I ping and when should I ping someone?
    - [Weston] Hopefully someone notices a PR being created. However, if not, then I think a ping after a few days is pretty reasonable
    - [Victor] But who would I ping?
    - [Ian] We solve this in the Arrow project with codeowners, maybe we can do the same thing here, but would we have people to be codeowners?
    - [Weston] I think most of the projects have a person or two that can serve that role. For example, I can be the codeowner for substrait-cpp
    - [Weston] Let's send something to the ML and see if we can get this going
- Re: Roadmap
    - [Victor] Is there a project roadmap or is it based on what people are interested/working on?
    - [Weston] BIggest roadmap item for Substrait is to become more accessible. Would be great to have more example use cases to point to of "Substrait solves this problem and nothing else does" This would be a good mailing list item.
    - [Dane] Would it make sense to break Isthmus tool out into its own library? For example, that way Python wouldn't need to break out its own SQL parser.
    - [Ian] There is some tooling (GralVM) that enables embedding Isthmus as a native library.
    - [Weston] To understand your question, is there a programmatic path to access Isthmus?
    - [Dane] Background that I'm thinking of. To provide better adaption, a standalone SQL -> Substrait producer would be nice.
    - [Victor] Which version of SQL would we target. Calcite has a lot of configurability to handle that.
    - [Gil] It's better to have multi-dialect support.
    - [Wes] If we do a good job of Substrait, and a language does a good job of SQL -> Substrate, we should be able to go from Substrait …
    - [Victor] As a datapoint, our usage of Isthmus doesn't use the parser. We're converting an internal language to Substrait.
    - [Weston] There is value in every language being able to go from SQL to Substrait for development reasons. There might be value in centralizing in one SQL to Substrait converter.
    - [Dane] I think you summarized it well, I'm coming from the accessibility perspective.
    - [Weston] From everyone I've talked to as well, one of the initial challenges with Substrait is looking / getting example plans which something like the above helps with.
- Re: Text Format Discussion

    - [David] Low level format. See prior discussion on Feb 1th. PR is open in substrait-cpp. [Paraphrase]
- [Weston] Not on the agenda, but the core repo has a number of issues that seem to have been abandoned. Plan on looking through them, labeling and following up with folks.
- [Weston] Brainstorming. There's a number of different companies talking about it and using it. A lot of the different pieces have one company working on. It would be interesting to talk about things that we would like to see to work on collectively.
    - [Ashvin] Issues w/ support for various functions. Example was DuckDB vs SqlServer [Paraphrase]

February 15th 2023 8AM PDT

Agenda

- [Richard] Substrait Tools
    - Is there a repo where tools could go?
    - What tools would be useful?
-

Links

- https://github.com/richtia/substrait-tools

Attendees

- David Sisson
- Ashvin Agrawal
- Victor Barua
- Richard Tia
- Weston Pace
- Gil Forsyth
- Krystian Sakowski
- Sanjiban Sengupta
- James Taylor
- Vignesh Chandramohan
- Ian Cook

Notes

- Re: Substrait Tools
    - [Richard] I posted in the slack earlier about some Substrait tools that I have created. I would appreciate if anyone would like to take a look. The tools aggregates APIs from some various tools and allows you create and consume

Substrait plans from various producers and consumers. Do we think we can create a repo?
    - [Weston] What do you see as the eventual scope for the project? Just this one tool or other tools as well?
    - [Richard] Other tools as well. I think other command line tools can go in there as well. Perhaps something with the text format as well.
    - [David] The text format I am hoping that all libraries will be able to convert to as well.
    - [Victor] I'm not sure we need a single repo for tools. For example, the validator has some good utility capabilities. But we probably wouldn't want to move it into the same repository.
    - [James] Maybe this is a docs concern? Do we have any place in the docs where we list the various tools, e.g. the validator?
    - [Weston] I agree we don't want everything in one repo but where would something like this tool that Richard has created go otherwise?
    - [Weston] On the positive side, a repo for tools like this would be useful for people to put in small tools. On the downside it's not clear who would own or manage the repository.
    - [Victor] Having all the tools in one-repo also means that if we want to update the substrait version they are using we will likely (probably) want to bump all of them making updates harder.
    - [Ashvin] In Hadoop there is a directory for a sort of beta / experimental tools and the ownership still more or less remains with the author but others can play around with it and then, if the community wants, it can be promoted to the official tools directory. So something like that might help.
    - [Gil] I don't think my mic is working, but is the desire here for "legitimacy" via association with the substrait org, or with discoverability?
    - [Richard] I am concerned with discoverability mainly.
    - [Ashvin] I do think the tool you have described is useful as different producers / consumers have a habit of producing slightly different Substrait
    - [Victor] We have been using the validator for a while and it has been extremely useful so these tools are helpful but it wasn't obvious to discover so I think a page mentioning other tools would be very helpful.
    - [Weston] Sounds like, in summary, we definitely want a page to list tools that people can add tools. We can defer the discussion about the repository for now.
    - [David] We do probably want to come up with a set of criteria for when we want to take something into the Substrait organization.
- Re: Null handling in anti-join
- Re: Understanding complex Substrait features
    - [Victor] I have a specific question (how to do a common table expression) but I've found that there are very few (no) examples on the site. Do we want more tutorial / explanatory content on the site?

    - [Ashvin] Yes, I have found Isthmus to be a useful tool for this. When I am curious how to do something or what an example I put it into Isthmus and can get an example plan. I'm curious what others think of this?
    - [Weston] I know there are a few minor points that Isthmus and the validator disagree. Also, it doesn't produce some common variations. For example, it will not create reads with a projection.
    - [James] Regarding the validator being out of sync. This is a bit concerning. Do we know of a way we can prevent these things being out of sync.
    - [
- Re: Isthmus / Flink
    - [Vignesh] Hello, I'm new to this meeting. I have been using Isthmus. I've been trying to use Isthmus with streaming queries (e.g. compatible with Calcite streaming operations) but there is no way I can specify a table is a stream.
    - [Weston] Isthmus doesn't support streaming queries today I think but I would guess that is a feature that they would like to support. James do you agree?
    - [James] I think it would be good.
    - [Weston] Do we have Substrait operators for the various streaming operators?
    - [Vignesh] I don't think so but that would be my eventual goal. I'd like to first get it supported by Isthmus to get plans and then discuss adding to the spec.
    - [Ashvin] This is also related to some of the discussions we had regrading constraints. Vignesh mentioned monotonically increasing ids and that ends up being implemented as a constraint.
    - [Weston] I think the plan you described is pretty good. Start by generating some sample plans, create a PR with the changes you would like to see and then we can have discussion in this meeting and start spreading awareness about how these things work.
- Re: advertising
    - [Carlo] Quick question for the group. Do we have any activities ongoing to advertise Substrait? Are there presentations to conferences or communities? Are we spreading the word?
    - [Ian] I think it's very hard to get people interested in Substrait unless it is packaged in something they care about. It's very hard to get people interested in "interoperable query plans". I think we need to find a need. For example, a standard for querying dataframes or lazy dataframes and then eventually explaining that Substrait is the solution. This is the strategy I've been taking.
    - [Carlo] I wonder if we can package some of these ideas that we've found, maybe also with some benchmarking information? For example, we have tools that someone can take and then quickly generate some benchmarks and run some experiments.
    - [Carlo] For example, maybe have have some tool that has two or three different consumers and then run a few benchmarks against different engines and then promote this as a "hey, look at the performance of these different engines, by the way, I was able to run this benchmark because all these tools speak Substrait." I think there is a good community that would love to do research or play around

with optimizers. Having this tool would make it easy for them to do this research. We can almost position it as the de facto standard already and then people will believe it. Maybe this is an experimental paper at VLDB that presents Substrait and also shows some of these benchmarks and proves that these tools support it.
    - [Ian] The important thing is to understand the audiences we are addressing and make a clear call to action. What you've described sounds very actionable to database researchers and academics. But this might not be as interesting to python people.
    - [Carlo] True, I am looking at a smaller, more core audience. In other words, people that should be in this call. If we capture them while they're getting started and save them from writing a parser and they can claim compatibility with Spark, Hive, etc. that will be great.
    - [Ashvin] I like this. For example, it is very similar to a proxy filesystem
    - [Weston] One thing I have been getting interested in is using Substrait to formally catalogue the differences between two SQL dialects (e.g. spark and presto). A lot of concern I hear regarding Substrait is "you'll never describe all the differences between functions" and "you'll never get consumers to agree on physical operators".
    - [Carlo] I think we can get people to agree on the common operators (e.g. join)
    - [Weston] The challenge is when those things start getting more physical and engine specific. For example, we haven't yet agreed on constraints and we don't have very many physical relations. These will be the output of the optimizers. Without these features it will be difficult to sell Substrait to a researcher that is interested in making an optimizer.
    - [Carlo] I think we can help here. We are looking at some optimization work and will be thinking about this hints and physical relations very soon. So maybe we will see some work here from Ashvin.
    - [Victor] I like this chat, I will mention that a major reason we are using Substrait is that we don't have to write a Substrait -> Calcite converter. So, in the terms of packaging this to someone that is interested, that is a very solid and practical use case that sells me.

February 1st 2023 8AM PDT

Agenda

- [David] Text Serialization Language
    - Sample Plans
    - How readable do we want our expressions to be?
    - What should types look within functions?
    - Should options for functions be per instance or be alternative function definitions?
- [Matt] substrait-go question about pinning substrait version

Links

-

Attendees

- Ashvin Agrawal
- David Sisson
- Victor Barua
- James Taylor
- Matt Topol
- Weston Pace
- Gil Forsyth
- Richard Tia
- Ian Cook

Notes

- Re: Substrait Text Serialization Language
    - [David] Made some progress converting from proto/json to proposed format
    - [David] Some sample plans added to the bottom of the linked doc. Plans are fairly simple, don't have any complex multi-pipeline plans available
    - <share, demo of sample plan from doc>
    - [Ashvin] It looks more readable, thank you. Can you share what your guiding principles are?
    - [David] Want it to be very clear what the pipelines are. I imagine each pipeline should be a small digestible chunk. Helps to understand how each of the relations are connected.
    - [David] Want to avoid source file details making relation hard to read (e.g. if there were hundreds of files)
    - [David] Put function declarations at the bottom. In many cases readers aren't even interested (e.g. if these are just basic function definitions) in this.
    - [David] For expressions, took the minimal implementation for being flexible and able to handle user defined functions. However, not very readable. For example, <look at this complex expression in the sample plan>, it's not very readable.
    - [David] Infix expressions could help. Variables (multi-line expressions) could help.
    - [David] Still some questions
        - Note, I could mark that this sum as overflow=error but where should I put that?
    - [Victor] Stepping back, is the plan to use this for human consumption while retaining the protobuf for computer-to-computer communication?
    - [David] Presumably anything that could take in a binary plan could also take in a text plan. However, if you have a binary plan you would just stay with that. The text format should be able to specify everything a binary plan can so it should be usable that way.
    - [James] Could you share the SQL behind these plans? That might help with understanding. Also, I could give you plans for the TPC-H queries (though that might be too large)
    - [David] I wouldn't worry about the plans being too large. The conversion should be pretty fast.

    - [David] I do have the tool in a branch of a substrait-cpp fork. However, it still complains about missing things and isn't quite complete.
    - [David] Going to my specific questions: how readable do we want our expressions?
    - [Weston] What's the tradeoff? What do we lose for more readability?
    - [David] The compiler will have to do more work. Some of the operators might be ambiguous (is this the 32bit or 64bit version of multiply?) Maybe some confusion with the types. So then the language becomes a bit magic.
    - [David] Variables aren't as likely to introduce ambiguity. So main tradeoff is probably parsing complexity and understanding what is going on behind the scenes.
    - [Weston] I think it should be clear what a piece of substrate should look like in the expression. Also, any kind of learning curve should be avoided if possible (e.g. for regular expressions they can be clear but you need to know all the rules)
    - [Victor] I know that field indices can be a little confusing but with the names here it may not be as clear what each name is referring to. For example, you have field#0 and I might not remember what field that is.
    - [David] Actually, that's a bug, it should be filling in the field name.
    - [Weston] Just as a caution, filling in the field names could lead to some challenges like duplicate field names and columns that never had a name in the first place (e.g. a column added by a project but removed before the sink)
    - [David] What should function signatures / types look like?
    - [Weston] I think we should stick to https://substrait.io/extensions/#function-signature-compound-names
    - [James] I thinks some indentation in a few of these spots could help
    - [David] Agreed, I can take a look at some places
    - [James] I do think infix would be more readable. Do you think an operation and the types going into that would fully define that operation so it would be clear?
- Re: Go version pinning
    - [Victor] I'm trying to go from Java to Go and back. However, since there's no version of go I don't know what version the spec was built. Having an explicit version in go would allow me to know what version it was built against.
    - [Matt] I was looking at buf and it does appear that it can take a tag version so I think it should be possible to pin to a specific tag. So you know at least what version it was generated with. This could match the Java pin to the submodule.
    - [Matt] Though we also need some CI setup for the go module as well. Should the go version be kept in lockstep with the substrait spec?
    - [David] The C++ version hasn't been updated in a while
    - [Java] The Java version is 0.4 so it does not appear to be doing lockstep.
    - [Matt] Ok, will probably go with independent versions then. I can update the readme to mention which version the go version is built with.
    - [Weston] This doesn't solve the problem but I will point out that there is a version field in the plans themselves and it might be nice if that started getting populated.

    - [Victor] One other benefit of using a submodule is that you get access to the extension YAMLs.
    - [Matt] Yes, we just got support for processing YAML so maybe that is something to think about again. We did make the decision intentionally but maybe this could make sense to avoid a runtime HTTP query. Though it introduces some staying in sync questions.
    - [Ashvin] We recently ran into this problem as well working with gluten and had some struggle with stale extensions. This is a problem we encountered in Hadoop. Every implementation had its own version of some of these things and we ended up needing to introduce a unification layer. It'd be nice if we can address this now and avoid that.
    - [Matt] That was one of my concerns. Though, avoiding the calls at initialization would be nice it does introduce the potential for stale data.
    - [Weston] It might depend on what you're trying to do with the YAML.
    - [James] A submodule does have a hash associated with it so you can find out what version something uses.
    - [Victor] Part of why we noticed this was that there was a breaking change to the YAML so when we pointed the library at the newer YAML it failed.
    - [Matt] CI could do a parsing test against the YAML. Many of these libraries have standard extensions builtin so it becomes a bit trickier in that situation. Do we want to avoid that?
    - [Weston] Hopefully backwards incompatible changes will be rare and backwards incompatible changes to functions will be achieved by name change.
    - [Matt] True, in many cases the only real thing we need to bind to is the name
    - [Matt] To summarize the previous discussion, I will work on pinning the go library

January 18th 2023 8AM PDT

Agenda

- [Weston] Clarity on what we mean by "lambdas"
    - Approach 1: Extend YAML definition to allow declaration of "fallback implementation(s)" (expressions in YAML)
        - Requires middleware or richer consumers
    - Approach 2: Extend proto expressions to allow declaration of "fallback expression"
        - Requires bigger plans with repetitive fallback declarations (could be pruned by middleware or richer consumers)
    - Approach 3: Actual YAML in proto expressions, not useful for fallbacks but useful for custom sort functions, etc. (YAML as UDF)
        - Appears to be solving a different problem, however, this is what lambdas actually are 🙂
- [Krystian / Ashvin] Is the schema in ReadRel/elsewhere incomplete without CONSTRAINTS? I could find one discussion around schema itself in issue 51, but it does not cover constraints.

    - Like NOT-NULL, does it make sense to include other constraints in the schema too; UNIQUE, PRIMARY-KEY, FOREIGN-KEY, and DEFAULT?
    - What about potentially advanced constraints like CHECK and CREATE-INDEX?
- [Weston] What are the biggest roadblocks to people getting started with Substrait?
    - [Ashvin] Start a survey / poll on this, so that we can get more inputs?

Links

- Substrait - Text Serialization Language

Attendees

- Weston Pace
- Ashvin Argrawal
- David Sisson
- James Taylor
- Krystian Sakowski
- Matthijs Brobbel
- Richard Tia
- Sanjiban Sengupta
- Victor Barua
- Jacque Nadeau
- Vignesh Chandramohan

Notes

- Re Lambas/Alternatives
    - [Weston] I'm wondering if we want to come up with new terminology for some of the concepts described in the agenda. Option 1 is not what I think of when I hear lambda.
    - [Jacques] A lambda is a function with some arguments / capture and that is what is needed for option 1
    - [Weston] Agreed. I think I would be ok with something like "a fallback expression is a lambda" but am not sure we should refer to the feature as "lambdas"
    - [Jacques] We can look at what is missing. For example, we can't say "between is equivalent to…" and we can't say "I have an expression tree to apply to a series of items". The second case is a little less clear to me, because I'm not sure what we need to introduce there. I think we just end up using expression trees there. I think the piece that is missing is the ability to replace pieces at the end of the expression tree with more detailed implementations. The piece that is missing is the ability to explain how to map arguments to the elements in the expression tree. It doesn't seem to be a dramatically new thing.
    - [Weston] I think part of the confusion for me is that this feature needs expressions in YAML and we've currently only defined expressions in protobuf.
    - [Jacques] The confusing part for me is that it feels inverted to call them fallbacks.
    - [Carlo] I can see what you are saying. I think of this as composition of smaller pieces to great something more complex. I don't start with the complex thing and decompose. So I see it more as "construction".
    - [Jacques] Right, so I think we can have two things, fallbacks and, for lack of a better term, lambda arguments.

- [James] Where does the binding aspect of lambdas fit in? Is this part of both?
- [Jacques] Thinking more, these might both be one thing but in two different contexts. I think binding is part of both.
- [David] Maybe “alternatives” is a better term than “fallbacks”?
- [Jacques] I do like alternatives better than fallback.
- [Carlo] I do also. Alternatives is a more general concept. We have alternatives in SQL server too but you can have entire plans as alternatives. Having alternatives also allows you to represent a large plan space with a small number of nodes because a single plan could have combinatorially many realizations.
- [James] I also like alternatives, it doesn’t imply an ordering between choices
- [Jacques] I think there is still a bit of distinction between “alternatives as part of a plan” and “alternatives a priori”.
- [Jacques] I think, from the discussions I’ve seen, that there is more interest in “alternatives” than “value arguments”. Do you agree Weston?
- [Weston] Yes
- [Jacques] I agree that we want to express alternatives in YAML and we don’t have a way to do that. We’re missing a few pieces in the protobuf (e.g. argument mapping) but we’re mostly there.
- [David] The alternatives could also be in their own file. There may be a large number of alternatives, many of which we may not consider using. The YAML file could get rather large if we try and list all the possible alternatives.
- [Jacques] One thing we’ve (James et al) have been looking at is a thing called optgen. It’s a go library, optimization rules are text based dsl. Since we mention having a separate file, perhaps there is a prior art here we can use.
- [Jacques] One thing about optgen is I think a lot of their work is focused on relational alternatives and I don’t know how much they have in terms of expression alternatives.
- [James] Optgen doesn’t have a strong distinction between expressions and relations and you can do replacements at both levels.
- [Jaques] David mentioned that we might not need the alternatives to be human readable. Is that true?
- [James] I think we want something human readable just to make it easier to understand.
- [Jacques] True, that tends to be very helpful to start.
- [Victor] I do think though that we have a lot of what we need defined already (in protobuf) and so it might be helpful to stick with the machine readable piece we have.
- [Jacques] True but I think the usability piece is a concern. Once David has his text form done then we could possibly just use that.
- [James] Going back to optgen, something that they have in their match patterns is the idea of “context”. Having context in your match is very important and something to consider when looking at alternatives.
- [Ashvin] One challenge, if we split this into multiple files (YAML & proto) then it might get difficult to follow references between nodes.

    - [Carlo] I think we need both alternatives and references.
- Re: Constraints
    - [Krystian] We are encountering a challenge in translating plans with power BI. For example, we want to be able to recognize and distinguish between multi-joins and regular joins and we are missing the context we need which is to know when a field is unique. There is information in the read relation but I'm not sure if that is sufficient.
    - [Jacques] I think an important concept is the ability to define generic annotation anywhere in the plan. This "hint" information is available for processors that want to consume it but it doesn't change the semantic information (e.g. if the hints are ignored you still get the correct result, but it might be less efficient).
    - [Jacques] So then I think a lot of this general / standard information could be put in as standard hints. If there was something more specific it could be added as an extension.
    - [Jacques] We kind of implicitly have this already because we have many places that can have extensions. Maybe what we need is the ability to add extensions to data types and schemas?
    - [Krystian] I like what you are saying. One concern is that, if we rely on these opaque metadata too much we might not be able to consume plans from other engines because we end up relying too much on these extensions and other components would not provide them.
    - [Jacques] Do you want to make a proposal?
    - [Krystian] Sure, and for discussion, how would people feel if we made uniqueness a sibling to nullability?
    - [Jacques] On the one hand it makes sense but on the other hand I think it blows up a number of things. For example, types are used in the signature of functions. Nullability can influence whether or not a function can be bound. Do we need that same behavior for uniqueness?
    - [Carlo] I'm not sure I agree. I think having, for example, sortedness,would be very valuable input for choosing a sort function.
    - [Jacques] Can type carry this information is separate from "is this information used to bind the function?" That's why I'd like to put this on the type, but put it into a "metadata" section. For example, would two types be "equal" if they had different sortedness?
    - [Jacques] Let's move to a proposal and discuss it then.
    - [Krystian] Sounds goo
    - [Ashvin] If you make it a "hint" or "metadata" then it is easy to drop the information. For example, other producers might not think about it and won't provide it.
    - [Jacques] I'm not sure if "hint" is the right word but I do think you can ignore this and still end up with the correct answer.
- Re: Wrapup
    - [Jacques] I see we didn't get to discussing the text format, I took a glance, it is super interesting, I wanted to talk about it and hope others take a look.

January 4th 2023 8AM PDT

Agenda

- [David] Text format for Substrait Plans

Links

- Recording
- Teams Transcript

Attendees

- Ashvin Agrawal
- Carlo Curino
- James Taylor
- Vibthatha
- David Sisson
- Sanjiban Sengupta
- Jacques
- Matt Topol

Notes

- TBD (Weston will not be here for this meeting so if someone else can take notes it would be appreciated. Otherwise, they will be updated from the recording)
- [David] Subtrait language for plans would make it easier to express plans and try things out.
    - Jacques had a number conversations in 2022
    - [Jacques] Similar to explain plans that DBs produce today
    - Human consumability, see the shape of the plan important
    - There was a JSON conversion library
    - If solid, would be updated into documentation
    - Compression to see the structure
    - [Ashvin] Use case to see the operators used in the plan
    - Would be useful be able to go from a plan to the text format to help make a plan more readable
    - [Jacques] Text could be used as validation (an output for something not just as an input)
- [Carlos] Talked with an Intel project that wasn't considering Substrait (but should have been)

Dec 7th, 2022 8AM PDT

Agenda

- [Ashvin] Building C# client
- [Jacques] Lambdas
- [Jacques] Text format

Links

- Recording
- Teams-Transcript

Attendees

- Ashvin Agrawal
- Richard Tia
- Jacques Nadeau
- Sanjiban Sengupta
- Vibhatha
- Weston Pace
- Jesus Camacho Rodriguez
- Carlo Curino
- Krystian Sakowski
- Jacques
- Rok
- Gil
- Matthijs Brobbel

Notes

- Re: Lambdas
    - [Jacques] PR proposed by Jeroen ~2 months ago supporting lambdas. At a simplest form I would think of this as something like an aliased expression tree. So you can build an expression tree, name it, and then use that to express different things.
    - [Jacques] One powerful example of what we could use lambdas for is being able to define complex functions as a lambda composed of other simpler functions. I wanted to bring it up here. I know Weston wasn't overly excited about it but I think it would be very valuable.
    - [Jacques] For example, someone recently proposed a top-n which would be a combination of a sort and fetch. If we had lambdas then it might be nice to support this name as just a composite of these two things.
    - [Jacques] It connects to another topic which came up a bit ago, which is the concept of annotations. This concept is that it might make sense for certain nodes to have an annotation node that can provide extra information. Though in retrospect it might not be the best fit for this.
    - [Carlo] Maybe a general pattern or related idea is the ability to express alternatives in plans. So a producer can explicitly list multiple alternatives. Would this pattern allow for what you want?
    - [Jacques] I don't think so. I think what I am describing could be used in the implementation of alternatives but I don't think alternatives solve this. This puts the burden on the producer (as opposed to some middleware library).
    - [Carlo] I do think this could still work with an alternatives library. It might not be the easiest but it would be possible.
    - [Jacques] The problem with alternatives is I have to introduce the alternative in my project and can't describe it in the spec. Then I would need to insert both alternatives in the plan. I think there are two ways to think of this. One as a header and one as a part of the tree. If you express alternative as a lambda then you can encode the process and not just the materialized result.

    - [Weston] I think this discussion has diverged a bit from the existing proposal which was meant to think of lambdas as expression trees. If we're talking about sort + fetch then I think we are talking about named sub-plans (plans not expressions).
    - [Carlo] I think we will want both in the long term. Maybe they are the same implementation or maybe this is two different implementations.
    - [Jesus] I do agree that some kind of alternative operator would be something we can and should do.
    - [Carlo] We can maybe think this a bit on the side and propose it. That might help advance the discussion. I don't think this is urgent though. And maybe we do want to decouple this from the lambda conversation if we are in a hurry to finish lambdas.
- Re: Text format
    - [Weston] I was hoping we could talk about this a bit so I will just summarize some of the conversations we've seen TBD. One challenge is that SQL tends to be a very convenient and well known format for text serialization.
    - [Carlo] SQL cannot fully represent Substrait though correct?
    - [Weston] Correct, for example, SQL does not have a way of representing function options.
    - [Carlo] Another thing is that it might be nice to have some different resolutions. For example, when debugging I might just want to see the plan at high level (e.g. I want to see there is a join but don't care about the details)
    -
- Re: C# client
    - [Ashvin] Some people have started contributing to a C# client. Goal is to increase usability and simplicity of interacting with the specification. Planning on starting with ideas from the Java client but expect more PRs coming from Microsoft.
    - [Ashvin] Please share any ideas or experiences that we've learned from Java, etc.
    - [Ashvin] We think this will be very helpful to bootstrap people into substrait.
    - [Ashvin] Would be great to get chime-ins from other people. We have enough internally to really drive this but want to make sure we get input from the community as well.
    - [Weston] I will try and keep up, though I am pretty busy, so can't promise anything
    - [Carlo] To chime in, we've definitely seen some initial "scare factor" when introducing people to Substrait. We've noticed a lack of welcoming & quick start content.
    - [Carlo] As a byproduct we are trying to get people to write some of these things. If we can't get to this then it is probably important that we get some of the people from this call to start working on this. This is important for the success of the project.

- [Weston] There is a PR from Will starting this so I encourage a look. I'll also mention this to Will as he might be interested in reviewing some of these things.
- [Ashvin] Question about the C# client. Is validation something we want to be a part of the client library itself?
- [Jacques] The validator was intended to be usable cross-language. There was always a hope that it could be used programmatically at the edge and embedded in other languages. I don't think this is in conflict necessarily. That being said, there is still work to integrate those C bindings. However, we chose Rust to hopefully make that work easy and minimize the amount of security risks.
- [Gil] Yes, the validator is already being used from python in this way.
- [Weston] I would say that a client library can own validation of user inputs. For example, if creating a producer the client library can check to make sure the user isn't giving null for a field reference. However, if the client library is loading a whole plan and validating that then it seems like a good use case to share the validator.
- [Carlo] Have we measured the performance of the validator
- [Weston] Performance was not a top goal of the validator. There are some places where the validator has made choices for maintainability and usability over performance. However, I think it would be ok.
- [Carlo] Ok, benchmarking this may be something we want to take on.
- [Jacques] I wouldn't expect, from my experience, that validation costs would be that large.
- [Carlo] Front-ends can also cache these plans so we can maybe cache some of this validation as well.
- [Weston] I think the big question for me around client libraries are what sort of problems the client library wants to solve. For example, we have seen some functionality like the ability to lookup function implementations in yaml files. There are also things like "expression manipulation" which might feel more like "tools" than client library apps. However, I would encourage the development of these tools and aiming for client libraries to be toolboxes. This is something Arrow has done initially and I think we benefited from it.
- [Ashvin] Yes, for example, another tool might be something like visualization.
- [Carlo] I think having these tools will be helpful for new users as well. This is similar to my point that we need better ways to introduce new users without the scare factor.
- [Jacques] Agreed. One of my regrets is that we ended up with many repos instead of one shared repo. That makes it a little difficult to know all of the progress that is being made and all the things available. We have spoken about improving the content on the website too so that it is more friendly.
- [Weston] On the topic of visualizers I would say that it is something I am very excited about. Having a plan visualizer will be very helpful for new users.
- [Jacques] I would ask that we try and keep things a bit modular. So that people that are building tools don't feel they have to source your tools to be able to use it.

-

Nov 23rd, 2022 8AM PDT

Agenda

- [Weston] Update on ongoing tasks
    - Code of conduct
    - Diversity & inclusion
    - Technical presentations during community sync meetings
- [Weston] Development guidelines
- Spark integration (gluten patch, etc)

Links

- Transcript
- Recording

Attendees

- Ashvin Argawal
- Ian Cook
- Steph
- Rok
- Weston Pace
- Sanjiban Sengupta

Notes

- Re: Ongoing tasks
    - [Weston] Code of conduct hasn't gotten much feedback recently, might be ready to wrap up.
    - [Jacques] Might ping the ML as some have just forgotten about it
    - [Weston] Not a ton of feedback, so I will interpret this as lazy consensus and encourage people to start following or implementing suggestions.
    - [Jacques] Can we put that email on the site somewhere as a sort of diversity statement?
    - [Weston] Yes, I can do this
    - [Weston] Didn't get any feedback on technical presentations. I think there is a desire for some knowledge sharing and learning here. However, I probably don't have the time to corral this and put something together. If people want to give a presentation they can do so in an ad-hoc manner. If someone wants to organize something then there is maybe still an opportunity here.
    - [Ashvin] We could maybe start capturing snippets from talks and so on and organizing and collecting them.
    - [Jacques] That sounds good but you need the content. I would encourage everyone to go out and start creating some of this content.
- Re: Development guidelines
    - [Weston] Started a document (https://docs.google.com/document/d/1NoRSzs4ht2z6UgSxIspi392UMLDOOA6JLGsoVT6URJ4/edit?usp=sharing) to collect best practices / development

guidelines. We probably want to start putting together some process (e.g. labels, bots, etc.) to help keep development moving smoothly.
    - [Jacques] Yes, I've been using "request changes" to start and keep track of these things. We might want an idle bot that automatically closes stale issues. So are you going to propose something?
    - [Weston] I'll send a ML topic too. I might want to just have a brainstorming meeting first so I'll try and find people that are interested.
    - [Jacques] My main request would be to avoid adding new manual steps.
- Re: Spark integration
    - [Jacques] I believe there have been several experiments with spark & substrait recently. Probably the most comprehensive has been the gluten project.
    - [Jacques] The gluten project has created some logical and physical mapping from spark to substrait and they would like to contribute it but don't have the effort to maintain it.
    - [Jacques] One caution is that this may end up being a lot of code to maintain and if the maintainers don't have effort to upstream then the upstreamers might get stuck with this work.
    - [Jacques] On the other hand, this is rather valuable, and I think bootstrapping some spark integration would help generate interest (e.g. finishing a spark connector is less work than starting a spark connector).
    - [Jacques] I wanted to see if there was some interest in here and what others opinions were.
    - [Ian] There is some interest here from Voltron Data as well as we have some interest in adapting to spark APIs and allowing tools that have been written to expect spark to continue working even if spark isn't in the loop. I think this is slightly different than gluten which still expects to use spark in a number of places.
    - [Jacques] Do you think if people started here we might have some help? I think it's a good idea for the project to have connections to spark. The question is whether it's important to enough people that we can get something done.
    - [Ian] I'm not entirely sure. There's interest but it is a challenge to make sure everything gets upstreamed.
    - [Jacques] What do you mean?
    - [Ian] When someone is working on something that needs to be upstreamed it can become a chore to upstream that work and so the moment a project needs to diverge and fork then it does so (because it needs to keep moving) and the upstreaming falls behind.
    - [Weston] One challenge I've seen with upstreaming is that it gets a lot harder when working on "generic language" projects (e.g. substrait-cpp or substrait-java) as these projects don't yet have a well enough defined goal and it's generally easy enough to just grab the protobuf and start with that. I don't think this necessarily applies to the spark effort.
    - [Ashvin] I would disagree that the specification / protobuf is a very easy tool start with and I think there is still a very good case for general abstraction libraries.

- [Jacques] This has a lot of parallels with arrow. Originally people just wanted to start with the spec and they ended up reinventing things that everyone else had to solve. It took time to get to the point where using these libraries would be helpful though, but once they fixed things, then people started to really favor the libraries instead of the spec because it can smooth over a lot of the really confusing pieces. The spec has a lot of stumbling blocks and abstraction libraries could solve these.
- [Jacques] For example, there is an ongoing discussion on something called expand. I haven't fully got my head wrapped around it however I think there is some chance that this is maybe just a different way of thinking about the spec. I think, once we have a better understanding of the libraries, they will mature and be more used.
- [Ian] Tying this back to spark I think there is a bit of a related dynamic at play. As you add bindings with higher level abstractions you build up choices and opinionated choices and incompatibility. Then I think you often need to come back to the standard to make different choices. E.g. for arrow we've built a number of choices when building arrow-c++ and now we see nanoarrow starting up. So I think a modular approach is the best route to go.
- [Jacques] I think modular is good, but it doesn't necessarily require different repos. You can have multiple modules within a repo.
- [Steph] Why the sudden interest in spark? Is this because of an existing project or sparks popularity? Because there are a lot of other frameworks (e.g. flink, etc.) I like the idea of getting 50% done and then leaving work remaining for others.
- [Jacques] I think we should support whatever we can get momentum around. In the conversations I have with people about Substrait I find that 50-75% of these involve spark in some way.
- [Jacques] Also, we actually have a request here from the gluten folks to help upstream their work.
- [Jacques] But yes, I would like to see Substrait integrate with many different platforms (e.g. snowflake)
- [Steph] What do you think would be the vision of Substrait & snowflake?
- [Jacques] For sundeck, we are trying to help people understand their workloads. So it would be valuable to turn snowflake plans into something that we can then take and use as input to tools that help users understand their workloads.
- [Jacques] Postgres is another connection that has come up before. Clickhouse has come up. However, these discussions have been smaller than the spark discussions
- [Steph] Do these projects exist?
- [Jacques] Just in the theory stage for the most part. Some things might also exist that we don't even know about. A lot of Substrait work happens outside of Substrait (e.g. Arrow has a lot of Substrait code but it isn't in the Substrait repos).
- [Steph] So the question is then, do we want to absorb and upstream those projects and support them and how do we make those decisions.

    - [Jacques] I think a lot of the time too it is a good thing that these projects are housed outside of substrait. For example, it is good that ibis-substrait is not a substrait project because then the ibis team maintains it and integrates with it and you can have more reliability it works and doesn't get out of sync with Ibis.
    - [Jacques] That being said, we should still make sure we're paying enough attention to the core repository that it doesn't become difficult to get patches through.
    - [Ian] You mentioned that it would be good to have information about projects that use Substrait. I think that sounds really useful and I'd be happy to open a PR on that.
    - [Jacques] That's great. There was some discussion on reorganizing the site a while back. I'll probably be taking on that reorg.
    - [Jacques] In the past the focus of the site was very much "this is why you should use substrait" and not so much "this is how you can use substrait" and so I'd like to do some of that shift at some point.

Nov 09th, 2022 8AM PDT

Agenda

- [Ashvin] Backup meeting coordinator (e.g. during holidays/vacation)

Links

- Transcript
- Recording

Attendees

- Ashvin Agrawal
- Carlo Curino
- Richard Tia
- Sanjiban Sengupta

Notes

- [Action item] Start a thread if users are interested in Graal
    - Spin a separate module for Graal?

October 26th, 2022 8AM PDT

Agenda

- [Carlo] Diversity of our community (as gender/race etc.)
- [Carlo/Ashvin] Simple tutorial or guide
- [Weston] General philosophy for extension yamls when functions differ between language standard libraries and existing data engines (e.g. string split / max_splits)
- [Ashvin] PR for Apply Rel

Links

- Meeting Transcript
- Recording

Attendees

- Ashvin Agrawal

- Carlo Curino
- Jacques Nadeau
- Jesus Camacho Rodriguez
- Ricahrd Tia
- Vibhatha Abeykoon
- Weston Pace
- Gil Forsyth

Notes

- Expanding site docs
    - [Ashvin] Suggested adding docs for a better getting started experience
    - [Jacques] We may want to alter the top-level nav. At the moment it is all spec so maybe take what we have in a “spec” tab and we could make room for other kinds of content.
    - [Jacques] But this seems like a good idea
    - [Carlo] Maybe also some more information about the things we are doing and doing in other projects
    - [Jacques] Yes, this can work with the new top-level nav. So we can have a section for ongoing work, a section for getting started (for users and for contributors)
    - [Carlo] It would be good also to help add comparisons or examples of why someone would want to use Substrait. For example, smaller plans, faster processing. We should point out that this isn’t just a common format for plans but also a good implementation of a format for plans as well.
    - [Jacques] Maybe a blog post covering some of these things you’ve learned?
    - [Carlo] The info could come from multiple places (duckdb, arrow) and mostly we just need numbers showing that this implementation has usable performance
    - [Jacques] Yes, I think that we have made some good decisions (don’t carry the schema around in all the nodes, use integer based references everywhere) which might not be obvious to people and should help keep plan size down.
    - [Jacques] I agree that it makes sense to have some metrics to keep people confident.
    - [Jacques] Have our tests validated any of this?
    - [Richard] Haven’t really been measuring this.
    - [Carlo] We have done some internal comparison and Substrait has been an order of magnitude better in some cases.
    - [Jacques] In my experience people often spend less time micro-optimizing plans since that is generally thought of as a different level of abstraction. Some of the ideas in Substrait have been inspired by the use case of more efficient planning. I hope someday we can get to a blog post demonstrating some of this stuff.
- Direction for extension functions when different between languages & data engines
    - [Weston] Was looking at string split function and noticed that duckdb, mssql, postgres all did not have a max_splits function. On the other hand, python, JS, Java, all did have a max_splits function. Which do we pick?

    - [Jacques] As a general paradigm we could have more grouping of extension functions (e.g. dataframe specific functions, engine specific functions, etc.)
    - [Jacques] In addition, there was a function extensions question regarding timestamps that had gotten me thinking about overloading, or at least default arguments (alias with pre-added arguments)
    - [Jacques] Could do this with lambda. String split without an argument is just a lambda that calls the other string split but provides the default argument. Makes it easier for an engine consumer
    - [Weston] That fixes the problem where an engine has a 5-arg version and doesn't want to map the 2-arg version. It doesn't help the case where an engine only has the 2-arg version.
    - [Jacques] Yes, in that case, you probably need two functions. One with the reduced args and also the more complete version.
    - [Ashvin] It's less ambiguous and more inclusive if we have both functions so it is clear.
    - [Jacques] If you only support the 2-arg version then you can still support the 5 arg version under certain circumstances (e.g. when the other 3 args match your default).
    - [Weston] Right, so in string split, the "default" is unlimited splits and so you'd be fine as long as the max_splits argument is unlimited.
    - [Jacques] You could technically also fix this with a lambda by splitting and then stitching back the remainder or slicing.
    - [Jesus] Is this discussion of lambdas theoretical or…?
    - [Jacques] There is a PR for lambda expressions.
    - [Jacques] But anyways, in this case, I think we still want the two functions. Maybe one of the functions has a lambda.
    - [Weston] Ok, I think we're good here. We can have both functions. Maybe one is replaced by a lambda at some point but we can worry about that later.
    - [Jacques] It might be a good idea to have a space to put this reasoning in the YAML docs themselves.
- Inclusive community
    - [Carlo] We've got some great people here but pretty abysmal diversity. I've noticed that, once you get a large non-diverse community, it is much harder to add diversity later. Do we want to try harder to get more diversity?
    - [Weston] It is a challenge. We need more outreach in general so it makes sense to focus on trying to do that in a diverse way to get a more diverse set of opinions.
    - [Carlo] There is also internal outreach or outreach within your company or organization. We still want to make sure it makes sense for the individual and the project but steering this a little bit makes a lot of sense.
    - [Carlo] Externally, yes, we can do more diverse outreach. For example, Jacques could give a presentation at the top university of Rio de Janeiro where there is a higher chance of getting diverse candidates.

    - [Jacques] It is very important to me. It's very hard but we can do it. As a community do we agree that it is important? I know I think it is but I don't know that we've asked the community and we should maybe do that. Not just the people in the call but the community as a whole. I know that there are people that disagree with me entirely. This sort of thing can turn into a flame war. For example, it is common for people to believe that decisions should be blind to these things, though that just perpetuates the problem.
    - [Carlo] I don't think we can be blind. There is evidence and studies out there that show that people are not blind. For example, studies that show that, once gender is made known, opinions of technical expertise change.
    - [Carlo] This is a non-negotiable deal breaker for me. If this is not a value for the community then I don't want to associate with it. I don't mind the specific method but I think the value of diversity and inclusiveness is not something we can compromise on.
    - [Jacques] I don't think that is the controversial part as much as the action that is taken to address this. Sometimes people like the value but don't think any action is warranted.
    - [Jacques] I think there is a conversation here that is good to have. It's easy to get people to agree that we don't want to be misogynist. It's harder to get people to agree to specific actions that we take to increase diversity.
    - [Jacques] As an example, when you have cultural norms, we often have a list of things we do and don't do that accompany it. It would be good to have a list of actions here.
    - [Carlo] What's the next step?
    - [Jacques] Let's start a discussion on the ML and reach consensus and then add stuff to markdown.
- Apply operator
    - [Ashvin] I was hoping we could have some discussion of the apply operator. I've created a PR. I don't know if people have had a chance to take a look.
    - [Ashvin] I think the apply operator is pretty common and I found a paper recently that covers some real world cases and uses the apply operator in some of them and so I'm starting with those.
    - [Ashvin] This is kind of like a join but instead of a right rel there is a correlated subquery.
    - [Jacques] One challenge for me is that I haven't recently spent the needed time to get into this space. The main question I have is "is it distinct from join? or is it another subquery type?" We have a number of subquery types already. Why is this subquery pattern important enough to have a relation. We spent a long time on subqueries and maybe had some discussion on this but I don't remember. When I look at your PR it is not clear to me how your PR can achieve your query.
    - [Ashvin] Apply is not a join because there is an inherent correlation between the two queries. Can a subquery be a table valued function?
    - [Ashvin][Jacques] Some technical discussion of the feature which is difficult to capture in notes.

    - [Jacques] Is anyone at Voltron working in this space?
    - [Weston] I'll check but I don't think so. This topic is hard for me to give an opinion on since I don't know much about it. Do we want to have some regular information sessions / technical meetings?
    - [Carlo] Maybe we can start by splitting the sync meetings to 50/50 so they are 50% general discussion and 50% technical content. Or maybe we alternate between technical and community topics.
    - [Ashvin] I prefer the first
    - [Jacques] I'm fine with either approach. Do we want to maybe use something more formal for publishing the agenda ahead of time like a meetup?
    - [Carlo] I think meetup is fine as long as it is also on the calendar
    -

October 12th, 2022 8AM PDT

Agenda

- [Ashvin] Apply operator
- [Weston] Separate Java library from Isthmus
- [Jacques] CLA feedback
- [Jacques] Updates on governance public/private

Links

- Recording
- Transcript

Attendees

- Ashvin Agrawal
- Weston Pace
- Carlo Curino
- Richard Tia
- Vibhatha Abeykoon
- Jacques Nadeau
- Jesus Rodriguez
- Gavin Ray
- Steph Wang
- Adam Kennedy

Notes

- Apply operator
    - [Ashvin] Jesus and I were talking about the apply operator. An operator which is like join but not quite the same because the tables are correlated. There are a lot of optimizations built around apply. Is this operator missing from the spec? I don't think it's a new join type.
    - [Jacques] I'm not quite sure what apply is.
    - [Ashvin] Apply is like join. It has two inputs. For each row in the left the right is applied. The right is a table valued function and it is applied to each row in the left. A table valued function is a function that produces a table (e.g. 0+ records).

    - [Ashvin] Generally subqueries / queries rely on nested loops. Apply has the look and feel of both a join and a subquery. I'm at the point where I think I understand it enough to start an issue but given existing issues around subqueries and joins I wasn't sure if there was already something going.
    - [Gavin] Apply is also known as lateral join or cross apply
    - [Jacques] Yes, I think having a discussion on this sounds good. It's possible we do have some of this. I believe there might have been some previous discussion on this with Velox.
    - [Ashvin] Sounds good. This is an important logical operation for us and so I wanted to make sure and raise it.
    - [Jacques] Yes, when we first discussed correlated subqueries we started with something much more general and then we ran into trouble trying to get it to map to sql. I have a suspicion the same thing might happen here. Possibly it fits with some other pattern, e.g. a table function pattern, but you cannot decorrelate that.
    - [Jesus] Agreed. I think there are some cases that cannot be correlated and some cases that are tough for calcite and optimization and there are some cases where it doesn't make sense. I think discussion is good and we can provide some examples and drive it that way.
    - [Jacques] Isthmus originally decorrelated but I think that's been turned off.
    - [Gavin] This might already be possible but my organizations use case for this is when relationships are arbitrarily nested (e.g. artists -> albums -> tracks) and I want all artists where … and for each artist I want tracks where … and …
    - [Gavin] This leaves you with two options .A lateral join on a subquery gives you offset and limit or you have to do some kind of hacky row number thing so you can do limit and offset on that. It should be possible to emulate with a row number window function but the big appeal is that it basically gives you a "for each" loop.
    - [Jacques] Agreed, let's take this to discussion.
    - [Ashvin] Let me share a link https://explainextended.com/2009/07/16/inner-join-vs-cross-apply/ which is a very good example of what Gavin is talking about. This is a more specific operation but it has some nice optimizations.
    - [Jacques] With the caveat that it's been a while since I spent a huge amount of time with SQL server, I think it is good to have a generic definition and map it to SQL server and others.
    - [Carlo] This looks like a tradeoff between avoiding complexity in Substrait and forcing consumers to do complex optimization to get the right thing. This seems like a case where we can offer users an ability to help guide the optimization and it is valuable even if it adds complexity.
    - [Jacques] Agreed. We don't want people to have to take knowledge out of a plan to convert it to Substrait. So even if there is a generic way to express it then it may not be worth it. We ran into this with decorrelation and this feels like a variation.
- Java library

- [Weston] We're starting to look into a need for us to take Calcite plans and convert them to Substrait. Possibly taking Isthmus and creating a more simple library focused on serialiation / deserialization. We have some Java engineers but none with groovy/grails exp.
- [Jacques] There is already two libraries, core & isthmus. The core library has serialization / deserialization from pojo to Substrait but doesn't deal with Calcite. The Isthmus module on the other hand is 90% conversion to/from Calcite but also has the Isthmus piece. So maybe it makes since to split those last two things out.
- [Jacques] The build is in Gradle. I agree that the Java version requirement is probably too high and so if we're going to split this out then lowering that makes sense. Pushing to maven central is good. Separating the CLI from the core calcite conversion stuff is good. I can probably open tickets for those things. I don't think they're massive pieces of work.
- [Weston] Who would be good to ping for reviews?
- [Adam] What we're bootstrapping at the moment is a fork / experiment from a few months ago (backported to Java 8). I don't think anyone disagrees that we should drop the 17 preview features. How do you feel about going earlier than Java 17? In my experience getting to 11 wasn't too bad, but getting to 8 was a disaster.
- [Jacques] I think 8 is EOL
- [Andy] Yes, anyone left is probably on Oracle support packages.
- [Jacques] But, Spark technically goes back all the way to 8, which is a powerful argument. Have you looked at my comments on Jabel (sp?)?
- [Andy] I think it would be nice to avoid another code-gen step if possible.
- [Jacques] It's technically a compiler. Most of the newer features are all syntactic sugar and not new byte code. So the new Java compilers know how to deal with the new syntax and they have features which allow them to compile to earlier Java. I haven't looked at the intricacies of Jabel but it's really just turning off some "convert to newer byte code" features of the compiler.
- [Jacques] I can see that this new tool would be painful if you were rebuilding the build system but it should be invisible to new devs.
- [Jacques] I think it makes sense that we target 11. For CLI / C library / end products, I think we can use Java 17. However, the more core components (those that will be dependencies for other projects) we should target 11. We don't have to figure out how to get there in the call.
- [Andy] A good first step would be to probably remove preview features from the source and then we can look at Jabel / etc.
- [Gavin] Can you just use 17 for your dev environment and target 11 when you compile?
- [Jacques] Java does not work that way unfortunately. The closest thing is probably Jabel.
- [Carlo] So, if I understand correctly, the proposal is to avoid the new preview features, stick with core 17 in the source. Then add Jabel later for 11 support.

    - [Andy] I don't know that it makes sense to release both. Part 1 avoids the preview feature. Part 2 is to make sure we target 11 (e.g. with Jabel).
    - [Andy] Yes, removing these preview features and publishing to maven central would be the highest priorities because they will allow us to use these libraries in our build systems.
    - [Jacques] I think those will be pretty straightforward. I still think we might want to consider 8 if we get it for free (e.g. Jabel)
    - [Andy] This does introduce a problem with byte buffers that is somewhat tricky. So my instinct is to avoid 8 to avoid this problem.
    - [Jacques] True, I was just thinking about the Spark community
    - [Andy] I imagine the java 8 question is probably coming up in Spark too.
    - [Jacques] For reviews, Jinpeng and James have done a lot of work in the Isthmus package so maybe CC both of them and myself.
    - [Ashvin] I can also help some if you'd like.
    - [Humorously] Jesus got volunteered by Java but Carlo says he is reserved for convoluted C++
- CLA
    - [Jacques] I've posted a generic CLA and activated a CLA assistant. I have seen a request for a corporate CLA so I will track down some boilerplate for that. Have people gone through the process? How do they feel about it?
    - [Carlo] I tried to click on the assistant and it was requesting way too many Github permissions so I'm not sure if that was the right link.
    - [Jacques] I think you might have started down the path of setting up CLA assistant on your repo. If you follow the link from the PR it should ask for a very small amount of information.
    - [Carlo] Going through this right now, odd that it asks for age in years instead of birthdate, but it seemed pretty easy.
    - [Jacques] Good, the entire thing is run from a gist. My goal has been to minimize the headache from this as much as possible and keep it self managing. Any concerns with the approach?
- Governance
    - [Jacques] I think discussion is settling. I discussed this to move towards a vote with the proposed SMC. We made some small changes (adding a private list dedicated for harassment issues and one for security issues). Comments / thoughts?
    - [Carlo] I haven't checked if you replied but my question was the SMC is stated to focus on the health of the community but the SMC seems to be reaching down a bit further. Either approach is fine but it should be a bit clearer.
    - [Jacques] I will think about that and include it in my wrap-up email. It's an interesting thing. We don't have release votes, which is the way Apache solves this, and because we don't have this we don't have as much structure around some things.
    - [Jacques] I would like there to be more people on the SMC. Historically though we have had a bunch of tactical contributions but not too much sustained. I

agree with the general feeling but not sure I see a great solution. Adding a bunch of people who may or may not be here for the long term or are not here for the general success of open source projects may not be the right solution. Even the committer list today is a little weird because some of the committers aren't very involved right now but they were added as committers to specific repos at one point.

- [Carlo] I think it's just a clarification. There is a point that specification modifications or breaking changes require an SMC vote.
- [Weston] I think it's just a wording thing. There is a paragraph "the SMC is about the health of the community" and another that is "the SMC is required for spec / breaking changes". So we can add a sentence to the first "and the direction of the project" or we can change the second to not require an SMC vote.
- [Carlo] And if we want to expand the SMC role to be responsible for the direction of the project, then we should probably grow the project. If it's just the health of the community, then it probably doesn't matter too much.
- [Jacques] I think we want to grow the size of the SMC regardless. Also, maybe this will become clearer as we go forward and not require SMC votes for specification changes.
- [Carlo] Makes sense and the alignment on project success is pretty successful. There are other options but we want one universal option. We want this project to be as welcome and inclusive as possible. E.g. we should make sure that Adam's use of Java can be successful and so making him happy should be our purpose.
- [Jacques] Or make it so that Adam can make himself successful.
- [Weston] I do think there is a short term tactical issue here. For example, a backwards incompatible change to a function is a pretty minor thing. I don't really want to get involved in a long discussion about time zones just because the change happens to be backwards incompatible.
- [Carlo] Maybe there could be a delegate.
- [Jacques] I'd push back on this. We want to be making sure that we are considering a wide variety of viewpoints, even for function definitions. E.g. right now a lot of the function definition is being driven by Voltron, and they might be biased (e.g. Voltron is currently very postgres-centric). So I don't think the SMC's role is to make the right technical decision there. Instead the idea is for the SMC to review the discussion and make sure that all viewpoints are being considered before the change is made.
- [Andy] Also, as a Voltron employee, I agree Voltron has been postgres focused, but I hope that can change, as I am very much involved with Calcite.
- [Jacques] I think we keep what we have, I'll review. I'm not keen on delegates. Hopefully a review of the discussion is not too onerous an SMC burden. We can iterate from here.

September 28th, 2022 8AM PDT

Agenda

- Governance next steps
- Optional Enum/Compat
- Enum versus Oneof Empty Messages

Links

- Recording
- Transcript

Attendees

- TBD

Notes

- [Various] Introductions
- Governance next steps
    - [Jacques] Comments have calmed down but encourage anyone to take a look. Soon, the SMC can ratify the doc as the next step.
    - [Jacques] Biggest open question is probably CLAs. We claim in the doc that people have CLAs on file but we don't really have this process figured out. We should formalize this process.
    - [Jacques] I think there is a github tool that exists already which should help here by automating collection of CLAs.
    - [Carlo] I don't know if it is there or not but I chatted with Microsoft folks that were nervous about the process and asked about their main concern. Belonging to a foundation is important but not sufficient. They would like to see that the steering committee (or whatever the equivalent is) is composed of people from a variety of companies and not dominated by any single company.
    - [Jacques] Understood and I tried to be more aware and active regarding corporate influence than Apache has been in the past. I've proposed a goal that as part of the consideration for different levels (committer, SMC, etc.) we be aware and considerate of corporate diversity.
    - [Carlo] So contribution isn't enough but that also granting these positions requires we are meeting our goals for corporate diversity.
    - [Jacques] Yes, though maybe not so strong as to say quotas. It is important that a person's corporate background and potential for corporate diversity be considered as a factor alongside the contributions.
    - [Jacques] I'll call out two potentially unusual / controversial things in the governance document (controversial meaning inconsistent with Apache). The first is awareness of corporate influence. The second is the proposal that there is no private communication in the project. I wanted to talk about this second one here too.
    - [Jacques] For example, in the Arrow project, I think the private mailing list is used fairly well. It is very rarely used and only for things that are highly controversial (e.g. around people). For example, "I think this person is being abusive, do others agree?"
    - [Jacques] On the other hand, I've worked on similar projects, where private lists are not used as well. In Apache there are a number of private lists (e.g. PMC private list) and I find an unbelievable level of poor behavior that wouldn't be able

to exist in a public forum. I'm going to lean towards brutal honesty and transparency.
- [Jacques] For example, when discussing moving someone to the SMC, I would imagine that to be a conversation that happens in public, which is something that would not happen in Apache.
- [Carlo] Feels a bit risky. I agree with the spirit of the proposal. If we limit all private conversations though I think there are a few failure modes. First, people will still be nervous about having public conversation and so will have private conversations anyways (e.g. calling up friends to check for consensus before making a proposal)
- [Carlo] Second failure mode is that we are afraid to be honest or controversial in a public channel and we just open the doors too much out of fear of hurting feelings. Finally, there is a risk the other way, where this honesty leads to conflict and strife.
- [Carlo] I'd suggest that all technical discussion should be public (e.g. no private discussion of features). However, when it comes to people issues, I think private communication is inevitable and we might just want to formalize it.
- [Ashvin] Re: number of committers I just want to throw out that I think having more committers is better than having too few, especially early in the project.
- [Carlo] I agree that committership should be inclusive and shouldn't be too high a bar but I still worry that we might end up with the bar being too low if conversation is all open and people are afraid to speak up.
- [Jacques] I feel these conversations around promoting people to committership or SMC would be really valuable to have in the open. It gives people guidance, both direct feedback to individuals, but also helps demystify the "secret club" feeling where people aren't aware of what they need to do for success.
- [Jacques] That being said, I agree there is a risk that people water down their opinions out of fear of being seen as controversial or negative.
- [Ashish] We should also keep in mind that some people might not be in a position to send out regular PRs (e.g. interacting with Substrait out of personal interest and not being paid professionally to do so) and we should make sure that we aren't limiting committership to those with professional sponsorship.
- [Jacques] That is one of the reasons I support this open conversation. It could lead to people giving feedback on the committership criteria (e.g. these conversations seem biased towards those being paid to work on the project). I'm also hoping the specific awareness of corporate affiliation will help to avoid this kind of thing.
- [Carlo] I think another profile that will be rather common is when corporations have a heavy investment in Substrait / are making a big bet on Substrait within the corporation and they want to have someone involved in the project (with some rights) even though they might not be making regular substantial contributions.
- [Ashish] Just to reinforce with an example, I've worked with Apache and I've noticed that in some projects it can feel like someone works on a project for a

year+ and doesn't get much notice and then someone else comes in and within a month or two is recognized.
- [Jacques] Again, an advantage to having these conversations in public is to avoid this. How do others feel about this?
- [Jeroen] In spirit, it feels good, I don't have a lot of experience with other open source projects. I do think there will be some inevitable conversation on private channels (e.g. I chat on Slack about Substrait on a regular basis) but I'm not sure if that is on topic here.
- [Jacques] Actually, I think that is a good topic too, but maybe let's pause that for a second. However, I do think we should, as stewards of the project, try and force those conversations out into the open. This happens in Apache too. People will start a conversation on a private channel and, at some point, someone needs to point out that the conversation move to a public channel. It may never get to 100% public convo but if we can move from 50% to 25% for example that can be a big benefit. This has to be driven by community members though.
- [Richard] I do like the idea of everything being public but what if, for example, someone has anger issues? Or something like that? How would that get handled?
- [Jacques] It can be a problem that sometimes people act toxic and others are resistant to call them out. I've seen this in other projects, and I guess I don't know if public/private would have really addressed this example I'm thinking of.
- [Carlo] I do feel like having a private channel allows a ground for conversation about these people.
- [Jacques] What's wrong with the conversation happening in public?
- [Carlo] I just don't think it will be an honest conversation in public. People will be afraid to address the root problem and instead will come up with proxy concerns (e.g. not mention the personality but complain about the # of commits)
- [Jacques] If we are afraid to mention a personality issue then I think that we just have to pull the band-aid off and be direct about it. I think community requires a shared set of goals and behaviors. We can choose, for example, slower progress for a more positive culture.
- [Carlo] Will we scare off potential donors / members if they feel we are airing our dirty laundry or fighting all the time.
- [Weston] My hunch is that having these conversations in public will be for the best.
- [Marc] I don't have a lot of hard data for too strong of an opinion but I sympathize with what has been said.
- [Jacques] Another approach could be to very specifically limit what topics are allowed to be private and make sure that everyone does what they can to police this.
- [Jacques] I haven't heard a lot of criticism of being corporate aware. For example, if we were to look at the proposed list today, it does seem like we have a pretty strong presence from Voltron Data, so we should be a little wary of

monopoly there (disclaimer, Jacques is an advisor for Voltron Data so even he isn't really independent)
    - [Carlo] I think the very healthy behavior we want is that code is merged only based on quality and not based on importance to any single corporation. I think the broad proposal is valid and we can measure over-monopoly by watching for concerning merges / commits.
    - [Jacques] Yes. And hopefully this leads to things like investing in getting new companies and organizations involved.
- Options (optional enums)
    - [Jacques] Weston has a proposal to allow optional enums to be omitted. There is some discussion there between two approaches. Weston's original approach was a bit more gradual of a change but I think maybe more confusing in the long run.
    - [Weston] I agree and support the more stark approach.
    - [Jacques] Any other opinions?
    - [Jeroen] It's going to be a breaking change no matter what so we might as well go for the best long term endpoint.
- Oneof vs. enum
    - [Jacques] Jeroen has pointed out that we should maybe always prefer oneof over enum as it leaves more room for expansion. I'm not sure exactly how the serialization changes but this seems like a good standard moving forwards.
    - [Carlo] So I would use a oneof for something like a join type? I guess this makes the spec a little bigger but maybe not too bad.
    - [Jeroen] Actually makes the JSON a little bit more readable because you don't have the long prefixes.
    - [Jeroen] Another thing to consider is if we want to go through and do a cleanup to remove some of our deprecated fields at some point. Also, we might want to bunch these changes to increase the odds of producers and consumers speaking the same version.
    - [Jacques] Let's start a discussion on that

September 14th, 2022 8AM PDT

Agenda

- [Weston] Retrospective: What are the biggest areas of concern? Or pieces of functionality that are missing?
- [Chaojun] Substrait-cpp proposal discussion
    - Substrait extension YAML parser: parse core extension YAML files and establish substrate function variants like substrate-java did. (mainly focus on parse function variant and types system)
    - Function resolver/validator: Given a function name and function arguments , return a matching function variant against substrate extension yaml files. (Would be useful for substrate plan consumer)

        - Plan converter utility API: collect extension functions and types and add them to the substrate plan. (e.g. addExtensionsToPlan help to reduce mental burden for substrait plan converter)
- [Jacques] Function compatibility learnings (and general versions/compatibility).
- [Jacques] Java Spark Stuff
- [Carlo] Governance (I got questions internally)

Links

- Recording
- Transcript

Attendees

- Cheng Xu - Intel, AI and Analytic Group
- Zhang Chaojun - Intel, AI and Analytic Group
- Gil Forsyth - Voltron Data
- Vibhatha Abeykoon - Voltron Data
- Marc Hannah - Voltron Data
- Richard Tia - Voltron Data
- Weston Pace - Voltron Data
- Krystian Sakowski - Microsoft, Power BI Analysis Services
- Jesus Camacho Rodriguez - Microsoft, GSL
- Carlo Curino - Microsoft, GSL
- Kevin (Vamsi) - Amazon
- Jacques - Sundeck
- Jeroen van Straten - Voltron Data

Notes

- Discussion of write rel PR questions
    - [Carlo] I have to leave early and the Github discussion is getting long here so I was hoping to talk this through.
    - [Jeroen] I'm not sure I'm prepared for the discussion
    - [Jacques] Me either
    - [Carlo] Ok, let's schedule a dedicated meeting for the topic
    - [Jacques] I will schedule via a thread on the google groups
- Areas of concern
    - [Weston] We might be wrapping up some things in the next months and I'm wondering what people are concerned about?
    - [Carlo] Governance, not for me particularly, but people I was talking with at VLDB were having concerns.
        - [Jacques] Yes, this is on me. I was going to start a ML topic about this but didn't so I need to. I've noticed that, other than the Apache foundation, the other governance models don't have much structure.
        - [Jacques] First, let's do the ML and get the formal structure established, then find a foundation.
        - [Jacques] Biggest risk to investors should really just be the trademark issue. Investor's need confidence that the brand won't be "owned" by

some company. Do you know if investors are worried more about trademark or the lack of structure?
        - [Carlo] I think the primary concern is that people spend a bunch of time and energy on the project and then have no say or control in the project. I agree that nothing can really prevent community shift but if you loudly declare a foundation, gather people, etc. then the odds of turning evil are harder. For example, the chances that Iceberg will shift is pretty slim. The odds that Hadoop will shift is even less.
        - [Jacques] That makes sense and I think step one is getting the formal governance model agreed to. Meanwhile we can work on step two which is finding a foundation.
    - [Jacques] The first thing I'm worried about is that our compatibility model for functions sucks. We just made a bunch of (valid) changes to functions and they were all backwards incompatible. It's also really hard for people interacting with the project to know if they are interacting with the right version. For example, Java is probably producing functions that aren't compatible with any other projects.
        - [Jacques] It probably won't matter a lot in a few years when things stabilize. But, the churn rate at the moment could be intimidating to new contributors.
        - [Jacques] The third thing is finding the right balance orienting with SQL. SQL is the inspiration because it has wide use and adoption. E.g. the recent discussion on nullability. The discussion was fine but the structure of "why is this here if it isn't in SQL?" is something I worry about.
        - [Jeroen] I agree with a lot of these things. I agree we need to fix the function compatibility issue. For example, options could be resolved by name instead of index.
        - [Carlo] I noticed a similar debate on the write rel as I was coming from a SQL mindset. This is an inherent challenge for this project and what makes this project useful. Some of what we are doing is connecting the dataframes and the SQL worlds.
        - [Carlo] SQL is successful. We should be able to go back and forth with SQL on most things. This should not prevent us from having a natural model for other systems as well.
        - [Jeroen] Agree that taking representation with SQL is good and aiming for most SQL functionality is good but we just need to be careful to avoid the weird stuff from SQL. For example, how aggregate functions deal with nulls is very odd to me. I struggle with the balance as well.
        - [Carlo] If we have more functionality than SQL (and maybe a SQL mode flag) that seems fine. But if a SQL user has to jump through all kinds of hoops to get the expected behavior that is not ok.
        - [Jeroen] Yes, for example, we could solve one of these problems with lambdas and that would be clean but a lot of work for a SQL person so maybe not best. That being said, I do think we should consider support

for lambdas. This would very much simplify things like custom sort functions.
        - [Weston] Are lambdas just another form of UDF serialization? Probably off topic
        - [Jacques] This is also related to the idea of a transformations library. A lambda function could be a way to express these transformations. The main point being that there are other places I think would be improved with lambdas. Whether this is a unique thing or something in the UDF bucket doesn't really matter. It isn't clear to me how much this will solve problems we have.
    - [Carlo had to leave early but Teams froze his expression in a creepy smile for the rest of the meeting]
    - [Weston] I have some slight concern that we have a lot of options, which is good for specifying everything, but consumers are not likely to support all of them. In the rare case where a producer does care about a particular corner case behavior, they will find that their plan is quickly rejected by most/all consumers.
        - [Jeroen] True, but that's just how it works isn't it? If someone wants something and no one provides it what can you do?
        - [Jeroen] I do hope that consumers will do their best to fulfill the options and not just choose to ignore all options.
        - [Jacques] One thing we don't have is a model where a user can specify precedence. For example, "I really want behavior X, but if that can't be delivered, I will live with Y".
        - [Jeroen] That also helps because our "option precedence" is currently based on spec order and that makes it hard to add new options (e.g. if we moved to name resolution then order doesn't make as much sense)
        - [Cheng Xu] We've talked with some customers and they care deeply about function semantics. They want to keep existing behavior even if things change. For example, if a Spark user swaps out their engine, they want this to be transparent.
        - [Jaques] There will always be some challenges there. There will be some mismatches between those two. For example, some Spark capabilities may not be available on the new engine and you would have to still run those parts in Spark.
        - [Cheng Xu] So the consumer side maybe needs to generate some description of what it can offer.
        - [Jacques] I think that the API for a while is going to be "here is a plan, can you do it?" Maybe in the future we can specify further (for example, allow the consumer to explain exactly why it can't fulfill a plan).
        - [Cheng Xu] So in that way the producer will have to handle the fact that a consumer might just fail and be able to fall back gracefully.
        - [Jacques] I think so. For example, Spark has a ton of functionality. I think newer engines (e.g. Velox) will start off with smaller sets of capability (focusing on the most popular).

- [Jacques] Point of order, we haven't done introductions today and I notice we have a lot of new people.
- [Jacques] I noticed that someone put something on Github for Spark <-> Substrait. I was wondering if anyone had any knowledge of that.
- [Chaojun] We have a proposal for a substrait-cpp project (see agenda).
    - [Jeroen] That first bullet is something I have been working on in Rust for half a year now. I've gotten to the point where I have an ANTLR grammar which is fairly complete. It may require some changes (I will send a proposal). Since this is so complicated it might be nice to leave the YAML parsing to a single too and so, for the validator, I've been creating protobuf files that represent the YAML content. Maybe it would make sense that we automate YAML -> protobuf and that would simplify the first bullet quite a bit. Otherwise, this sounds good, using the validator for these things would be maybe a bit heavyweight so having a C++ utility could be nice.
    - [Chaojun] I have some questions about the validator. One of the parts of my proposal has some validation but it is just to validate a part of the plan. Yaml parsing sounds good if we could do just a part.
    - [Jeroen] You can't just do Yaml or partial validation now but I had planned on making improvements on these areas in the future and adding more export capability.
    - [Jacques] I agree we probably don't want to build the same thing multiple times. Should the validator be broken into a set of tools / utilities that other projects could use? It might make these things more consumable.
    - [Jeroen] One challenge in the validator is that each of these utilities have to go above and beyond because, for example, a validator can't just throw an error, it needs to explain why. So this would mean that generic utilities would have extra functionality that doesn't really belong in the general utility. I think I'd prefer to have the validator support more inputs / outputs / behaviors. That's why I agree that this generic cpp library would be useful for these utilities.
    - [Jacques] I do agree that there are many common utilities and a common format that are useful. I know the Acero project has it's own internal representation of IR but it would be nice for them to keep an eye on this too.
    - [Weston] Agreed. Even though Acero has it's own internal representation we still have users and they will want help manipulating Substrait plans and so it makes a lot of sense for us to help out here.
    - [Jacques] I know that protobuf is complicated when it comes to object sharing. Do we think this is going to create problems for this kind of library?
    - [Jeroen] I think it will be possible, or more accurately, I think it will be equally problematic regardless of language. Mainly I would just advise against using Google's protobuf.
    - [Jacques] To summarize, this will be useful, but we need to take care to make sure this is usable (e.g. usable in Arrow).
    - [Jeroen] Yes, though it may be very difficult to detect when these problems occur. I don't think it's worth even trying to get google's protobuf lib to work because you

might think you are successful and get quite far before you run into these problems.

August 31st, 2022 8 AM PDT

Agenda

- [Weston] Governance update request
    - Main repo: Lot's of discussion around function specs, how to keep it moving?
    - Testing repo: We have some cooperation between Voltron (Richard) and DuckDb (Pedro), is this enough to get some initial work done here?

Links

- Recording
- Transcript

Attendees

- Jeroen
- Ashvin Argawal
- Vibhatha Abeykoon
- Richard Tia
- Nic Crane
- RJ
- Gil
- Krystian Sakowski

Notes

- Re: Governance Update
    - [Weston] Can we get a governance update. Jacques is in the car, so maybe via mailing list
    - [Jacques] Yes, I can send something on the mailing list.
    - [Weston] Also, on a related note, it might be useful if we had more people with write privileges, even if they can be vetoed.
    - [Jacques] There are a number of people with write privileges aren't there
    - [Jeroen] The main branch is protected, so admin privileges are required
    - [Jacques] That is an oversight, I will address that today
    - [Weston] Great, thanks. On a semi-related note, we also will wnat to figure out who can own/merge/move forward the testing repo. We have a few people working there now, Richard, Gil, Pedro, though it likely won't get as much attention as the spec repo which requires more cooperation.
    - [Jacques]: That sounds like a good thing to follow up on the mailing list
- Re: Questions / PR on writes, deletes, references (#218)
    - [Jeroen]: I have a PR that I would like to get some more attention on
    - [Ashvin]: I'll work with Jesus to get some people to take a look at it
    - [Jacques]: Also, the PR has several things in it, most of which is straightforward but some might be more controversial.
    - [Jeroen]: Thanks
    - [Jacques]: That merge was important to Microsoft, so it might have moved forward to get things moving and we can clean up after

    - [Ashvin]: On that note, there is increasing excitement / attention at Microsoft, moving forward builds confidence.
- Re: Functions
    - [Jacques]: I have to imagine there is some structure / order in all of the functions work that is going on. Is there some kind of underlying list? If might be helpful to expose this list as an issue so that others have some context.
    - [Weston] There is a list, Ian Cook has been maintaining it, we can share this. I'm not sure of the motivation beyond Ian's brain.
    - [Nic] Agreed, there is quite a mix of things, I can contact Ian.
- Re: Common transformations
    - [Jacques] At one point, Intel (not sure if anyone from there is on the call) was discussing a library to provide common transformations. For example, providing polyfill behavior (between <=> greater & less) or flattening / transforming subqueries. I'm hoping to get some contribution here and am curious if others see a need for this and any perspective on approaches.
    - [Jeroen] I haven't been thinking in detail but I have been assuming something like this will be there eventually. One thing I have been thinking about is parsing YAML files and type derivations. I need to do this in the validator which has to do a bit more than normal here (e.g. to provide error recovery info).
    - [Jeroen] One question is what language we want to do this in. We probably don't want to reimplement this 20 times in 20 languages. I'm worried that Java may be used and isn't quite the easiest to interface with.
    - [Jeroen] Another problem is that this program will need to be able to ask consumers what they can or cannot do and we don't have anything here yet.
    - [Jacques] That is probably the end goal. However, it seems like this is something that could be done in stages. Initially you could choose to use a particular transformation. So one could apply a between transformation explicitly. Then, later, we can worry about automatic determination of what transformations to apply.
    - [Jacques] Regarding languages, I agree that we need a C API for this stuff and a library that doesn't cause problems with namespaces. It's pretty much the same set of requirements we had for the Rust validator. This is possible in Java (with graal) but I'm not saying that is necessarily the right place for it. Historically been thinking that Rust is the right place but I don't have a strong opinion.
    - [Jacques] The cockroach-db was originally Apache licensed, in go, before it switched to a proprietary license. It has a DSL for transformations, called optgen (sp?). This is an interesting approach and I think there are lessons to be learned here. We have been experimenting some with an older version but haven't yet arrived at anything we can share. I'll send an email out linking it and suggest people take a look.
    - [Phillip] What level of complexity did these transformations get to?
    - [Jacques] It's pretty flexible. Some of the trickier logic was put into functions and not in the DSL itself.
    - [Phillip] Like an escape hatch for complexity?

- [Jacques] Yes. We've been toying with it and haven't really taken it too far but have gotten some good study of it.
- [Ashvin] This also overlaps a bit with the tiering concept we discussed at one point regarding grouping consumer capabilities into "tiers".
- [Jacques] Yes, for example, you could use transformations to bring a consumer / producer up to a higher tier with potentially lossy transformations
- [Phillip] Very interested from an Ibis perspective as we do some of these things ourselves
- [Jacques] My observations from Calcite is that most of the value comes from the transformations that are prebuilt. It's very unfortunate that we have to keep implementing these over and over again.
- [Jacques] Does anyone see any short/medium/long term need for this? I think a need for this will get it moving fastest.
- [Weston] In Acero there is definitely a need for medium/long term. For example, subquery flattening and join order rewriting and other pretty basic transformations. We have avoided doing any of these things in Acero today.
- [Jeroen] In addition, we would probably want to use some of these simpler function rewrites to address better mapping of functions and filling in some of the gaps.
- [Jacques] I suspect, if we don't approach it generically, then it will end up getting implemented in several places piece by piece since it tends to be a very slippery slope. The rewrites start simple but they can get complex quickly and so engines might find themselves bogged down. So sooner rather than later we might want to address thi.
- [Phillip] Are we thinking of encoding some of the simpler optimizations like projection pruning & collapsing?
- [Jacques] That seems like it is the same thing.
- [Phillip] We've been prototyping some of this in Ibis using a library called match-py (sp?) so now I'll think about this regarding Substrait
- [Weston] So what is the difference between this transformation library and something like Calcite?
- [Jacques] I think there are two things here. One is a transformations library. The other is cost-based. For example, in Calcite, there is a heuristic planner which will just run transformations until I can't run any more. There is a more cost based approach which will only run transformations until the improvments are not substantial. So a planner uses a transformations library to achieve it's goal, which is to optimize the cost.
- [Jacques] That being said, we do have something of a transformations library in Calcite. So it might be pretty close to being able to provide these transformation capabilities, we would just need to hook up the appropriate APIs. For example, it could be used as a decorrelator. Calcite is also a SQL parser, a reference engine for execution, a transformation framework, and then optimizer modules which optimize in different ways.

    - [Jacques] Given that this is very close to being available in Calcite, would there be interest in exposing this capability? We don't need it for SunDeck but if someone could use it we are very close to exposing the capability and it would be a shame if we didn't do that. The things that are missing (which shouldn't be too much work) are:
        - Finishing some of the round tripping
        - Packaging as a C library (we already compile with graalvm)
    - [Jacques]: Doing the above would make a C API available for ~100-150 transformations with 1-2 weeks of work. Would anyone use this? Would anyone want to collaborate.
    - [Weston]: We have some people that are more knowledgeable re: Calcite, I can ask them.
    - [Jacques]: Some of the things you mentioned (e.g. TPCH subquery flatting) would be available to you.
    - Some discussion of GraalVM
    - [Weston] I think one question would be whether this is a sensible stop on the path to a long term goal which is a planner for Acero. For example, would a user be domain-aware enough to know what transformations to apply.
    - [Jacques] Fair, and I don't know the answer, but there is probably some set of transformations that you would want to just always apply in a heuristic fashion.
    - [Jacques] Also, as I mentioned earlier, there is a C++ library that is being worked on to do this, that will hopefully be open sourced in the future.
    - [Ashvin/Weston]: This sounds very interesting, we will follow up.

August 17th, 2022 8AM PDT

Agenda

- Feel free to add items

Links

- Recording
- Transcript

Attendees

- TBD

Notes

- TBD

August 3rd, 2022 8AM PDT

Agenda

- [Weston] I'd like to revisit URIs again (we can hopefully be brief). There are a number of tools trying to coordinate now. I'm just trying to look for one option for going forward
    - Acero is looking for https://github.com/substrait-io/substrait/blob/main/extensions/functions_arithmetic.yaml

    - Isthmus is generating /functions_arithmetic.yaml
    - Substrait validator accepts ?
    - Consumer testing appears to be using Isthmus plans
    - Ibis is generating ?

Links

- https://github.com/substrait-io/substrait/issues/274
    - Issue discussing function versioning scheme.
- Recording
- Transcript

Attendees

- TBD

Notes

- [Nic]: Gave brief intro of dplyr / R integration and goals on request
- Re URIs
    - [Weston]: Originally concerned about integrating Acero with various tools that don't agree on what the URI should be at the moment. Ended up deciding to treat a URI of / or <EMPTY> as a wildcard and falling back to name-only matching.
    - [Jacques]: Some concern about using github URIs as they are not guaranteed to be stable, or at least, we don't "fully own" the URI.
    - [Carlo]: We could maybe just check with Github and see if they think that URI might ever change.
    - [Jacques]: I don't think this will be super controversial. Perhaps lets just pick something. Weston, do you want to just create a PR.
    - [Jacques]: Formalizing the versioning scheme might also be a good idea.
    - [Weston]: See links section
- Relations with multiple / shared outputs
    - [Jacques]: Problem statement: there is currently no formal way of defining a "tee" or a node with multiple outputs.
    - [Jacques]: My plan was originally to use multiple top-level plans. The shared part would be a top-level plan. The dependent parts would then reference the other plan. However, this "reference" capability hasn't been defined yet. Does this work?
    - [Carlo]: So to represent an arbitrary DAG we encode it as multiple trees that reference each other.
    - [Jacques]: Yes. A single "Plan" message can have multiple "PlanRel" objects. Then we would create a new operator that can refer to another relation. This would be a reference to a PlanRel.
    - [Jacques]: Currently everyone just has one root rel but this implementation would rely on people having one or more non-root rels.
    - [Carlo]: We have both (dag and collection of merged trees) at Microsoft. The referenced trees seems like it might be more flexible / powerful since this idea of references might be similar to a mechanism used to refer to a named view. I can imagine people coming from a spark dag world might struggle more.

    - [Jacques]: I don't think it will be all that burdensome. This also fits patterns where you can create a reference even if you don't have to. For example, if you have a CTE, even if it isn't shared, this approach might be clearer.
    - [Carlo]: No concern, I've found that Substrait always ends up being correct, but often the opposite of my intuition. My only worry is people thinking DAGs are not supported because they don't intuitively understand. Maybe there is some syntactic sugar we can do in a viewer.
    - [Jacques]: One related question is "in two years, how much are people going to be working with the protobuf, versus working with the tools?" For example, when doing Arrow, we found that the Java implementation could paste over spec oddities to make things more natural for the Java language. Now people's first interaction tends to be with the libraries and not with the format. So I'm not as worried about novices encountering Substrait in the future, but this is hard to predict.
    - [Jacques]: As another example, when we worked on Isthmus, we realized it would be very useful to have a representation of Substrait that was tree based, has a proper tree.
    - [Various]: Some discussion on different projects and how they use an intermediate representation as well.
    - [Jacques]: I think we need some more documentation / definition, at least on the reference node.
- DDL PR needs love
    - [Jacques]: Still don't understand why the write rel has an output
    - [Carlo]: The write rel prepares the data and there could be multiple physical points where it is written. The write rel could then be shared across multiple outputs.
    - [Jacques]: So if you want to write to two different places you would have a single write rel plan and then an arbitrary number of other rels which take this write and persist the output.
    - [Carlo]: The input rel tells me what tuples to operate on. E.g. for delete, the input is telling us what 20 tuples we want to delete. The output in this case could be the number of tuples affected by this operation. In other cases, people want to have the deleted tuples returned so that they can do some kind of assert, filter, etc. In SQL server you can even do something like "in a query that updates salaries compute the total difference between the 'before salaries' and the 'after salaries'.
    - [Jacques]: My inclination would be to decompose this. So the simplest way to do this would be to say that the write rel has the ability to pass through tuples and you can stack whatever operations you want to work on this.
    - [Carlos]: This works for certain deletes. I don't think it works for the updates. Some systems want to be able to support both the before and after image of the tuples updated.
    - [Jacques]: Maybe we have passthrough and before/after as two different modes. The before/after view could, for example, return a struct with before and after

keys. This seems preferred to having subtrees within the operator because then you would need to come up with a way to reference the input.
    - [Carlo]: Let's pretend DELETE FROM foo WHERE X > 10. This would have a subplan SELECT FROM foo WHERE X > 10. An afterimage mode seems odd.
    - [Jacques]: I had not understood that your decision was for this to be an arbitrary tree with no references.
    - [Carlo/Jacques]: Some discussion on this PR
    - [Jacques]: Would it be sufficient to have the output be # of tuples modified, before data, and after data.
    - [Carlo]: Perhaps a simpler version is that the write rel always the after-image of all affected tuples. If you don't want to see them then don't look at them. If you want the count then count them. If you want the before image…
    - [Jacques]: I think 99% of the time people want the count of changed items so maybe just add a special output for this
    - [Carlo]: True, but it would be really simple to just COUNT(*) the output.
    - [Jacques]: And use an emit clause to exclude all the columns. Still, it seems useful to have some kind of flag to just truncate the results.
    - [Carlo]: And, if you want the before image, you can create a shared rel and merge with the after image.
- Prepared statements
    - [Jacques]: There is an outstanding patch about variables that are bound and then used later. I'd like more input on this. Right now it feels like it might be solving too many problems / be too generic and it would be helpful to constrain.
    - [Jacques]: However, one of the things that seems very worthwhile is an "expression annotation node" (and maybe the same thing for relation). It's a node we define which allows you to add arbitrary description information about the node. So the output of this node is identical to the input (it is a passthrough) and it is just meant for annotations.
    - [Jacques]: Could be used for labeling errors in a parsed plan, for informing people looking at the plan in a viewer. Could label names to help understand what parts of the plan correspond to what "labels" in a prepared statement.
    - [Jacques]: So, this could be useful, but we could also just add a notes field to arbitrary expression nodes. My concern with prepared statement was that this was going to end up specific to prepared statements and it should be more general.
    - [Calro]: So this is a way to label the "?"
    - [Jacques]: In this case named bindings was very popular so it was "foo" and not "?" but yes.
    - [Carlo]: An arbitrary annotation operator seems useful. Are we saying this annotation needs to be processed to understand the prepared statement.
    - [Jacques]: No, this is supplementary.
    - [Carlos]: Then, I'm fine with it. We just want to make sure these annotations are not used in the computation of the result at all. Agree it is fine and should probably be generic.

    - [Jacques]: We could maybe just put it in advanced extension (this is available in more places than rel common).
    - [Carlos]: I don't know how I feel about advanced extension, but it very well might be the right answer. By the way, I'd also like to be able to attach these annotations to the plan itself.
    - [Jacques]: Some further explanation of advanced extension and how it works.

July 20th, 2022 8AM PDT

Agenda

- See notes

Links

- Recording
- Transcript

Attendees

- Carlo C.
- Ashvin A.
- Richard T.
- Nic C.
- James T.
- Jacques N.
- Weston P.
- Jeroen V.
- Vibhatha A.

Notes

- Re: table creation
    - [Jacques]: Need to strike a balance. Don't want Substrait to be too closely bound to SQL but also don't want to make things difficult for SQL engines to adapt either.
    - [Carlo]: So ok to create things from SQL, just need to make sure they aren't reliant on SQL. For example create table / create dataset.
    - [Jacques]:There are definitions with data (create table as), definitions with trees (create view), definitions without either (create table / drop table)
    - [Carlo]: Create table as and create view are very similar though. Only difference is persistence
    - [Jacques]: Create view has a subtree that is treated as a definition. Substrait would assume that subtree is executed. Is that subtree a "property" or "input"? I see it as a "property" since it isn't really "data". Create table as on the other hand has a table which is "input".
    - [Carlo]: Still, the syntax is pretty much identical, I'm not sure the difference is significant
    - [Jacques]: I won't fight too hard on the issue, but I still wonder whether it might be important at some point.

    - [Carlo]: Is this just about having two different Rel properties on the operator? One that is “input” and one that is “description” and which one is used depends on the operation type?
    - [Jacques]: Yes
    - [Carlo]: That’s fine, but it seems redundant. You’re free to make whatever nuanced distinction you want based on the operation type so why be verbose / potentially ambiguous?
    - [Jacques]: Let’s table this for a moment. I still think the distinction between “create table” (which has no tree at all) is significant
- Re grouping
    - [Jacques]: Not sure if anyone has been following the grouping functions PR closely. Also Jeroen recently pointed out that the spec states a grouping aggregation includes the group identifier as output (if there is more than one grouping set)
    - [Jacques]: I’m leaning towards just not including these grouping functions. They don’t feel quite the same thing. As a result I’m not really giving this full attention. Do we want to make a final decision?
    - [Jeroen]: Disclaimer, I don’t have a lot of background here, but I agree that it doesn’t feel like a normal function because it doesn’t look at the input. I’m not really opposed to making it a function, but it also seems redundant given the output of the aggregate operator.
    - [Jacques]: Agree that the operations we have might be sufficient. The grouping functions came from a desire to express TPC-DS. Some of those queries use grouping functions so I think we just need to show these queries can be expressed using the grouping set column added by the aggregate operator.
    - [Jeroen]: FYI, you said the grouping set column is only present if there is more than one grouping set. My interpretation was that the grouping set column was always present. We should clarify.
    - [Jacques]: My intention at least, was to avoid surprising people that don’t even know this feature exists or how it works, thought it is admittedly a slightly more complex model. Agree we should clarify.
- Re function kernels
    - [Jacques] Another side note, notice that Jeroen created a PR about kernels which describes my feelings pretty good on when something deserves to be a kernel (e.g. implicit casting vs explicit casting). Clearly Jeroen and I are in agreement but several recent function PRs seem to have some question on this so it would be good to make sure we are aligned.
    - [Weston/Jeroen]: Some discussion about Arrow’s implicit casting and why this might be the case
    - [Jacques]: Also, Arrow is probably going to have its own YAMLs and functions that are Arrow-specific and that is fine. Also, just an observation, Arrow has been working at the logical level vs a physical level.
- Regarding intervals

    - [Weston]: Question, Arrow's intervals support a wider range than that specified by Substrait. How does a consumer express whether this is a problem or not? For example, an interval multiplication kernel could yield intervals outside Substrait's range.
    - [Jacques]: I'd probably just document it and not worry too much about it.
    - [Jeroen]: I think the way the spec reads right now the underlying types are allowed to exceed the range as long as they respect the kernel / overflow rules. Since there is no overflow option for interval multiplication then an overflow is undefined.
    - [Jacques]: I'm supportive of "you can use larger ranges as long as you respect the rules of the functions"
- Regarding overflow
    - [Jeroen]: Silent means 2's complement overflow right?
    - [Jacques]: My interpretation is that silent means "I don't care, do whatever you want" and the consumer shouldn't rely on the value being meaningful in any way.
- Regarding function descriptions
    - [Jacques/Jeroen]: There seems to be a lot of potential for duplication in function & function argument descriptions.
    - [Jacques]: We did have something that didn't have so much duplication in kernel specification originally. At the time we decided that YAMLs weren't going to grow all that fast and being a bit simpler and more verbose is fine.
    - [Weston]: I think Jeroen was specifically talking about documenting parameter descriptions, which could get rather repetitive with lots of kernels. Overall it seems like we are leaning towards under-documenting functions.
    - [Jacques]: YAML does have anchors/references that we can use to re-use snippets. But it isn't terribly readable and there isn't great support in most YAML packages. There is a balance here but I think repetitive copy/paste is still the way to go.
    - [Jeroen]: That matches my gut feeling as well.
    - [Jacques]: We could also maybe build some scripts to make it easier to add names/descriptions, etc.
- Re Carlo Update
    - [Carlo]: We have a system which is translating SQL queries into tensor computations which we then send to pytorch. We are trying to introduce Substrait here. And we are doing the same in a few other systems but not ready to talk about yet.
    - [Carlo]: I went to sigmod a few weeks ago and talked with a bunch of people about Substrait and people were very excited about this. This isn't very scalable to talk to one academic at a time. It would be good to have a paper where we describe a bunch of ideas. For this to work we often need a bunch of experiments and systems. So maybe it is too early but still the sooner we put something out there the sooner we get a bunch of students playing around with this and we can get great academic reach / flywheel effect.

    - [Carlo]: Is anyone interested? Is anyone at the place where they can run TPC-H queries?
    - [Weston]: We're close, maybe a month or two out, with some caveats that we have manually decorrelated subqueries in some cases
    - [Carlo]: Sounds good, we should also start thinking about what things we want to measure. We'd want to show the benefits of Substrait and not just focus on engines.

July 6th, 2022 8AM PDT

Agenda

- [Ian] In the next few weeks, a group of engineers at Voltron Data are planning to define more functions in substrait-io/substrait/extensions
    - This work will probably raise more questions like the ones here: https://github.com/substrait-io/substrait/pull/230#issuecomment-1170064164
    - Are there any up-front concerns these engineers should consider?
- [] TBD

Links

- Recording
- Transcript

Attendees

- TBD

Notes

- TBD

June 22nd, 2022 8AM PDT

Agenda

- [Weston] Recommendations for URIs? What form? Should they be versioned? Should they host actual content?
- [Weston] Integration tests, getting started

Links

- Recording
- Transcript

Attendees

- Ashvin Agrawal
- Guy Khazma
- Jacques Nadeau
- Jeroen van Straten
- Jesus Camacho Rodriguez
- Weston Pace

Notes

- Regarding URIs

- Weston: When do we want to start narrowing down what URIs we use? Do we want to use Github URIs or something simpler? Those should point to the YAML file right? Do we want to start working on a way to solve the YAML file composition problem (e.g. how do files refer to extension types in other YAML files).
- Weston: First, the URI question. People have been using the Github link (to the YAML) for URIs. Is that what we want to keep using going forwards? Is there a significant cost to change this later?
- Jacques: Lately people have been using URIs that point to the current master which seems brittle (YAML could change without URI change). We could use tags but that has a different problem (URI could change without YAML change)
- Jacques: Do we have a concrete situation that we are struggling with right now? Best to solve this problem one real need at a time. You (Weston) asked if we're ready to talk about this and I suppose the answer is, "is there a real need?"
- Weston: I can think of two needs, one more concrete than the other. First, Sanjiban recently created a PR adding a bunch of new functions. What happens if we get one of those wrong? Do we just create a new function (e.g. add2, add3, …) or do we try and maintain forward/backward compatibility. However, this use case is not terribly concrete.
- Weston: Second, and a bit more concrete. Arrow has a function registry internally. Some of these are not in the Substrait spec. A YAML file for these functions will be generated dynamically. My current thinking is the consumer will look at the URI, and if it matches the "dynamic Arrow function" URI, then instead of mapping manually to a specific registry we will automatically interpret the call. This doesn't really map to a YAML file so I wasn't sure what the URI would be in this case. We could publish a YAML file for this on a semi-regular basis manually and use that as the URI but what URI? How do you express you want to use a function defined since the last release?
- Jacques: I think what we want, at an abstract level, is versioned functions, and not versioned YAMLs. Adding a new function shouldn't force a version change because the other functions didn't change. I'm not sure what the solution for this would be. Any ideas? We could have "open" and "sealed" YAMLs.
- Ashvin: What happens if functions depend on other functions?
- Jacques: That shouldn't be possible though functions could depend on types.
- Jesus: We should probably identify which changes are "breaking" changes and see if we can automate the versioning somehow. It's too much of a burden on the user to figure that out themselves.
- Weston: Agree. In protobuf it's possible to know what all the possible changes are and which changes should be breaking changes. For a recent example, we noticed that "divide" was missing overflow behavior, and adding this optional argument should be backwards compatible.
- Weston: I'll take, as a homework assignment, walking through which changes would be breaking changes.

- Jacques: I agree with Jesus in theory but I think most changes are breaking. Perhaps we should be putting a version number in every function. Also, that “divide” change wasn’t actually backwards compatible because we optional arguments come first so this inserted a new argument at the beginning which would break backwards compatibility. This is because optional arguments must always be specified (they can just be set to “I don’t care”). Although, the current Java isn’t necessarily doing this correctly (it sometimes omits optional arguments).
- Jacques: We could put optional arguments at the end. Or maybe we could even have two lists, one for optional and one for normal arguments. We probably need to think this through a bit more if we want backwards compatibility.
- Weston: True, we could also make named/numbered arguments which is how protobuf handles this for fields.
- Jacques: Circling back, if there aren’t that many backwards compatible changes possible, function versioning will probably work. Not sure what that means for URIs though.
- Jeroen: What do we use the YAML files for again? Most people either ignore the URI completely or treat it as a namespace identifier and never resolve the URI. In what context are these file actually interesting?
- Jacques: I think right now most consumers and producers are collaborators and not strangers. In the future that might change and the YAML may be more key. Also, interactive cases like an IDE will want to read the YAML. Also, maybe with consumer-registered UDFs where the consumer knows about functions that a producer has no idea of.
- Jeroen: But most of that should be accomplished with just a namespace and name matching. Why is the YAML important?
- Jeroen: Ignoring YAML for the moment, for versioning, another way is to just list all versions of all functions in the YAML file so that the latest YAML file is always the correct YAML file (e.g. this is how package managers work)
- Jeroen: Coming back to the YAML files, I worry they are too complex to parse and understand correctly.
- Jacques: The YAML files are probably underdocumented. The goal was simplicity so something is probably missing. The expression language is hard but a necessary evil because we need to be able to express type derivations.
- Jeroen: It’s not just a lack of documentation. The spec is missing and ambiguous in places.
- Jacques: So lets fix that problem.
- Jeroen: Ok, I’ll work on it. Could the producer just explicitly declare the output type?
- Jacques: What about a UDF? How would the producer know the UDF type?
- Jeroen: That sounds like producer-specific problem.
- Jacques: Then you just have a side-channel. We’re getting off topic, let’s go back to the versioning topic. It doesn’t sound like there is a strong consensus. Weston is the most in-the-thick-of-it so why don’t you try tackling this.

    - Weston: I'll document a list of fictional yet concrete cases for versioning and post an issue.
- Integration testing
    - Weston: I've maybe got some resources spinning up to look at integration testing. Specifically, creating tests to ensure that different consumers interpret the same plan in the same way. Should I just create a repo and add in test data, test queries, and test results?
    - Jacques: Rather than checking in data it would probably be better to check in tools that synthesize the data. SCM for data is difficult. For example, TPC-H.
    - Jacques: For results you might also consider using an Oracle such as postgres. If we can avoid checking in results that would be a big win.
    - Jacques: For results, the challenge is that a lot of properties can be optional (e.g. ordering) and so there could be many different valid results. Also, tolerance is another concern. I'll try and dig up some of our past knowledge to see if I can share.
    - Jacques: Making a repository still makes sense.
    - Weston: Thanks, probably still a few months out, but if someone else is interested it would also be nice to have a second system to be the integration test partner.
    - Jacques: Thinking about the producer side, I think that will be challenging as well because there are many right answers. For example, different producers could generate different plans for TPC-H. We can maybe share some "valid" plans for a query (as in, here is a reference query that is valid) but not sure how useful.
    - Weston: I think it also comes down to capabilities discovery. For example, given a query, and a set of capabilities, does the producer come up with a query that is valid and respects the capabilities of the consumer.
    - Jacques: Calcite does have an inefficient reference implementation of an execution engine. You could possibly use that as a second implementation for integration testing. It is a fairly rich engine even if it isn't very performant.
    - Ashvin: Just wanted to add that having some integration tests would be very valuable for attracting new people.
- Optional arguments:
    - Jacques: Circled back to the idea that optional arguments are often just plain omitted by producers and that will probably be a problem for consumers which are presumably expecting them to always be specified.
    - Jeroen: Isn't that the way the spec is defined? Shouldn't the consumers be more flexible?
    - Jacques: I think the spec is probably wrong here. We should keep the plans as simple to understand as possible. I think the plan was always that all arguments are always specified. Otherwise it is too hard to understand which argument is which. Jeroen, can you take the action to clarify the spec?
    - Jeroen: Sure, I'll make a PR.

June 8, 2022 8AM PDT

Agenda

- [Weston] I feel like there have been a number of “our docs should be better here” comments in the last few weeks. Is there a particular area of the docs we can have a sort of “mini-sprint” on?
- [JN] Couple updates from Java, Governance, etc.
- [Carlo channeling Subru] Protobuf layers of code
- [Carlo] INSERT/UPDATES/CTAS/VIEWS

Attendees

Ashvin A
Carlo Aldo Curino
Jacques Nadeau
James Taylor
Jeroen Van Straten
Sanjiban Sengupta
Weston Pace

Notes

- Regarding docs
    - Carlo: Landing page is very friendly, at a high level of abstraction “Page 2” suddenly gets very technical
    - Jeroen: Would be nice if website could be generated from protobuf, less likelihood of conflicts too
    - Jacques: There is a lot of content that doesn’t belong in protobuf. Some stuff could be done entirely in protobuf. But not all stuff. Also, might not want to go too far into protobuf if we are going to add a human readable format later. E.g. some things might be unique to protobuf and should not be part of the spec. If we constrain ourselves to the vocabulary of protobuf we are constraining ourself.
    - Jeroen: Protobuf-specific stuff does belong only in protobuf. The other stuff is protobuf-oriented. Users need to know what the protobuf messages are. Precision is lacking from the website. Documentation for type expression grammar is difficult (essentially by example)
    - Carlo: There are few places where we are out of sync (something in spec but not in protobuf). Also, if protobuf becomes constraining, then we need to tackle those constraints regardless. Also, makes it much clearer which parts of the spec have been implemented / not yet implemented.
    - Jacques: There are two levels of abstraction, logical concepts and physical representation. Original idea was to align on concepts (the spec is the concepts) and then move onto representation. Although, we aren’t seeing this translation of spec to protobuf happening when gaps are encountered (e.g. people just give up

if it isn't in the protobuf instead of inventing the protobuf needed). Agree there are some gaps / lack of precision, but that's partly just immaturity on the project too. Haven't even had anyone run TPC-H / TPC-DS fully yet. Would like more real world use cases.
    - Carlo: The pattern of spec before protobuf is important now. In the future, the protobuf is likely to be more important. Planning on more intense use of Substrait, trying to introduce it between components internally. Likely to start bringing up new people, and they are going to be expecting more polish on the protobuf.
    - Jacques: Let's get more tactical. We aren't conceptually far off. Proposals to improve documentation aren't being rejected. Let's just make some PRs and we're probably more aligned than we expect. Anything that improves the content is a win right now. Let's just focus on comprehension. Tools, example, content that helps understanding is welcome and lets just do it.
- Governance
    - Carlo: I feel like a guest at the moment. People might be a bit timid because the current governance model doesn't have a lot of formal "contributor/committer" roles and if people have such a role they might be more bold.
    - Jacques: I'd like to propose / put forward an Apache model. Conceptually I feel like we are kind of doing this already. People that are starting a new project are committers. Core repository has a slightly higher bar at the moment.
        - E.g. right now: Phillip / Jacques are the PMC. There are a number of committers (e.g. rust, c#, etc.) and they are focused on their areas.
    - Jacques: Minor clarification, people starting new projects need to have some history of open source elsewhere or have some history on the Substrait project. So there is a bit of background required.
    - Carlo: For existing repository will we add committership as people show investment (commits, etc.)
    - Jacques: Yes.
    - Jacques: There is a bit of weirdness that it is easier to become a committer on a new project than an existing one and that is just kind of how it works. But we should generally find some good level of initial investment to show interest in committership. Also, slowly grow the group of people in the PMC as people show interest / involvement above and beyond "the stuff I need for my current project" and more "interested in success of the project as a whole".
    - Carlo: I do like that each repository can have its own subcommunity. There is some risk that each subcommunity (or some particular subcommunity) is rather small and if they walk away there is a gap.
    - Jacques: There is some trust concern possible as people come in on a new niche project and start a repo and become a committer. At some point though, the project will mature, and the bar for starting repo will be higher. Instead you can create your own repository and get established first.
    - Carlo: Does it really matter if you're working on a new project or an established project? Let's just say "X amount of work / investment means committer".

    - Jacques: Sounds good, but just to be clear, “X amount of work” isn’t really concretely defined.
    - Jacques: I’ll write up a proposal for the mailing list. Also, can someone find the Apache documentation on this?
- Java update
    - There are 4-5 people working on the Java repo. Trying to get to the point where TPC-H and TPC-DS can go from SQL to Substrait. For example, one of the problems we are running into now is issues with enumeration arguments (eg. extract date) and so we are running into and tackling these issues as we go.
    - Jacques: Trying to round trip SQL / Calcite / Substrait. Goal is to get a wider selection of plans / etc. and build up the example corpus. Getting a lot of requests from Microsoft for plans
    - Carlo: There is a team that is converting relational algebra to tensorflow. They’re looking into using Substrait and that’s probably where the requests are coming from. Calcite integration will also help as we use that elsewhere.
- Insert / Update / Create view
    - Carlo: Are insert, update, delete, create view a part of Substrait?
    - Jacques: 80% yes. A lot of these can be a pretty simple plan but there are more complex variations too. Let’s just try proposing something.
- Protobuf & layers of code
    - Carlo: Protobuf gives you Java/etc. objects natively. But protobuf is very much against hierarchies. Might not be the most useful code representation. Might be interested in tooling to help bridge this gap.
    - Jacques: Specific to Kotlin, not sure if there is enough flexibility. Might be some challenges as a consequence of the Java language.
    - Jacques: For reference, the Java bindings have two layers. One has a richer abstraction and more methods around traversal, etc. However, this is all tightly coupled to the protobuf and a pain to maintain. Kotlin may have been the answer to avoid some of this maintenance pain.

May 25, 2022 8AM PDT

Agenda

- Is there consensus on versioning / backwards compatibility?
- What is the current thinking on capabilities discovery?
    - [jn] I'd like to start by defining tiers or levels of compatibility. Then use that as springboard/basis for defining more advanced permutations.
- How should we deal with reviewing the behemoth that is the validator PR?
- ~~Disappearing sync meeting~~
- Governance update
- Materialized Views
- Generic Rewrite Engine

Attendees

Carlo Aldo Curino
Jacques Nadeau
Vibhatha Abeykoon
Jeroen Van Straten
Ashvin A
James Taylor
Jesus Camacho Rodriguez

Notes

- Is there consensus on backwards compatibility?
    - Jacques: We are in a place where we can make mistakes and move on from them. Definitely good for anyone to get involved in the mailing list.
    - Jeroen: Some question on what unspecified means and if we can even have an “unspecified” oneof. General consensus on backwards compatibility
    - Carlo: Auto-release is very light on developers, good when we have few users. At some point, there is something social about the number changing too much. Don’t know how to settle without being too heavyweight. Coming from Hadoop which favors the user but releases end up being very heavyweight.
    - Jacques: Definitely want to get to a point where there are way more users than builders. There are dynamics around “the current state of the project”. If we behaved the same as Arrow, no more releases after 1.0 since there have been no changes. Long term will the users be users of the spec? Or users of the tools consuming the spec? So long term, this may not be a big deal, as this isnt changing as much.
    - Jeroen: The 1.0 release is a good spot for that state where developers think there might not be any changes. Kind of liked the idea of using minor version number being used for breaking changes at the moment.
    - Jacques: Let’s create a document and iterate on that.
- Capabilities discovery
    - Weston: Had been thinking about this recently. Not ready to work on in great detail but curious what people’s opinions were on the feature
    - Jacques: Has thought about this a lot and can probably write things down at some point. Suggesting tiers of compatibility. There may be peculiar relationships and capabilities of some of these systems. One example might be different levels (tier 1, tier 2, …) Another might be different levels combined with different dimensions (types, logical operations, physical operations, …) Realistically it is probably going to be a long time before we get to a point where engines can understand dynamically what other engines are capable of but we will get there. Start with packages of capability defined by hand and start learning the language / terms needed to express capabilities.
    - Carlo: I agree with what you’re saying about starting with some concrete examples by hand and see what it looks like and learn how to do this. Not clear

on whether we will get to an explicit tiering or not. Every engine will likely have different opinions on what things are high priority. Not sure there would be consensus on tiers (or there would be lots of exceptions). Maybe names will come for tiers once we start to see patterns over time but hard to predict it ahead of time. Whether this is here or not, tools will likely have clear contracts outside Substrait about the capabilities and Substrait will make good documentation of this.
    - Jesus: We will likely need to understand differences at a very fine level of granularity. Grouping in tiers is good but the finer granularity will still be a need.
    - Jacques: Whether tiers come in sooner or later will depend on the profile of consumption. For example, defining a tier / group of capabilities is a way to communicate a common definition for new systems adopting Substrait. A new consumer, for example, can know if they define everything in "tier A", they will be consumable and usable by a set of producers. Allows the tools to be out of sync with each other and decoupled. Similar to Arrow. Early adoption was just connections between systems. Later, new tools came around which had Arrow as the internal representation. Substrait likely to follow similar pattern.
- How to review the validator PR?
    - Jeroen: Since this is basically the first set of commits for a new project, not sure a detailed review is possible. Could split things up into a series of smaller PRs but it would be a long series and changes in the early PRs would have significant rebase cost. Would prefer to avoid.
    - Weston: Agree that a full detailed review is unlikely. Volunteering to take a look at the C interface at some point. Detailed review implies shared ownership, not sure there is a second owner at this point.
    - Jacques: Agree about shared ownership. Will the validator be Jeroen's thing? Or a shared thing? Barrier for entry is pretty high. Is it possible to avoid this sort of thing? Similar in some ways to the Java patch although the Java patch is almost required for people working in Java but the validator is more optional.
    - Carlo: In the future we want to try and avoid creating mega-PRs. It doesn't make sense to rewrite now. We can still file issues as we use it and fix it up as we go on. If that's not working we can change course at a later date. Better to go with it and deal with issues if they occur. Being able to validate plans is super valuable. Let's make sure we keep it sufficiently isolated to cut it if it ends up bit-rotting.
    - Ashvin: Agree the tool is important and the PR is a good idea. Agree it is not feasible to split it now. Can we create some kind of high level code walkthrough for people that want to make changes later on?
    - Jeroen: I have been including readmes in each directory and I think the inline documentation is pretty thorough. There are a few complications with compiling protobuf but that part of the code is likely to be pretty static going forwards and maybe doesn't need to be understood by that many people. Always happy to sit down for a day and do a deep dive if we think that will be helpful.

    - Jacques: We want to get to a place where there are active maintainers of a project. Need to think about "what are the things we can do to encourage that?" Substrait faces a similar problem. We often have to write more than we normally write, create extra tools (e.g. Isthmus cli tool) to lower barrier of entry, etc. I do think it is important to the community we have multiple owners of the validator in the future. Is there anything we can build that will make it easier to future devs to drop in and fix bugs?
    - Jacques: Separate request, can we make sure the messaging in the validator is clear that the validator itself, and the spec, are experimental so it is clear to users that an error from the validator doesn't mean 100% the implementation is incorrect.
    - James: As Substrait evolves it seems the validator will need to change, almost in lockstep, with the validator. Would we want to get to a point where any change in Substrait must accompany a change to the validator?
    - Jeroen: That was the goal. However, since the validator is in a different repository, that is probably not possible. The validator does include the Substrait version so it knows what version it is validating against.
    - James: Do we (should we) have any gold files / plans that we all agree are correct, which can be used to address issues in the validator?
    - Jacques: Gold files are something we want at some point. The current binary representation is not very friendly towards this. Will be hard to get to gold without text representation.
    - Jeroen: Will be hard to get to gold without implementing systems to test against. The validator has some files from Isthmus that it uses for internal testing as "gold" files.
    - Carlo: Working on round tripping to SQL server execution plans. Then we will likely be able to share some of these queries.
- Governance
    - Jacques: Working on a proposal
- Materialized view engine
    - Jacques: If you have a plan can we generate alternatives? Would love to start working on this as a community.
- Rewrite engine:
    - Jacques: Lot's of interest in starting a conversation around this engine. Proposal for this (and the above item) is to start in Rust, targeting a reusable C API.
- Data governance:
    - Ashvin: We have seen data governance be a very important consideration for many users. Would like to use Atlas / Substrait integration to get much of this for free.
    - Jacques: Not something I had previously thought about but it looks like Atlas makes sense as a type of Substrait consumer. As a library it could be useful to get this kind of information out of a Substrait plan so it can make sense in a Substrait project. Also makes sense for this to be a component of Atlas itself.

    - Ashvin: Traditionally, the converter has been alongside the engine and not part of Atlas. The engine is responsible for pushing the data to Atlas and Atlas doesn't have capabilities to pull the data.
    - Jacques: Would be helpful to have a real use case end-to-end.
    - Carlo: I'm involved in a project called purview pulling in provenance tracking data from as many spots as possible. Substrait will enable collection from a large number of systems. If purview supports ingestion from Substrait then that means engines can get provenance tracking as a feature merely by adopting Substrait.
    - Jacques: Is there a way to make this available as a CLI in the Substrait project? For example, could I type output from Isthmus into this tool and spit out the provenance tracking information to understand what all can be done? For example, with Calcite, having a suite of basic tools was very valuable to explaining what the capabilities of the tool are. Would really love a plan visualizer for Substrait as a tool like this as well.
    - Jeroen: FWIW: The validator does spit out an HTML visualization.
    - Jacques: Let's put it an example of this on the website
    - Carlo: Let's use this to visualize plans on the website.
    - Jacques: All of this is a good conversation. We should be thinking about how we build these kinds of things because they will be essential to teaching the project to new users.
- Misc
    - Carlo: Microsoft has a lot of internal queries that can't be shared. However, it might be possible to use these internally to round-trip test Substrait and then share what queries failed to round trip (or more accurately, some vague information about capabilities that failed to round-trip).
    - Jacques: How do we start getting more (paid for) CI infrastructure up and running. People also want to do ML research against queries (e.g. similarity analysis). Long term question: is there a way to expose this corpus privately so that people can run their ML training against it to build a model?
    - Carlo: Let's also start building up an open source corpus of queries. Motivation to contribute will be that you can help prevent Substrait changes from breaking your existing queries. Could even someday offer a bounty for people that can write queries that break Substrait. Lots of interesting things here.

May 11, 2022 8AM PDT

Agenda

- Introductions / how are people planning on using Substrait?
- Rules of engagement / expectations for contributors/committers
    - Jacques to create ticket that covers some key tasks we need to include
    - CLA/ICLA, governance doc, compatibility guarantees, contribution guidelines, commit rules (lol!), etc.

- Cross-projects dependencies/release structure and expectations (for community and users)
- Anyone working on UDFs? Growing interest in Arrow ML in being able to serialize python UDFs
- Is the Github discussions feature working out? Do we want to add a mailing list?