Monthly TSC meeting

The OpenLineage Technical Steering Committee meetings are **Monthly on the Second Wednesday from 9:30am to 10:30am US Pacific.** Here's the meeting info.

All are welcome.

Next meeting: May 8, 2024 (9:30am PT)

April 10, 2024 (9:30am PT)

Attendees:

- **TSC:**
  - Julien Le Dem, OpenLineage project lead, LF AI & Data
  - Michael Robinson, Community Manager, Astronomer
  - Harel Shein, Lineage at Datadog
- **And:**
  - Sheeri Cabral, Product Manager, ETL, Collibra
  - Eric Veleker, Partnerships, Atlan
  - Jens Pfau, Engineering Manager, Lineage, Google
  - David Twaddell, Architect, HSBC

Agenda:

- Announcements
- Recent release highlights
- Discussion items
  - supporting job-to-job dependencies in the spec
  - improving naming conventions
- Open discussion

Meeting:

- Slides

March 13, 2024 (9:30am PT)

Tentative agenda:

- Announcements
- Recent release 1.9.1 highlights
- Scala 2.13 support in Spark overview by @Damien Hawes
- Circuit breaker in Spark & Flink, built-in lineage in Spark @Pawe Leszczyski
- Discussion items
- Open discussion

February 8, 2024 (10am PT)

Attendees:

- TSC:
  - Julien Le Dem, OpenLineage project lead, LF AI & Data
  - Michael Robinson, Community Manager, Astronomer
  - Damien Hawes, Booking.com
  - Harel Shein, Datadog
  - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage committer
  - Mike Collado, Sr. Software Engineer, Snowflake
- And:
  - Suraj Gupta, Atlan
  - Eric Veleker, Atlan
  - Sheeri Cabral, Product Manager, Collibra
  - Ernie Ostic, IBM/Manta

Agenda:

- Recent releases
- Announcements
- Coming soon: simplified job hierarchy in the Spark integration
- Discussion items
- Open discussion

Meeting:

Notes:

Summary

1. We have added a new communication resource, a LinkedIn company page.
2. We announced a new committer, Damien Hawes, from Booking.com, who has made significant contributions to the project.
3. Astronomer and Collibra are co-sponsoring a data lineage meetup on March 19th at the Microsoft New England Conference Center.
4. Members have talks upcoming at Kafka Summit and Data Council.
5. We discussed upcoming improvments to job hierarchy in Spark and how this can help answer questions about job scheduling and dependencies.
6. Damien shared his contributions to the Apache Spark integration, specifically addressing versioning conflicts with Scala.
7. Eric provided a general update on the interest in and adoption of OpenLineage, particularly in the enterprise space.
8. Atlan is considering releasing a DAG (Directed Acyclic Graph) instead of a plugin to help users with configuration and troubleshooting.
9. The next monthly call will be held at a different "location," and participants were encouraged to look out for the updated Zoom link.

Outline

Welcome and Announcements
- Michael Robinson welcomes everyone to the monthly call of the Open Lineage TSC, which is recorded and archived on the wiki. He mentions that the list has one more person since the last meeting and teases an exciting announcement.
- Michael Robinson shares a new communication resource, the LinkedIn company page, and asks for quick introductions from the participants.
- Harel introduces himself as an Open Lineage committer and hints at an interesting workplace announcement for the next meeting.
- Other participants introduce themselves, including their roles and companies.

Introductions
- Maciej introduces himself as a software engineer and warns about possible background noise due to copyright music.
- Eric, Suraj, and Damien introduce themselves and express their excitement to be part of the call.

Agenda Overview and New Committer Introduction
- Michael Robinson outlines the agenda for the call, including announcements, recent releases, and discussion items.
- Michael Robinson announces a change in the Zoom link and welcomes a new committer, Damien from Booking.com, who has made significant contributions to the project.
- Harel and Michael Robinson express their gratitude for Damien's contributions and explain how he added support for multiple Spark versions for the integration, which saved a lot of time and effort for the community.

Upcoming Events
- Michael Robinson announces a data lineage meet up on March 19th at the Microsoft New England Conference Center, co-sponsored by Astronomer and Collibra. More details and sign-up link available on Meetup.com.
- An updated agenda and information about speakers will be provided soon.
- Michael Robinson informs about an exciting talk at Kafka Summit on March 19th called "Open Lineage for Stream Processing" by Baimache and Pavel. There will also be a data standardization panel moderated by Julien at Data Council on March 27th, with participants to be finalized soon.

Recent Releases and Contributions
- Michael Robinson shares about the successful first London meetup with speakers from Decathlon, Confluent, and Astronomer. Decathlon's lineage graph was showcased, and more details about their architecture and use case will be shared in the future.
- Open Lineage 1.8 was released with contributions from Damian, Mata, Meta, Bertel, Peter, and Natalie.
- Michael Robinson thanks all contributors and welcomes Matea's first contributions to the project. Open Lineage 1.8 can be read about on the GitHub repo and docs.
- Maciej is asked if he would like to share his screen.

UI Feature and Streaming Integration
- Maciej explains two topics for the call: a store and a description of how they think of job-specific park. He discusses the job hierarchy and how they can answer questions about why a job ran at a certain time.
- He gives an example of a parent job and how it schedules events. He explains that for a spark job, there can be multiple events and actions, but they want to simplify it to one event at the start and one at the end with each action having a parent job.
- He gives a complex example of a sequence of events for a spark application. He explains that open consumers can collapse the information they receive for a simplified view of the spark application.
- Maciej explains the new UI feature that allows for a top-level view of data in spark levers, without distinguishing the internal actions. He also mentions the higher level execution feature that allows users to see what is scheduled across the platform.
- Harel praises the addition and mentions that it helps visualize dependencies and governance, making it easier to answer use cases visually. Maciej adds that the complete events feature allows users to know when a spark drop ended.
- Michael Collado asks about how well the feature works in the data bricks environment, which Maciej acknowledges as a great question and mentions that they need to try it more in data bricks, as it is always slightly different from the standard.
- Maciej explained that they wanted to have a streaming integration with Pink, which is currently the most popular streaming system. They had an idea to make a Pink integration, but the code they copied from the integration was not very beautiful and had a lot of reflection and instance checking.
- They decided to create a workaround to get as much value as they can and propose an interface that allows them to create a better integration. They had other things to do in the meantime, but then they discovered that a support customize job was created by Dance, which introduced several interfaces.
- They realized that the perfect interface was already created, but it had only one piece of information. The problem was that the IP had already passed, and the listener would have to know every connector Emerson to get information from it, which is impossible.
- Maciej explains the limitations of open source connectors and how it affects their integration process. They have resolved this issue by adding a data set interface to make connectors implement it and make the lineage vertex implement the list of data sets that they actually attach.
- This breaks the capping between the collector and listener because they both are bigger face that basically doesn't change and changes. It takes only forward compatible.
- The end result is that they have an interfacing thing that is open lineage but not quite named open lineage. This solution is easier to convince the community to create an interface, that there's concerns is done to be find like on a library that the third part and they can have a clear one to one mapping without breaking anything.
- Maciej asks if there are any questions.
- Michael Robinson thanks Maciej for his contribution and acknowledges that he joined the call after work hours. Julien also thanks Maciej for coordinating with the link commuters on this great collaboration.
- Eric offers to give a general update at the end of the call.

Open Discussion
- Michael Robinson moves on to open discussion and asks if anyone has any discussion items.

Update on Spark Integration
- Damien shared his experience with the scalar two point 13 support to Apache Spark integration. They deployed the open line spark integration into their own internal pipelines and it worked well.
- However, when they moved to new clusters running different versions of scalar, the jobs failed due to conflicting scalar major versions. The reason for this is that when Java code is compiled, the compiler injects the full class names or full type signature of a method, which includes what its return type is and what its input ran types are.
- When calling a method in Apache Spark, if the same method has two different types signatures, the JVM throws a runtime error. The solution to this is to compile the entire application for an entire project against the Apache Spark libraries.
- Damien explains how to configure the app to consume relevant jars and run integration tests for different versions of Spark, with the exception of Spark 2.4 which only uses Scala 2.12. Maciej thanks Damien for his contribution and expresses a desire for faster reviews.
- Michael Robinson congratulates Damien on becoming a committer and thanks him for his contributions. Eric provides a general update on interest in airflow and spark integrations, with a focus on enterprise adoption and versioning conflicts.
- They plan to release a Dag instead of a plugin to help with configurations. Michael Robinson concludes the call and announces the next meeting.

January 11, 2024 (10am PT)

Attendees:

- **TSC:**
  - Julien Le Dem, OpenLineage project lead, LF AI & Data
  - Harel Shein, Datadog Engineering
  - Michael Robinson, Community Manager, Astronomer
- **And:**
  - Tatiana Al-Chueyr, Staff Software Engineer, Astronomer
  - Alex Jaglale, Executive, DataGalaxy
  - Jens Pfau, Engineering Manager, Google
  - Eric Veleker, Atlan

Agenda:

- Recent releases
- Announcements
- Discussion items
- Open discussion

Meeting:

Notes:

Summary

1. We closed their first ever annual ecosystem survey and the results will be published soon.
2. There is a meetup coming up on January 31st in London, which will be our first in London. It will be an in-person event.
3. We have a talk at the Kafka Summit in London in March, with key contributors speaking.
4. We recently released version 1.7.0, with important compatibility notice for the Airflow integration.
5. There was a discussion about possible improvements to job hierarchy semantics in the Spark integration.
6. Julien updated the registry proposal and it is close to being implemented.
7. Eric (Atlan) shared that there is growing demand and adoption of OpenLineage, and organizations are pressing forward due to the perceived business value.
8. Eric mentioned the need for better documentation and support for different versions and integrations.
9. Jens suggested expanding the integration matrix to include more dimensions, such as types of data sources and facets.

Outline

Announcements and events
- Michael outlines the agenda for the meeting, including announcements, recent releases, updates on Airflow provider and Spark integration, and discussion items.
- Michael announces upcoming events, including the publication of the annual ecosystem survey results, a meetup in London on January 31st, and a talk at Kafka Summit in March.
- Alex asks if the meetup will be online as well, and Michael clarifies that it will be in person only.

Release updates
- Michael Robinson informs the participants about the recent release of version 1.7.0 and mentions an important compatibility notice regarding the airflow integration. He encourages the use of their official open lineage airflow provider and explains that the transition is easy and straightforward.
- He also mentions the addition of the parent run facet to all events in the airflow integration and the removal of support for airflow two. Michael thanks all contributors, including Koch Bermuda, who provided fixes for the release.

Spark job hierarchy
- Michael Robinson plays a recorded update from Maciej, who provides important updates on the provider and ongoing discussions of possible improvements to job hierarchy schematics in the spark integration. Julien acknowledges the recording, and Maciej provides updates on recent changes to the Airflow Provider.
- He mentions the addition of support for multiple GCS industry-related operators and bug fixes.
- He proposes having more granular event semantics and consensus on having a single parent run for all actions.
- Inputs and outputs of the job hierarchy for spark and the need for more information about how they are related are discussed. They mention the open lineage feature called "parent" that allows specifying that a run was scheduled by something else or is a sub-run.
- They agree on having a single parent run that contains all actions but note that it is still being discussed.
- Maciej explains how the application run and parent run work, allowing customers to correlate jobs and understand execution. He mentions the power it gives to consumers who want to display aggregate data and make sure users understand how jobs look like.
- Maciej shares links to issue #1672 and the PyPI doc and download for the Airflow Provider, encouraging questions or contributions to the ongoing discussion.

Simplify jump a key
- Michael invites discussion on topic #1672 and asks if anyone wants to add a topic. Jens brings up the simplify jump a key for spark issue and suggests having a quick discussion on it.
- Julien explains that the explanation they just saw was recorded because Maciej couldn't join the meeting. Jens realizes his check is not there and will discuss it with Maciej separately.
- Michael asks if there are any other items for discussion.

Registry proposal
- Julien updates the registry proposal and shares his screen to show the recent updates, including clarification for consumers to independently discover and support custom facet opening, acceptance guidelines for claiming a name and entity, and examples of how to use them. He believes it's close enough to implement the first version and see where they're going.
- Julien reviewed the recording of a meeting and integrated feedback. The core facets will be moved in the registry under the core name and follow the same rules as all other custom facets.
- Examples for each facet will be moved to the registry as well, ensuring consistency and validation. Additional metadata is available to show documentation on use cases.
- The first version of the registry will be managed and improved over time. Jens asked about the format for spec versions, which could be extended.
- Michael Robinson expressed happiness with the progress and thanked Sheeri for driving the conversation.
- Jens asked about the format for spec versions and Julien explained that it's currently exact only but could be extended. He suggested tagging Sheeri to discuss further on the extension of these versions.

Learning since last call
- Eric shared some learning since the last call.
- Eric reports growing demand and adoption of OpenLineage, with no hesitancy from organizations due to its perceived business value. He mentions the need for better documentation to accelerate adoption and optimize for speed in two areas: proper versions of everything in place and diagnosing if there are needs that the community needs to build out for support.
- Eric suggests an Airflow plugin to provide a report on misconfigurations and help stakeholders understand the details. He also mentions the need for access to the boundary or threshold of support to get organizations up and running and showing business value.
- Michael Robinson asks Eric about a specific document that would be helpful for version requirements and coverage information. Eric explains that the plugin they developed will identify things that need to be done to press forward for the organization implementing the lineage.
- He gives an example of an organization using AWS Glue and how they had to throw on the brakes because they didn't have knowledge of the community's investment in building up support where it's needed. Eric puts out a problem statement about the need for all the folks adjacent to the core community to know the boundary or threshold of support to get organizations implemented and up and running.
- Michael Robinson and Julien acknowledge the information.

Column lineage in Spark
- Eric explains that they have been reaching out to the community for information about coverage, but having it in one place would be helpful. He encourages opening issues and shares that a new resource on the subject is available.
- Julien agrees.
- Michael Robinson asks if anyone else has similar experiences.
- Eric asks if anyone else has experienced the same.
- Jens confirms that he understands the question and suggests having the information in a single place would be helpful.
- Eric thanks Jens.
- Michael Robinson shares a new resource on the subject and encourages opening issues. He asks Eric about plans for a plugin.
- Eric was looking at the repo and asks Michael to repeat the question. Michael asks about plans for the plugin.
- Eric suggests following up in the community slack and promises to contribute.
- Michael Robinson acknowledges Eric's contribution.

Integration matrix
- Jens suggests expanding on the integration matrix and mentions issues with iceberg support in Spark.
- Eric reflects on Jens' suggestion.
- Michael Robinson thanks Jens for the input.

December 14, 2023 (10am PT)

Attendees:

- **TSC:**
  - Julien Le Dem, OpenLineage project lead, LF AI & Data
  - Harel Shein, Datadog Engineering
  - Michael Robinson, Community Manager, Astronomer
  - Mandy Chessell, Egeria Project Lead
  - Pawel Leszczynski, Software Engineer, Astronomer/GetInData
- And:
  - Eric Veleker, Atlan

Agenda:

- Recent releases
- Announcements
- Proposal updates
- Open discussion

Meeting:

Notes:

Summary

1. Harel Shein provided announcements about upcoming meetups and shared metrics on community growth.
2. Harel Shein discussed the release of version 1.6.2, highlighting new features and bug fixes.
3. Harel Shein shared metrics on contributors and commits, showing an increase in both.
4. Jens Pfau presented two proposals for column-level lineage, focusing on transformation types and descriptions.
5. Mandy Chessell suggested including the name of the masking function as an additional property for masking transformations.
6. Harel Shein expressed appreciation for the proposals and encouraged community members to review and provide feedback.
7. Eric Veleker expressed gratitude for the momentum and adoption of open lineage, thanking the community for their hard work.
8. Harel Shein echoed Eric's sentiments and acknowledged the project's growth and industry standard status.
9. Harel Shein thanked all contributors and adopters for their contributions to the community.

Outline

- Michael Robinson from Astronomer welcomes everyone and goes through the agenda, which includes brief announcements, a release update, metrics on community growth, an update on dataset support in Spark, and open discussion items. He also reminds participants about the ecosystem survey and announces an upcoming meetup in London co-hosted with Confluence.
- He shares the success of a recent event in Warsaw and thanks contributors and attendees.
- Michael Robinson provides details on the recent release (1.6.2) which includes support for version 1.5, metadata sending without running a dbt command, and improvements to job listeners and lineage in Flink and Spark. He also mentions bug fixes and contributions from new contributors.
- He shares exciting news about streaming job support in Marks project and expects a larger release soon.
- Michael Robinson moves on to share some metrics on momentum and new partners added in the last year, including Google Cloud, Grai, and Metaphor. He directs participants to GitHub and the revamped ecosystem page at OpenLineage for more details.

Metrics on Community Growth
- Michael Robinson shares insights from the lfai and data dashboards, showing increases in total and active contributors as well as commits.
- Harel shared that there may be an issue with the way commits are being counted, but the general trend of 5,000+ commits per month is accurate. He also shared details about their global community membership and contributors using the Orbit tool.

New Job Facet: "Job Type"
- Pawel Leszczynski presented a new job facet called "job type" which contains information about processing type, integration, and pricing on the query command. This job type is used for streaming jobs and is already being implemented in their Link integration.
- Harel thanked him for the presentation.
- Harel expressed excitement about seeing events stream into Marquez and Pawel shared that they are able to merge the PR, but there are still some issues with CI.

Open Proposals Discussion
- Harel expressed excitement for an upcoming release and suggested that encouraging messages on Marquez might help. The next item on the agenda is discussing open proposals.
- Jens discusses two proposals related to the column level line asset, which have been discussed with Aba. He explains the current state of the column level line and its issues, including the lack of a clear contract between producers and consumers regarding transformation types.
- The first proposal is to create a taxonomy of types to address this issue. The second proposal addresses situations where the transformation type would be different for a given pair of input field and output field.
- Jens presents a document with more details on transformation types for column level lines, which should be complete, disjunct, unambiguous, and optional. He also proposes adding a transformation sub-type for more extension.
- Jens proposes adding a subtype and a separate field for masked transformation, creating a transformation object, and moving the fields related to transformation into their own object. Papa suggests adding a masked field to allow users to send information if they wish to.
- Harel asks about adding the name of the masking function as its own property, and Mandy suggests it could be a free form name or an extra property for masking algorithm. They agree to swap the masked field into the name of the mask, and recognize that masking can mean different things in different use cases.
- They discuss the possibility of coalescing on some naming convention or using reference data management to control values.
- Jens asks Mandy to check the GitHub issue with the proposal and provides the slide number. Harel links both proposals in the chat.
- Mandy thanks Harel for doing the proposal.
- Harel expresses gratitude for the proposals and invites others to open a proposal on the project. The next item on the agenda is discussion, but there are no items for this month.

Reviewing New Core Facets
- Jens asks about the process for reviewing new core facets and suggests discussing them before they get merged. Pawelleszczynski explains the process of creating a JSON file and creating a PR, and suggests waiting a few weeks for others to review the proposal.
- Jens agrees and suggests highlighting spec changes more frequently.
- Pawel suggests asking Julien for review and acknowledges that it may take longer during Christmas time. Harel emphasizes the responsibility of the community to each other and suggests allowing for more duration before merging and releasing.
- Eric presents another item on the agenda.
- Harel thanks everyone for their input and moves on to the next item on the agenda.

Adoption of Alina
- Eric shares details on adoption of Atlan supporting different flavors of implementation and how brands adopting OpenLineage speak to the momentum of the community building. He thanks all committers for backing something that's making a difference in the data ecosystem.
- Harel echoes Eric's words and appreciates everyone who contributed over the past few years, making this project an industry standard. He thanks all contributors and adopters like admin, Google, and everyone else on the call and in the ecosystem.

November 9, 2023 (10am PT)

Attendees:

- TSC:
  - Pawe Leszczyski, Software Engineer, GetInData
  - Julien Le Dem, OpenLineage project lead
  - Michael Robinson, Community team, Astronomer
  - Jakub Dardziski, Software Engineer, GetInData
  - Harel Shein, Engineering Manager, Astronomer
  - Maciej Obuchowski, Software Engineer, Astronomer/GetInData, OpenLineage committer
  - Pawe Leszczyski, Software Engineer, Astronomer/GetInData
- And:
  - Eric Veleker, Atlan
  - Harsh Loomba, Engineer, Upgrade
  - Sheeri Cabral, Product Manager, Collibra
  - Peter Huang, Software Engineer, Apple
  - Jens Pfau, Engineering Manager, Google
  - Shubhambharadwaj, Associate Manager

Agenda:

1. Announcements
2. Recent releases
3. Recent additions to the Flink integration
4. Recent additions to the Spark integration
5. Proposal updates
6. Discussion items
7. Open discussion

Meeting:

Notes:

Announcements

- A warm welcome to new committer Harel Shein (harels)! Harel's main contributions have been to project leadership, facilitating discussions, and advocating for the project. Thanks, Harel!
- Upcoming talks include one by Pawe Leszczyski at the Data Science Summit in Warsaw/online, November 23-24, and another by Julien Le Dem at Scale By The Bay in Oakland, CA, on November 15.
- The call for papers deadline for Data Council has been extended to November 17th.

Recent Releases

- OpenLineage 1.5.0
    - Added
        - Flink: add Flink lineage for Cassandra Connectors #2175 @HuangZhenQiu
        - Spark: support rdd and toDF operations available in Spark Scala API #2188 @pawel-big-lebowski
        - Spark: support Databricks Runtime 13.3 #2185 @pawel-big-lebowski
    - Changed
        - Airflow: loosen attrs and requests versions #2107 @JDarDagran
        - dbt: render yaml configs lazily #2221 @JDarDagran
    - Thanks to all the contributors, including new contributor **@sophiely**!

Recent Additions to the Flink Integration - Peter Huang (Apple)

- I work on the Flink team at Apple with a focus on meeting legal requirements
- Current priorities include improving lineage from Iceberg
- Users here also employ Cassandra, so we have contributed Cassandra support
- Apple has an open-source contribution review process, and I can't contribute more at the moment
- I hope that the review process will be completed in the coming weeks, so we can make more contributions
- Planned improvements include:
    - addition of more catalog information to Iceberg lineage
    - support for Flink 1.18

Recent Additions to the Spark Integration - Pawe Leszczyski (GetInData)

- Added support for Spark 3.5
- Added support for Databricks Runtime (most recent version)
- 2188: fix in Scala integration
    - RDD issue was hard to reproduce
- 2233: Jackson library upgrade
    - Jackson library in the project was an old version
    - upgrade includes a security vulnerability fix
    - merged but not yet released
- Planned:
    - Support for Iceberg and Delta for Spark 3.5
    - Spark parentRun AKA Spark Application Events (by mobuchowski)
    - Meetup talk: "How to become a spark-openlineage contributor in 5 steps?"

Proposals in Discussion - Julien Le Dem (Project Lead)

- Open proposals:
    - 2187: ColumnLineageDatasetFacet
        - privacy use cases
    - 2186: formalizing transformation types
        - column lineage facet improvements
    - 2163: define an integration certification process for OpenLineage
        - defines integration certification process
        - currently collecting use cases
        - related to registry proposal
        - input/feedback needed
    - 2162: dataset support in Spark LogicalPlan Nodes
        - optional API we could add to the Nodes
        - prototype coming soon
    - 2161: registry of producers and consumers
        - comments welcome on the PR on GitHub
        - producers would be able to register custom facet prefix, URI and link to documentation, etc.

- consumers would be able to declare the facets you consume, link to documentation, etc.
- name registration:
    - unique naming
    - name would be used in shorter URI prefixes
- CI validation would enforce consistent facet naming and validate facet schemas
- documentation would be published automatically
- additional documentation for specific use cases
- self-contained registry containing all facets for producers and consumers
    - name path in registry with CODEOWNERS file for delegation to circumvent review process
    - path for facet JSON
    - more information
- Pros:
    - producers and consumers would be able to define codeowners to approve changes to the registry
    - CI could guarantee that changes would not produce inconsistencies
    - producers would not need to host and maintain their own subset of the registry
    - publication would be automated
    - freedom and independence for defining custom facets without the project being a bottleneck
- Cons:
    - registered entities would have to maintain their list of codeowners
- Q&A:
    - producers that define multiple facets?
        - granularity of this and other aspects might or might not be desirable
    - consumed facets: mandatory or optional?
        - always optional
    - custom facets or core facets?
        - core facets currently in a different dir, but it would be nice to move them to the registry
    - add tests as with core facets?
        - would be useful as examples and for validation
        - could be optional
        - please add this to the PR

October 12, 2023 (10am PT)

Attendees:

- TSC:
    - Pawe Leszczyski, Software Engineer, GetInData
    - Julien Le Dem, OpenLineage project lead
    - Michael Robinson, Community team, Astronomer
    - Jakub Dardziski, Software Engineer, GetInData
    - Willy Lulciuc, Marquez Project Lead
- And:
    - Harel Shein, Engineering Manager, Astronomer
    - Harsh Loomba, Upgrade
    - Sheeri Cabral, Product Manager, Collibra
    - Ernie Ostic, Manta Software
    - Jeevan Paul, Accel Data
    - Ann Mary Justine, Research Engineer, HP Enterprise's CMF team
    - Jason Yip, Grainger
    - Sunder, JLR
    - Peter Huang, engineer at <>, on Flink team
    - Jens Pfau, engineering manager at Google working on GCP
    - Martin Foltin, member, HP Enterprise's CMF team
    - Austin Bennett, architect at Chartboost
    - Eric Veleker, Atlan

Agenda:

1. Announcements
2. Recent releases
3. Airflow Summit recap
4. Tutorial/demo: migrating to the OpenLineage Airflow Provider
5. Discussion: observability for OpenLineage+Marquez
6. Open discussion

Meeting:

Notes:

Announcements

- The first annual Ecosystem Survey is still open. Submit your response today: https://bit.ly/ecosystem_survey
- Our next meetup will be on November 29th in Warsaw, Poland, at Google. Sign up: https://www.meetup.com/warsaw-openlineage-meetup-group/events/296705558/?utm_medium=referral&utm_campaign=share-btn_savedevents_share_modal&utm_source=link

Recent releases

- 1.2.2
  - **Added**
    - Spark: publish the ProcessingEngineRunFacet as part of the normal operation of the OpenLineageSparkEventListener #2089 @d-m-h
    - Spark: capture and emit spark.databricks.clusterUsageTags.clusterAllTags variable from databricks environment #2099 @Aniruddh181001

  Thanks to all the contributors, including new contributors **@d-m-h, @tati** and **@xli-1026**!
- 1.3.1
  - **Added**
    - Airflow: add some basic stats to the Airflow integration #1845 @harels
    - Airflow: add columns as schema facet for airflow.lineage.Table (if defined) #2138 @erikalfthan
    - DBT: add SQLSERVER to supported dbt profile types #2136 @erikalfthan
    - Spark: support for latest 3.5 #2118 @pawel-big-lebowski

  Thanks to all the contributors, including new contributor **@erikalfthan**!
- 1.4.1
  - **Added**
    - Client: allow setting client's endpoint via environment variable #2151 @mars-lan
    - Flink: expand Iceberg source types #2149 @HuangZhenQiu
    - Spark: add debug facet #2147 @pawel-big-lebowski
    - Spark: enable Nessie REST catalog #2165 @julwin

  Thanks to all the contributors, including new contributors **@julwin** and **@HuangZhenQiu**!

Migration from standalone Open Lineage package to Airflow provider
- Jakub explained how to migrate from the standalone openly the flow package to the airflow provider. He gave reasons why they wanted to become an airflow provider, including making sure that the metadata collected in airflow is not breaking airflow itself.
- They also keep the latest code up to date with all the providers and become part of these providers of the operators. There were a couple of changes introduced in the provider package, and the main question is how to migrate.
- The simplest way is to just do the install for the specific package. One of the things they would like to walk away from this customer structures, and there was and still is a possibility to write a customer structure that was controlled by the open infrastructures environment variable.
- Jakub explains that if a user has implemented some get open age assets method previously based on the old module and class, they do not need to worry about it because it is translated. However, if they install opening flow, they will fail to import the old class and need to change the import path.
- There are changes introducing configuration, and there is a whole section called open image in conflict. Many of the features that were previously available in opening package are also compatible with the provider.
- People usually like open in URL, which is pretty common and still works. But some entries in the open in age section take precedence over what's been previously handled by environment variables.
- Jakub gives examples of how the logic for like conflict takes precedence over open in URL. He mentions that the documentation has more information on how it works.
- He also explains how to add new integration in the provider or other providers that make use of opening provider. They want to give up on using open in age common data set module and use just the classes from the open in age python client.
- Jakub gives quick advice on how to grab some information from execution of the operator. Previously, when they didn't have any control or influence on how to grab some information from execution of the operator, they needed to read the code and see that maybe job ID is returned as an ex come.
- Now when they added the integration in the query operator itself, they can just change the code so it saves the job ideas and attributes.
- Jakub gives a quick demo of how it works. He is using breeze, which is a mostly development environment and cli for airflow.
- He is using on two point seven point one and is also using integration open in age, which instant Marcus also that's an option that they have in their flow. The only package that he is using is posts because he'll be using or provider.
- He shows how it works and mentions that the beauty of e-mail life is that he doesn't know if it should work.
- Jakub says that it should work in a minute.
- Jakub types in his password.
- Jakub says that he doesn't need to run post scripts, but actually he doesn't have just to prove he doesn't have any.
- Jakub says that it's working. He is running some example that uses focus as back end.
- Jakub says that previously, there was nothing to configure more if a user has like opening the CR.
- Jakub explains that he changed the next piece and this is development, but the name is changed because he hasn't experimented with something. Eventually, the events came to market.
- Jakub tries it again.
- Jakub demonstrates a quick demo of three options for package installation and rerunning history. Julien thanks Jakub and asks if there are any questions about migration from the old open age integration into the new airflow provider.

Observability for OpenLineage markers
- Julien introduces the discussion topic of observability for opening age markers and invites Harel to start. Harel asks the audience about ensuring liability of lineage collection and what kind of operability they would like to see, such as distributed tracing.
- He suggests gathering feedback on a slack channel. Julien thinks the metrics added to the airflow integration by Harel are a good starting point for observability.
- Hloomba mentions enabling retention policy on all environments and suggests observability on database retention to help with memory or CPU performance. Harel suggests enabling metrics out of the box and instrumenting more functions using drop wizard as a web server.
- Julien and William discuss having metrics on the retention job to track how the data retention job keeps the database small.
- Jeevan asked about the possibility of having an open lineage event for Spark applications, and Pawelleszczynski explained the need for a parent run faster to identify each Spark action as part of a bigger entity, the Spark application. Jens suggested having unique job names for Spark actions and the parent Spark application.
- Pawelleszczynski explained that the current job name is constructed based on the name of the operator or Spark logical note and appended with a dataset name, but they can make it optional to have a human-readable job name or use a hash on the logical plan to ensure uniqueness.
- Harel mentioned having good news for Bob and suggested discussing it next week.
- Jens added that having unique job names would help distinguish each Spark action and its runs, and Pawelleszczynski explained the current job naming convention and the possibility of making it unique using a hash on the logical plan.
- Julien asked if anyone had more comments on the topic.

Creating a registry for consumers and producers
- Julien presented four items and discussed them in detail. The first item was about creating a registry for consumers and producers, which was summarized in a Google doc.
- Two options were discussed, and the second proposal with a self-contained repository was preferred. Notes and open items were added to the document, and everyone was encouraged to contribute to it.
- The second item was about proposing an optional contract for providers for airflow operators to exclude their age. A proposal was made to expose open lineage data set directly into DBT's manifest file, and feedback was sought from DBT contributors.
- The third item was about spark integration, which knows how to define unique data sets based on various data sources. However, custom data sources with their own implementation become opaque, so an optional contract was proposed to address this issue.

Proposing an optional contract for providers for Airflow operators
- Julien presented four items and discussed them in detail. The first item was about creating a registry for consumers and producers, which was summarized in a Google doc.
- Two options were discussed, and the second proposal with a self-contained repository was preferred. Notes and open items were added to the document, and everyone was encouraged to contribute to it.
- The second item was about proposing an optional contract for providers for airflow operators to exclude their age. A proposal was made to expose open lineage data set directly into DBT's manifest file, and feedback was sought from DBT contributors.
- The third item was about spark integration, which knows how to define unique data sets based on various data sources. However, custom data sources with their own implementation become opaque, so an optional contract was proposed to address this issue.

Spark integration
- Julien presented four items and discussed them in detail. The first item was about creating a registry for consumers and producers, which was summarized in a Google doc.
- Two options were discussed, and the second proposal with a self-contained repository was preferred. Notes and open items were added to the document, and everyone was encouraged to contribute to it.
- The second item was about proposing an optional contract for providers for airflow operators to exclude their age. A proposal was made to expose open lineage data set directly into DBT's manifest file, and feedback was sought from DBT contributors.
- The third item was about spark integration, which knows how to define unique data sets based on various data sources. However, custom data sources with their own implementation become opaque, so an optional contract was proposed to address this issue.

Certification process in the Open Lineage ecosystem
- Julien discussed the need for a certification process in the Open Lineage ecosystem, and suggested creating a document to start a discussion on how to implement it. He mentioned the possibility of providing data set support for scans and action notes, and creating a contract for implementing data sources to expose lineage in relation notes.
- Julien also talked about the goal of Open Lineage to be built into systems like Airflow, and encouraged attendees to share their opinions and ask questions on Slack.
- Julien discussed the need for a certification process in the Open Lineage ecosystem, and suggested creating a document to start a discussion on how to implement it. He mentioned the possibility of providing data set support for scans and action notes, and creating a contract for implementing data sources to expose lineage in relation notes.
- Julien also talked about the goal of Open Lineage to be built into systems like Airflow, and encouraged attendees to share their opinions and ask questions on Slack.

September 14, 2023 (10am PT)

Attendees:

- TSC:
  - Pawe Leszczyski, Software Engineer, GetInData
  - Julien Le Dem, OpenLineage project lead
  - Michael Robinson, Community team, Astronomer
  - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage committer
  - Mandy Chessell, Lead of Egeria Project
- And:
  - Harel Shein, Engineering Manager, Astronomer
  - Harsh Loomba, Upgrade
  - Sheeri Cabral, Product Manager, Collibra
  - Ernie Ostic, Manta Software
  - Mars Lan, CTO/Co-founder, Metaphor

Agenda:

1. Announcements
2. Recent releases
3. Demo: Spark integration tests in Databricks runtime
4. Discussion items
5. Open discussion

Meeting:

Notes:

1. Announcements [Julien]
2. Recent releases [Michael R.]
3. Recent Releases
   - Michael shared a release update on 1.1.0, including support for configuring OpenLineage based on the Flink integration, solving the problem of multiple jobs writing to different data sets with the same job name in Spark, and adding missing Java docs to the Java client. The default

behavior can be turned off with an environment variable, and more information is available in the release notes.
- Michael also thanked new contributors and mentioned bug fixes.
- Maciej and Julien discussed the fact that Airflow changes are not included in the changelog and that the Airflow-OpenLineage is now part of the Airflow project.

4. Demo: Spark integration tests in Databricks runtime [Pawel]
   - Pawel thanked the participants and introduced himself. He talked about upgrading the Spark version and the issues they faced with Databricks integration.
   - They had to manually test the changes which was time-consuming. However, Databricks released a Java library that allowed them to run integration tests easily.
   - They also implemented a file transport system to capture lineage events and verify that the events contain what they expected. This change helped speed up their work and have better code.
   - Julien asked if there were any questions.

5. Discussion items
   a. Open Lineage Registry Proposal [Julien]
      - Julien explained the concept of OpenLineage and the need for a registry to define custom facets and producers. He shared a Google doc for feedback and listed the goals of the registry, including allowing third parties to register their implementation or custom extension and shortening the producer and skim URL values.
      - Custom facets are an easy way to extend the spec without requiring any approval, and producers and consumers can do the list of facets they produce without requiring approval.
      - Mandy joined the call and expressed support for the idea of a registry but suggested that facets should be themed to avoid every producer defining their own facets. She proposed having a set of themes like data facets and meeting assets to cluster similar facets together in the registry.
      - Mandy expresses concern about naming custom facets after specific technologies, as it can lead to unnecessary duplication. Julien explains that the airflow facet is specific to airflow and provides benefits for generic things.
      - Core facets are sometimes added, and there are things specific to what people are doing. Mandy agrees and gives an example of how types are aligned with technologies, leading to duplication.
      - Ernie suggests adding a protocol for something in the registry to become a core facet. Julien explains that there is a template for adding to the spector and that custom facets can be defined as long as they have a prefix to the facet name and publish the schema.
      - To become a core facet, a proposal can be opened on the open is project and usage of the custom facet can be leveraged to show that it works.
      - Mandy suggests having a state on the registry to show whether something is private, under proposal, or being adopted. Julien agrees and explains that some custom facets are specifically in the domain of the producer and should live in the registry, while others are shared.
      - Nick interjects and expresses his appreciation for the community aspect of the open lineage. He suggests that producers provide examples and tests for consumers to use.
      - Mandy asks for clarification on what he means by tests, and Nick explains that it could be a set of payloads or actually running the runtime to produce events.
      - Nick would like to see both examples and payloads for consumers and producers, respectively. He suggests that putting them in a registry would facilitate everything all around like the tests.
      - Julien explains that for the core spec, they have the definition of facets, Jason schema for each asset, and documentation. They also added an example of each core asset and a test for the schema validation.
      - He suggests making it easier for producers to describe what facet they're producing.
      - Mandy asks who did the recent addition, and Julien explains that it was part of getting data. Mandy thanks him for the information.
      - Julien suggests that there could be more done to make it easier for producers to describe what facet they're producing. Nick agrees and suggests a framework for testing where producers can provide enough information for the test to be generated.
      - Julien explains that they currently use schema validation, but it's just a small portion of what Nick is describing. Nick agrees that it's a start.
      - Julien suggests that producers need a registry mechanism to create their own facets and make them explicitly defined. Consumers would also benefit from a programmatic definition of facets they're consuming.
      - He mentions the open lineage website's ecosystem page and how it points to documentation, but a more programmatic definition would be great.
      - Nick agrees that it would be great to have a more programmatic definition of facets.
      - Julien proposed a registry and discussed the trade-offs between a self-contained registry and delegating to other registries. He also mentioned the benefits of using shorter URLs for custom facets.
      - Nick asked about how other communities handle this and suggested looking at successful practices of similar organizations. Pawelleszczynski agreed.
      - There were questions about whether there should be a registry folder under spec or in the opening tab organization, and how to handle core facets and versioning. The group discussed using an owners file in a repo to approve updates to the registry.
      - Julien emphasized that this was just to start the conversation and that there were many different ways to implement the registry.
      - Julien mentioned producing a list of schema URL as a third party and discussed the benefits of a self-contained registry, including the ability to run checks against it and ensure consistency.
      - Julien explained that defining a name and putting a list of information would allow for shorter URLs for custom facets.
      - Julien used ol: as an example of a shorter prefix for schema URLs.
      - Julien mentioned that there were questions about whether there should be a registry rep in the opening tab organization and whether it should be a registry folder under spec.
      - Julien discussed using a Jason file to contain information about customers and their defined names.
      - Julien compared the registry to the even repository and discussed using an owners file to approve updates to the registry.
      - Julien mentioned using ti to verify consistency and avoid breaking the registry.
      - Nick asked about successful practices of similar organizations in handling registries.
      - Nick mentioned that smaller organizations might be more flexible while larger organizations might have more legal requirements for using other registries.
      - Pawelleszczynski agreed with Nick's suggestion to look at successful practices of similar organizations.
      - Julien explains that data-driven decisions are important and mentions the trade-off of how complicated it is to maintain a repository and whether it is self-service for producers. He suggests adding files to an existing open source repo for small organizations, while big organizations may need legal approval to contribute.
      - He also mentions the need for licensing and PR processes.
      - Nick responds with agreement.

- Julien shares that he will share the draft dock on Open Lineage Slack for feedback and follow the OpenLineage proposal process. He mentions other ideas for implementation, such as the Men repository and the Evan repository, and welcomes other examples.
- He also asks if there are any questions or things people want to share about OpenLineage.

August 10, 2023 (10am PT)

Attendees:

- TSC:
  - Julien Le Dem, OpenLineage project lead
  - Michael Robinson, Community team, Astronomer
  - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage committer
  - Willy Lulciuc, Marquez Project Lead
  - Mandy Chessell, Lead of Egeria Project
- And:
  - Harel Shein, Engineering Manager, Astronomer
  - Harsh Loomba, Upgrade
  - Peter Hicks, Astronomer
  - Sheeri Cabral, Product Manager, Collibra
  - Ernie Ostic, Manta Software
  - Athitya, Intuit India
  - Cory Visi, Solutions Architect, AWS

Agenda:

1. Announcements
2. OpenLineage 1.0 overview
3. OpenLineage Airflow Provider update
4. Discussion items
5. Open discussion

Meeting:

Notes:

1. Announcements [Julien]
   a. Ecosystem Survey still needs responses: https://bit.ly/ecosystem_survey
   b. OpenLineage graduated from the LF AI on 7/27
   c. The 3rd issue of our monthly newsletter shipped on 7/31. Sign up here: https://bit.ly/OL_news
   d. Upcoming meetups:
      i. 8/30 in S.F. at Astronomer
      ii. 9/18 in Toronto at Airflow Summit
      iii. Marquez meetup on 10/5 in S.F.
2. LF AI Update [Michael R.]
   a. Topics covered by Julien in presentation to LF AI TAC for graduation included trends in adoption
3. Recent releases [Michael R.]
   a. **1.0.0: Added**

      - Airflow: convert lineage from legacy File definition #2006 @mobuchowski

      Removed

      - Spec: remove facet ref from core #1997 @JDarDagran

      Changed

      - Airflow: change log level to DEBUG when extractor isn't found #2012 @kaxil
      - Airflow: make sure we cannot fail in thread despite direct execution #2010 @mobuchowski
      - **https://github.com/OpenLineage/OpenLineage/releases/tag/1.0.0**

        https://github.com/OpenLineage/OpenLineage/compare/0.30.1...1.0.0

   b. **0.30.1: Added**

      - Flink: support Iceberg sinks #1960 @pawel-big-lebowski
      - Spark: column-level lineage for merge into on delta tables #1958 @pawel-big-lebowski
      - Spark: column-level lineage for merge into on Iceberg tables #1971 @pawel-big-lebowski
      - Spark: add support for Iceberg REST catalog #1963 @juancappi
      - Airflow: add possibility to force direct-execution based on environment variable #1934 @mobuchowski
      - SQL: add support for Apple Silicon to openlineage-sql-java #1981 @davidjgoss
      - Spec: add facet deletion #1975 @julienledem
      - Client: add a file transport #1891 @Alexkuva

      Changed

      - Airflow: do not run plugin if OpenLineage provider is installed #1999 @JDarDagran
      - Python: rename config to config_class #1998 @mobuchowski
      - **https://github.com/OpenLineage/OpenLineage/releases/tag/0.30.1**

https://github.com/OpenLineage/OpenLineage/compare/0.29.2...0.30.1

4. Update on the OpenLineage Airflow Provider [Maciej]
    a. Pypi package version 1.0.1 available at: https://pypi.org/project/apache-airflow-providers-openlineage/1.0.1/
        i. installable with `pip install apache-airflow-providers-openlineage==1.0.1`
    b. Development progresses in the Airflow repo
    c. What's there already:
        i. Operator coverage:
            1. A lot of SQL-related operators, especially based on SQLExecuteQueryOperator
            2. Some GCP ones: BigQueryInsertJobOperator, GCStoGCSOperator
            3. Some Sagemaker-related operators
            4. FTP, SFTP operators
            5. Basic support for Python and Bash operators
        ii. Changed:
            1. Airflow: do not run plugin if OpenLineage provider is installed #1999 @JDarDagran
            2. Python: rename config to config_class #1998 @mobuchowski
    d. Next steps
        i. Operator coverage:
            1. Popular operators around BigQuery: BigQueryUpsertTableOperator…
            2. Transport operators, like MySQLToSnowflakeOperator, GCSToBigQueryOperator
            3. S3 support, like S3CopyObjectOperator
            4. Add support for XCom-native operators like BigQueryGetDataOperator
            5. This list is not a promise
        ii. "Core" changes
            1. Add interfaces around OpenLineage-implementing operators - making implementation more native
            2. XCom dataset support - this relates to XCom operators mentioned above
            3. Hook-level lineage support
5. OpenLineage 1.0 with Static Lineage Update
    a. Putting things together for 1.0 release
        i. Important features and PRs
            - Proposal: add static lineage deletion #1839 @julienledem
            - Emit job and dataset runless metadata #1880 @pawel-big-lebowski
            - Marquez: Ability to decode static metadata events #2495 @pawel-big-lebowski
            - Add facet deletion #1975 @julienledem
            - Spec: remove facet ref from core #1997 @JDarDagran

Clarify docs on `RunEvent` lifecycle (link)

July 13, 2023 (8am PT)

Attendees:

- TSC:
    - Julien Le Dem, OpenLineage project lead
    - Jakub Dardziski, Software Engineer, GetInData
    - Michael Robinson, Community team, Astronomer
    - Mandy Chessell, Egeria Project Lead
- And:
    - Anirudh Shrinivason, Data Engineer, Grab
    - Julian LaNeve, Senior Product Manager, Astronomer
    - Harel Shein, Engineering Manager, Astronomer
    - Jens Pfau, at Google working on GCP
    - Alexandre Bergere, DataGalaxy
    - Ernie Ostic, SVP of Product, Manta

Agenda:

1. Announcements
2. Updates
3. Recent releases
4. DataGalaxy integration demo
5. Open discussion

Meeting:

June 8, 2023 (10am PT)

Attendees:

- TSC:
    - Julien Le Dem, OpenLineage project lead
    - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage committer
    - Michael Robinson, Community team, Astronomer
- And:
    - Cori Visi, Solutions Architect, AWS

- Harel Shein, Engineering Manager, Astronomer
- John Lukenoff, Software Engineer, Asana
- Suparna Bhattacharya, HPE Labs
- Ann Mary Justine, Research Engineer, HP Enterprise's CMF team
- Anirudh Shrinivason, Data Engineer, Grab
- Chris Olivares, CTO, Hum Capital
- Martin Foltin, HPE Research Labs
- Sheeri Cabral, Technical Product Manager, Lineage, Collibra
- Harry, works at a Bay area-based fintech firm
- Julian LaNeve, Senior Product Manager, Astronomer

Agenda:

1. Announcements
2. Recent releases
3. Static lineage progress update
4. Open discussion

Meeting:

Notes:

1. Announcements [Julien]:
    a. Our first annual ecosystem survey is live and accepting responses: https://bit.ly/ecosystem_survey. Your participation matters!
    b. We recently published the first issue of our monthly newsletter: https://mailchi.mp/18826f97904e/openlineage-news-may-2023. It's a great way to learn about upcoming meetups and recent blog posts, etc.
    c. Two meetups are happening soon:
        i. New York on 6/22 at Collibra's HQ: https://www.meetup.com/data-lineage-meetup/events/294065396/
        ii. San Francisco on 6/27 at Astronomer: https://www.meetup.com/meetup-group-bnfqymxe/events/293448130/
    d. Upcoming talks:
        i. Pawe Leszczyski and Maciej Obuchowski, “Column Lineage is Coming to the Rescue,” Berlin Buzzwords, June 18-20, 2023
        ii. Julien Le Dem and Willy Lulciuc, “Cross-platform Data Lineage with OpenLineage,” Data+AI Summit, June 28-29, 2023
        iii. Maciej Obuchowski, “OpenLineage in Airflow: A Comprehensive Guide,” Airflow Summit, September 19-21, 2023
2. Recent releases [Michael R.]:
    a. OpenLineage 0.25.0
        - Added
            - Spark: add Spark/Delta merge into support #1823 @pawel-big-lebowski
        - https://github.com/OpenLineage/OpenLineage/releases/tag/0.25.0
        - https://github.com/OpenLineage/OpenLineage/compare/0.24.0...0.25.0
    b. OpenLineage 0.26.0
        - Added
            - Proxy: Fluentd proxy support (experimental) #1757 @pawel-big-lebowski
        - Changed
            - Python client: use Hatchling over setuptools to orchestrate Python env setup #1856 @gaborbernat
        - https://github.com/OpenLineage/OpenLineage/releases/tag/0.26.0
        - https://github.com/OpenLineage/OpenLineage/compare/0.25.0...0.26.0
    c. OpenLineage 0.27.1
        - Added
            - Python client: add emission filtering mechanism and exact, regex filters #1878 @mobuchowski
        - https://github.com/OpenLineage/OpenLineage/releases/tag/0.27.1
        - https://github.com/OpenLineage/OpenLineage/compare/0.26.0...0.27.1
    d. OpenLineage 0.27.2
        - Fixed
            - Python client: deprecate client.from_environment, do not skip loading config #1908 @mobuchowski
        - https://github.com/OpenLineage/OpenLineage/releases/tag/0.27.2
        - https://github.com/OpenLineage/OpenLineage/compare/0.27.1...0.27.2
3. Static Lineage Progress Update [Pawe]:
    a. Overview
        i. Up to this point, operational/runtime metadata has been the focus of OpenLineage
        ii. But there is also a need for lineage metadata about datasets not associated with runs
        iii. To address this, a proposal has been created
            1. It answers the question: how can we add new data types to support static lineage?
            2. We decided to add two new types:
                a. job event
                b. dataset event
            3. A schemaURL provides a distinguishing mechanism
            4. Generic client code will not be affected
    b. Demo
        i. Approach taken: serialize and deserialize without modifying the database
    c. Conclusion
        i. This approach does not break existing usage scenarios while nonetheless adding new event types
        ii. Changes will be implemented in the clients and the spec
    d. Q&A
        i. Initial work on Marquez to support static lineage has also been completed (adding the capability to distinguish between the event types), but Marquez is not currently able to store static lineage metadata
        ii. Ability to convert from static to dynamic anticipated?
            1. Formats not very different
            2. Job event is subtype of a run event, making it easy to extract the data you care about
            3. Marquez UI should not change

iii. Ownership change notification possible?
1. This data accessible via the REST API but not currently built in
2. Contribution of such a feature would be welcome
3. Alternative solution: add a listener
iv. Job events are static but not dataset events?
1. Both are static events

4. Discussion items
   a. Marquez search – how robust?
      i. Recommended: visit the GitHub repo and use GitPod to try it out (or use the up.sh script in the docker directory there to deploy locally)
         1. Tags are accessible in some facets in the UI, which would provide one way
   b. Row-based lineage – are there any facets or models that would help with this use case?
      i. We are trying to keep the metadata store smaller than the data itself
      ii. Row-level lineage could be captured in a data model, which would be accessible in Marquez
      iii. Challenge: the volume of data
      iv. It might be helpful to have a doc about solutions for this in the project
   c. Another good forum for asking questions: https://bit.ly/OLslack

May 11, 2023 (10am PT)

Attendees:

- TSC:
  - Julien Le Dem, OpenLineage project lead
  - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage committer
  - Michael Robinson, Community team, Astronomer
  - Jakub Dardziski, Software Engineer, GetInData
- And:
  - Natalie Zeller, Software Engineer, Natural Intelligence
  - Cori Visi, Solutions Architect, AWS
  - Harel Shein, Engineering Manager, Astronomer
  - John Lukenoff, Software Engineer, Asana
  - Harshini Devathi, Data Engineer
  - Danilo Mota
  - Suparna Bhattacharya, HPE Labs
  - Ann Mary Justine, Research Engineer, HP Enterprise's CMF team
  - Ernie Ostic, SVP of Product, MANTA
  - Anirudh Shrinivason, Data Engineer, Grab

Agenda:

- Announcements
- Recent releases
- Custom transport types support
- dbt Cloud integration
- Discussion items
- Open discussion

Meeting:

Notes:

1. Announcements [Julien]:
   a. Upcoming meetups
      i. Boston Data Lineage Meetup (tentatively scheduled for June)
      ii. San Francisco OpenLineage Meetup at Astronomer (tentatively scheduled for June 27)
   b. Upcoming talks
      i. Pawe Leszczyski and Maciej Obuchowski, “Column Lineage is Coming to the Rescue,” Berlin Buzzwords, June 18-20, 2023
      ii. Julien Le Dem and Willy Lulciuc, “Cross-platform Data Lineage with OpenLineage,” Data+AI Summit, June 28-29, 2023
      iii. Maciej Obuchowski, “OpenLineage in Airflow: A Comprehensive Guide,” Airflow Summit, September 19-21, 2023
2. Recent releases [Michael R.]
   a. OpenLineage 0.24.0
      i. Additions
         1. Support custom transport types #1795 @nataliezeller1
         2. Airflow: dbt Cloud integration #1418 @howardyoo @JDarDagran
         3. Spark: support dataset name modification using regex #1796 @pawel-big-lebowski
      ii. https://github.com/OpenLineage/OpenLineage/releases/tag/0.24.0
      iii. https://github.com/OpenLineage/OpenLineage/compare/0.23.0...0.24.0
3. Custom transport types support [Natalie]
   a. OpenLineage supports a set of predefined transport types (HTTP, Kafka, others)
   b. Previously, adding a new or custom type required changing the transport config and transport factory to recognize the new type
   c. This change allows for extending functionality without having to change anything in the OpenLineage codebase
   d. Example: my company, where we work with an OpenMetadata backend
      i. This required a custom transport type
      ii. With this change I can do this without changing anything
   e. Implementation
      i. New interface: TransportBuilder

        ii. Implementable via methods:
            1. getType() // set in transport.type config param
            2. getConfig() // extension of TransportConfig, containing the required configuration
            3. Transport build(TransportConfig config) // builds a custom Transport instance based on the custom configuration
        iii. Additionally you need to have a file (META-INF/services/io.openlineage.client.transports.TransportBuilder) that must be included in a jar in the class path, containing the fully qualified name of the implementing class
        iv. Using the service loader pattern, implementations of TransportBuilder will be discovered and loaded at runtime.
    f. Q&A
        i. What are some use cases for other cool transport mechanisms?
            1. Native cloud, your queue system to send events
            2. Preferred way: the provider, data catalog, or something to implement over the lineage
            3. Maybe someone wants to do MSMQ or MQSeries
            4. You can also apply some transformation logic as part of your transport provider, so you can have your own ways of transporting the data
        ii. Should we have some sort of repository where people can put their custom transport types that their building in a single place?
            1. They can put them in the repo; I don't think we need a separate place, at least right now
4. dbt Cloud integration [Jakub]
    a. Previously:
        i. The dbt-ol script invoked dbt metadata processing and sent OpenLineage events
        ii. Worked only with a local dbt project
        iii. How events were created:
            1. each run was a separate supported dbt node
            2. parent run reflected dbt-ol command call
    b. New dbt Cloud integration:
        i. each run in dbt Cloud might have multiple steps, each producing separate JSON files
        ii. Each step is considered a parent run
        iii. DbtArtifactProcessor was separated as a parent for DbtCloudArtifactProcessor and DbtLocalArtifactProcessor classes; the naming convention stays the same
        iv. Used with *DbtCloudRunJobOperator* & *DbtCloudJobRunSensor* operators in Airflow integration, also makes use of *DbtCloudHook* to retrieve metadata from the dbt Cloud API
    c. Artifact retrieval and processing
        i. Due to a 10-sec thread timeout in the OpenLineage-Airflow integration, there is the following process for fetching dbt metadata:
            1. each run is a separate supported dbt node (models, tests, sources, snapshots)
            2. parent run reflects *dbt-ol* command call
        ii. The issue will be resolved with the Airflow OpenLineage provider release (learn more about AIP-53 here)
5. Discussion items
    a. Can we help ensure efficiency by narrowing the scope in some pragmatic ways? For example: is validation necessary in the case that an OpenLineage client is being used to send events? Are there other similar cases where validation might not be necessary?
        i. Work on adding validation to the project is ongoing, e.g., in the proxy where there is some schema validation happening
        ii. It would be useful to have some testing facility, e.g., for people consuming OpenLineage and potential implementers
        iii. From a producer's point of view, we could check if the consumer consumes them; this would have to be specific to each consumer
        iv. We could have a dataset of events that contain all the assets, which would be useful for anyone who wants to do their own testing – like examples of all the facets that exist (instead of having to create them by hand for internal teams)
        v. Maybe just pump demo payloads out to disk and keep them somewhere
    b. Improving column lineage: there are lots of other elements that would be useful
        i. People want to add selected rules and filters
            1. Is there an anticipated traffic level, typical volume in a plan for design lineage
        ii. Column metadata is well covered by other standards in the industry, but there are some lineage ones related to expected performance, flags that people want such as for PII data that's being managed on that edge, etc.
        iii. One question: are those properties of a transformation itself, or just a property of a resulting column?
            1. In some cases, transformation; in others the actual edge, which is interesting. Option: have the ability to define the kinds of edges
            2. for PII, there is a tagging facet we were discussing that is still in progress
            3. Action item: get feedback on this and complete it
    c. Spark integration: merge into and aggregate functions don't provide column lineage
        i. A fix has recently been made, but when will this be released?
        ii. Anyone can request a release in the #general Slack channel. You're encouraged to do this if you'd like a fix before the next regularly scheduled release (on the first work day of the month).

April 20, 2023 (10am PT)

Attendees:

- TSC:
    - Julien Le Dem, OpenLineage project lead
    - Pawe Leszczyski, Software Engineer, GetInData
    - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage committer
    - Michael Robinson, Community team, Astronomer
- And:
    - Sheeri Cabral, Technical Product Manager, Lineage, Collibra
    - Julian LaNeve, Senior Product Manager, Astronomer
    - John Montroy, Big data/backend engineer
    - Anirudh Shrinivason, Data Engineer, Grab

Agenda:

1. Announcements
2. Updates (new!)

    a. OpenLineage in Airflow AIP
    b. Static lineage support
3. Recent release overview
4. A new consumer
5. Caching support for column lineage
6. Discussion items
    a. Snowflake tagging
7. Open discussion

Meeting:

Notes:

1. Announcements [Julien]
    a. A New York meetup will be happening on 4/26 at the Astronomer offices in the Flatiron District
    b. **Julien Le Dem** will be speaking at the Data+AI Summit in June: "Cross-platform Data Lineage with OpenLineage"
    c. Recent talks:
        i. Last month: **Ross Turk**, **Pawe Leszczyski** and **Maciej Obuchowski** all spoke at Big Data Technology Warsaw Summit 2023
        ii. Also last month: Julien spoke at Data Council Austin
    d. Recent meetups:
        i. Last month: OpenLineage Meetup at Data Council Austin
        ii. Last month: Data Lineage Meetup in Providence, RI
2. Updates [Julien]
    a. OpenLineage in Airflow (AIP-53)
        i. Goal: make operators responsible for their own lineage
        ii. Goal requires additions to the Airflow infrastructure
        iii. Development process will progress in 3 phases
            1. add an OpenLineage library conforming to Airflow processes and coding style
            2. work on other providers, implementing OpenLineage methods
            3. add OpenLineage support to TaskFlow and Python operators
        iv. Timeline: aiming for June Providers release
        v. We have begun with the Snowflake operator
        vi. A significant benefit: operators will support it
    b. Static lineage support
        i. Next stage: add formal proposal to the OpenLineage repo, where it will be easier for members to comment
        ii. To recap:
            1. OL is designed to capture lineage as pipelines run, as well as some info that is more static (schema, schema changes, etc.)
            2. Goal: capture lineage about views, etc., that have not run yet
            3. Focus will remain on everything that has been deployed
            4. Parallel discussion: lineage from job-less events, e.g., ad-hoc events
                a. challenge: these could pollute the namespace
            5. Basic proposal: to make the job name optional, which will require changes on the Marquez side, as well
        iii. Comments are welcome
            1. See the #general channel in Slack for links to the two relevant docs
3. Caching support for column lineage [Pawe]
    a. Personal opinion: the Spark integration is amazing because it extracts from the logical plan; also, it is easy to configure (requiring just 4 lines of code)
    b. Caching: a popular concept for Spark jobs
        i. a separate logical plan is used for cached datasets, meaning that two logical plans must be merged
        ii. we will know how inputs are affecting outputs even when logical plans have been merged
4. Open discussion
    a. A question about duplicated events when setting env variables [Anirudh]
        i. we have needed to employ filtering
        ii. Spark reuses jobs for actions that are not really jobs

March 9, 2023 (10am PT)

Attendees:

- **TSC:**
    - Julien Le Dem, OpenLineage project lead
    - Minkyu Park, Senior Engineer, Astronomer
    - Michael Collado, Staff Engineer, Astronomer
    - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage committer
    - Willy Lulciuc, Co-creator of Marquez, OpenLineage committer
    - Michael Robinson, Community team, Astronomer
    - Jakub Dardziski, Software Engineer, GetInData
    - Tomasz Nazarewicz, Software Engineer, GetInData
- **And:**
    - Sam Holmberg, Senior Software Engineer, Astronomer
    - Brad, Fivetran
    - Prachi Mishra, Senior Software Engineer, Astronomer
    - Sheeri Cabral, Project Manager, Collibra
    - Anirudh Shrinivason, Data Engineer, Grab
    - Ann Mary Justine, Research Engineer, HP Enterprise's CMF team
    - John Thomas, Software Engineer, Dev. Rel., Astronomer

    - Atif Tahir, Data Engineer, Astronomer
    - Martin Foltin, Data Engineer, HP Enterprise's CMF team

Agenda:

- Recent releases
- Demo: custom env variable support in the Spark integration
- Async operator support in Airflow
- JDBC relations support in Spark
- Discussion topics:
    - new feature idea: column transformations/operations in the Spark integration
    - the thinking behind namespaces
- Open discussion

Meeting:

Slides:

Notes:

- Announcements [Julien]
    - Two meetups will be happening soon:
        - Data Lineage Meetup cohosted with Collibra, Providence, RI, March 9 at 6 PM ET
        - OpenLineage Meetup at Data Council Austin on March 30th at 12:15 PM CST
    - Talk happening soon:
        - Julien Le Dem, "Ten Years of Building Open Source Standards: From Parquet to Arrow to OpenLineage," Data Council Austin, March 30th, 10 AM CST
- Recent releases 0.20.6, 0.21.1
    - 0.20.6
        - **Added**
            - Airflow: add new extractor for FTPFileTransmitOperator #1603 @sekikn
        - **Changed**
            - Airflow: make extractors for async operators work #1601 @JDarDagran
    - 0.21.1
        - **Added**
            - Clients: add DEBUG logging of events to transports #1633 by @mobuchowski
            - Spark: add CustomEnvironmentFacetBuilder class #1545 by ***New contributor*** @Anirudh181001
            - Spark: introduce the new output visitors AlterTableAddPartitionCommandVisitor and AlterTableSetLocationCommandVisitor #1629 by ***New contributor*** @nataliezeller1
            - Spark: add column lineage for JDBC relations #1636 by @tnazarew
            - SQL: add Linux-aarch64 native library to Java SQL parser #1664 by @mobuchowski
        - **Changed**
            - Airflow: get table database in Athena extractor #1631 by ***New contributor*** @rinzool
        - **Fixed**
            - dbt: add dbt seed to the list of dbt-ol events #1649 by ***New contributor*** @pohek321
    - Thanks to all our contributors!
    - More details:
        - https://github.com/OpenLineage/OpenLineage/releases
        - https://github.com/OpenLineage/OpenLineage/blob/0.21.1/CHANGELOG.md
- Custom env var support in the Spark integration [Anirudh]
    - adds ability to capture environment variables from a Spark cluster
    - required the addition of a new class to extend an existing class
    - does not override variables already being captured
    - desired variables must be specified by the user
    - variables are visible in environment properties of OpenLineage events
    - Q & A
        - Q: is it possible to accidentally include sensitive data in these variables?
        - A: users must "opt in" by selecting variables in advance
        - Q: what was the experience like interacting with the community?
        - A: really great! I got a lot of help from a lot of people, including Pawel

February 9, 2023 (10am PT)

Attendees:

- **TSC:**
    - Julien Le Dem, OpenLineage project lead
    - Ross Turk, Senior Director of Community, Astronomer
    - Benji Lampel, Product Manager, Astronomer
    - Minkyu Park, Senior Engineer, Astronomer
    - Michael Collado, Staff Engineer, Astronomer
    - Howard Yoo, Staff Product Manager, Astronomer
    - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage contributor
    - Willy Lulciuc, Co-creator of Marquez
    - Danny Henneberger, OpenLineage committer

    - Michael Robinson, Developer Relations Engineer, Astronomer
- **And:**
    - Prachi Mishra, Senior Software Engineer, Astronomer
    - Sheeri Cabral, Project Manager, Collibra
    - Enrico Rotundo, Bacalhau Project
    - Brad, Fivetran
    - Harel Shein, Director of Engineering, Astronomer
    - Robert Karish, Data Engineer, AdTheorent
    - Eric Veleker, Atlan
    - Ben Sandler
    - Peter Hicks, Senior Software Engineer, Astronomer
    - John Thomas, Software Engineer, Developer Relations, Astronomer
    - Nikhil Wadhwa, Engineer, Fivetran
    - Sam Holmberg, Senior Software Engineer, Astronomer
    - David
    - Matthew Krubski

Agenda:

- Recent releases
- AIP: OpenLineage in Airflow
- Discussion topic: real-world implementation of OpenLineage (i.e., "What IS lineage, anyway?")
- Announcement & discussion topic: the thinking behind namespaces
- Open discussion

Meeting:

Notes:

- Announcements [Julien]
    - The first Data Lineage Meetup will be taking place in Providence on March 9th at 6 pm. More information: https://openlineage.io/blog/data-lineage-meetup/
- Recent release 0.20.4 [Michael R.]
    - Added
        - Airflow: add new extractor for GCSToGCSOperator #1495 @sekikn
          Adds a new extractor for this operator.
        - Flink: resolve topic names from regex, support 1.16.0 #1522 @pawel-big-lebowski
          Adds support for Flink 1.16.0 and makes the integration resolve topic names from Kafka topic patterns.
        - Proxy: implement lineage event validator for client proxy #1469 @fm100
          Implements logic in the proxy (which is still in development) for validating and handling lineage events.

      Changed
        - CI: use ruff instead of flake8, isort, etc., for linting and formatting #1526 @mobuchowski
          Adopts the ruff package, which combines several linters and formatters into one fast binary.
    - Thanks to all our contributors!
    - More details: https://github.com/OpenLineage/OpenLineage/blob/main/CHANGELOG.md
- AIP: OpenLineage in Airflow [Julien]
    - Motivations
        - Key goal of project: provide a central spec everyone case use for lineage
        - Ultimate goal for integrations: house them in their home projects, not OpenLineage
        - Specific challenge of separate, locally hosted integrations: changes to Airflow have broken the integration
        - First-class, built-in support would mean more stability and less effort
    - Two-fold proposal
        - turn the integration OpenLineage-Airflow package into an Airflow provider
        - the lineage extraction logic will live in the operators themselves, not in separate extractors
    - Benefits
        - increased stability
        - easier maintenance over time
    - Downside
        - burden of maintenance shifts to Airflow community
        - but this is logical, and the Airflow community will grow as a result
    - More information:
    - Next step: to hold a vote on the Airflow mailing list
    - Q & A:
        - Maciej: Jakub and I will be there to help in the Airflow community
        - Julien: I agree, and contributors will likely become Airflow committers
        - Enrico: if you were to write a provider today, would you start externally or in Airflow?
        - Julien: I would start externally and iterate, then submit for provider status
        - Julien: Ross, is the current posture in Airflow to expect provider codebase owners to maintain their code in separate repositories?
        - Ross: yes, due to ease of maintenance when APIs change, etc.
- Discussion topic: real-world implementation of OpenLineage (i.e., "What IS lineage, anyway?") [Sheeri]
    - Ross: opened an issue about creating a validation suite
        - ideas: make Marquez into a validation suite, use the seed data
    - Sheeri: minimum coverage: nodes and transformations
        - what do you think?

    - Brad: best practices for clean extractions but allow for extensibility (e.g., external extractors)
        - we plan to use all the core elements (datasets, runs, jobs, etc.)
    - John: two pieces are involved: validating emitted events and assessing compliance of facets
        - also: naming conventions are becoming unwieldy
    - Maciej: we have been experimenting with providing different facets – custom facets are not a bad thing, and not everything belongs in the core spec
    - Julien: custom facets are intended for specific requirements not supported by the core spec
        - we need to balance between centralization, where everything must be approved, and chaos, where nothing is – it's a trade-off
    - Sheeri: would everyone be willing to write down their custom facets somewhere?
    - Julien: we need a place where core and custom facets are all defined – maybe we should work from a Google doc or a PR
    - Eric: there is a lot of opportunity to discover custom facets
        - setting up an incentive structure to create/share custom facets would be valuable
    - Julien: there is a mechanism for discovering custom facets
        - a list of all the existing custom facets is available at runtime
        - a registration process might be useful for static discovery
    - See the Slack channel that is available for continuing this discussion: #spec-compliance

January 12, 2023 (10am PT)

Attendees:

- TSC:
    - Mike Collado, Staff Software Engineer, Astronomer
    - Julien Le Dem, OpenLineage Project lead
    - Willy Lulciuc, Co-creator of Marquez
    - Michael Robinson, Software Engineer, Dev. Rel., Astronomer
    - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage contributor
    - Mandy Chessell, Egeria Project Lead
    - Daniel Henneberger, Database engineer
    - Will Johnson, Senior Cloud Solution Architect, Azure Cloud, Microsoft
    - Jakub "Kuba" Dardziski, Software Engineer, GetInData, OpenLineage contributor
- And:
    - Petr Hajek, Information Management Professional, Profinit
    - Harel Shein, Director of Engineering, Astronomer
    - Minkyu Park, Senior Software Engineer, Astronomer
    - Sam Holmberg, Software Engineer, Astronomer
    - Ernie Ostic, SVP of Product, MANTA
    - Sheeri Cabral, Technical Product Manager, Lineage, Collibra
    - John Thomas, Software Engineer, Dev. Rel., Astronomer
    - Bramha Aelem, BigData/Cloud/ML and AI Architect, Tiger Analytics

Agenda:

- Announcements
- Recent release 0.19.2
- Update on column-level lineage
- Overview of recent improvements to the Airflow integration
- Discussion topic: real-world implementation of OpenLineage (i.e., "What IS lineage, anyway?")
- Announcement & discussion topic: the thinking behind namespaces

Meeting:

Notes:

- Announcements
    - OpenLineage earned Incubation status with the LFAI & Data Foundation at their December TAC meeting!
        - Represents our maturation in terms of governance, code quality assurance practices, documentation, more
        - Required earning the OpenSSF Silver Badge, sponsorship, at least 300 GitHub stars
        - Next up: Graduation (expected in early summer)
- Recent release 0.19.2 [Michael R.]
    - Added
        - SQL: add column-level lineage to SQL parser #1432 #1461 @mobuchowski @StarostaGit
        - SQL: add ExtractionErrorRunFacet #1442 @mobuchowski
        - Airflow: add Trino extractor #1288 @sekikn
        - Airflow: add S3FileTransformOperator extractor #1450 @sekikn
        - Airflow: add standardized run facet #1413 @JDarDagran
        - Airflow: add NominalTimeRunFacet and OwnershipJobFacet #1410 @JDarDagran
        - dbt: add support for postgres datasources #1417 @julienledem
        - Proxy: add client-side proxy (skeletal version) #1439 #1420 @fm100
        - Proxy: add CI job to publish Docker image #1086 @wslulciuc
        - Spark: pass config parameters to the OL client #1383 @tnazarew

      Fixed

        - Airflow: fix collect_ignore, add flags to Pytest for cleaner output #1437 @JDarDagran
        - Spark & Java client: fix README typos @versaurabh
    - Thanks to all the contributors, including new contributor @versaurabh!

    - More details: https://github.com/OpenLineage/OpenLineage/blob/main/CHANGELOG.md
- Column-level lineage update [Maciej]
    - What is the OpenLineage SQL parser?
        - At its core, it's a Rust library that parses SQL statements and extracts lineage data from it
        - 80/20 solution - we'll not be able to parse all possible SQL statements - each database has custom extensions and different syntax, so we focus on standard SQL.
        - Good example of complicated extension: Snowflake COPY INTO https://docs.snowflake.com/en/sql-reference/sql/copy-into-table.html
        - We primarily use the parser in Airflow integration and Great Expectations integration
        - Why? Airflow does not "understand" a lot of what some operators do, for example PostgreSqlOperator
        - We also have Java support package for parser
    - What changed previously?
        - Parser in current release can emit column-level lineage!
        - Last OL meeting Piotr Wojtczak, primary author of this change presented new core of parser that enabled that functionality https://www.youtube.com/watch?v=Lv_bODeAVYQ
        - Still, the fact that Rust code can do that does not mean we have it for free everywhere
    - What has changed recently?
        - We wrote "glue code" that allows us to use new parser constructs in Airflow integration
        - Error handling just got way easier: SQL parser can "partially" parse SQL construct, and report errors it encountered, with particular statements that caused it.
    - Usage
        - Airflow integration extractors based on SqlExtractor (ex. PostgreSqlExtractor, SnowflakeExtractor, TrinoExtractor…) are now able to extract column-level lineage
        - Close future: Spark will be able to extract lineage from JDBCRelation.
- Recent improvements to the Airflow integration [Kuba]
    - OpenLineage facets
        - Facets are pieces of metadata that can be attached to the core entities: run, job or dataset
        - Facets provide context to OpenLineage events
        - They can be defined as either part of the OpenLineage spec or custom facets
    - Airflow generic facet
        - Previously multiple custom facets with no standard
            - *AirflowVersionRunFacet* as an example of rapidly growing facet with version unrelated information
        - Introduced *AirflowRunFacet* with Task, DAG, TaskInstance and DagRun properties
        - Old facets are going to be deprecated soon. Currently both old and new facets are emitted
            - *AirflowRunArgsRunFacet*, *AirflowVersionRunFacet*, *AirflowMappedTaskRunFacet* will be removed
            - All information from above is moved to *AirflowRunFacet*
    - Other improvements (added in 0.19.2)
        - SQL extractors now send column-level lineage metadata
        - Further facets standardization
            - Introduced *ProcessingEngineRunFacet*
                - provides processing engine information, e.g. Airflow or Spark version
            - Improved support for nominal start & end times
                - makes use of data interval (introduced in Airflow 2.x)
                - nominal end time now matches next schedule time
            - DAG owner added to *OwnershipJobFacet*
            - Added support for S3FileTransformOperator and TrinoOperator (@sekikn's great contribution)
- Discussion: what does it mean to implement the spec? [Sheeri]
    - What is it mean to meet the spec?
        - 100% compliance is not required
        - OL ecosystem page
            - doesn't say what exactly it does
            - operational lineage not well defined
            - what does a payload look like? hard to find this info
        - Compatibility between producers/consumers is unclear
    - Important if standard is to be adopted widely [Mandy]
        - Egeria: uses compliance test with reports and badging; clarifies compatibility
        - test and test cases available in the Egeria repo, including profiles and clear rules about compliant ways to support Egeria
        - a badly behaving producer or consumer will create problems
        - have to be able to trust what you get
    - What about consumers? [Mike C.]
        - can we determine if they have done the correct thing with facets? [John]
        - what do we call "compliant"?
        - custom facets shouldn't be subject to this – they are by definition custom (and private) [Maciej]
        - only complete events (not start events) should be required – start events not desired outside of operational use cases [Maciej]
    - There's a simple baseline on the one hand and facets on the other [Julien]
    - Note: perfection isn't the goal
        - instead: shared test cases, data such as sample schema that can be tested against
    - Marquez doesn't explain which facets it's using or how [Willy]
        - communication by consumers could be better
    - Effort at documenting this: matrix [Julien]
    - How would we define failing tests? [Maciej]
        - at a minimum we could have a validation mode [Julien]
        - challenge: the spec is always moving, growing [Maciej]
        - ex: in the case of JSON schema validation, facets are versioned individually but there's a reference schema that is versioned that might not be the current schema. Facets can be dereferenced, but the right way to do this is not clear [Danny]
        - one solution could be to split out base times, or we could add a tool that would force us to clean this up
        - client-side proxy presents same problem; tried different validators in Go; a workaround is to validate against the main doc first; by continually validating against the client proxy we can make sure it stays compliant with the spec [Minkyu]
        - Mandy: if Marquez says it's "OK," it's OK; we've been doing it manually [Mandy]

        - Marquez doesn't do any validation for consumers [Mike C.]
        - manual validation is not good enough [Mandy]
        - I like the idea of compliance badges – it would be cool if we had a way to validate consumers and there were a way to prove this **and** if we could extend validation to integrations like the Airflow integration [Mike C.]
    - Let's follow up on Slack and use the notes from this discussion to collaborate on a proposal [Julien]

December 8, 2022 (10am PT)

Attendees:

- TSC:
    - Mike Collado, Staff Software Engineer, Astronomer
    - Julien Le Dem, OpenLineage Project lead
    - Willy Lulciuc, Co-creator of Marquez
    - Michael Robinson, Software Engineer, Dev. Rel., Astronomer
    - Howard Yoo, Staff Product Manager, Astronomer
    - Ross Turk, Senior Director of Community, Astronomer
- And:
    - Enrico Rotundo, Data Scientist, Winder.AI
    - Petr Hajek, Information Management Professional, Profinit
    - Sheeri Cabral, Technical Product Manager, Lineage, Collibra
    - Ernie Ostic, SVP of Product, MANTA
    - Piotr Wojtczak, Software Engineer, GetInData
    - Minkyu Park, Senior Software Engineer, Astronomer
    - Prachi Mishra, Senior Software Engineer, Astronomer
    - Ann Mary Justine, Research Engineer, HP Enterprise
    - John Thomas, Software Engineer, Dev. Rel., Astronomer
    - Benji Lampel, Ecosystem Engineer, Astronomer
    - Henoc Mukadi, Data Engineer, Prodigy Finance
    - Brahma Aelem, BigData/Cloud/ML and AI Architect, Tiger Analytics

Agenda:

- Announcements
- Recent releases
- The new Rust implementation of the SQL integration (15 min.)
- Presentation and discussion: the meaning of "implementing" the spec (35 min.)
- Open discussion

Meeting:

Notes:

- Recent releases [Michael R.]
    - 0.18.0
        - Added
            - Airflow: support `SQLExecuteQueryOperator` #1379 @JDarDagran
            - Airflow: introduce a new extractor for `SFTPOperator` #1263 @sekikn
            - Airflow: add Sagemaker extractors #1136 @fhoda
            - Airflow: add S3 extractor for Airflow operators #1166 @fhoda
            - Spec: add spec file for `ExternalQueryRunFacet` #1262 @howardyoo
            - Docs: add a TSC doc #1303 @merobi-hub

        Bug fixes and more details: https://github.com/OpenLineage/OpenLineage/blob/main/CHANGELOG.md
    - 0.17.0
        - Added
            - Spark: support latest Spark 3.3.1 #1183 @pawel-big-lebowski
            - Spark: add Kinesis Transport and support config Kinesis in Spark integration #1200 @yogyang
            - Spark: disable specified facets #1271 @pawel-big-lebowski
            - Python: add facets implementation to Python client #1233 @pawel-big-lebowski
            - SQL: add Rust parser interface #1172 @StarostaGit @mobuchowski
            - Proxy: add helm chart for the proxy backed #1068 @wslulciuc
            - Spec: include possible facets usage in spec #1249 @pawel-big-lebowski
            - Website: publish YML version of spec to website #1300 @rossturk
            - Docs: update language on nominating new committers #1270 @rossturk

        Changed
            - Website: publish spec into new website repo location #1295 @rossturk
            - Airflow: change how pip installs packages in tox environments #1302 @JDarDagran

        Bug fixes and more details: https://github.com/OpenLineage/OpenLineage/blob/main/CHANGELOG.md
- Rust implementation of the SQL integration [Piotr]
    - About me: dev with GetInData

    - Goal of project: to make adding more language support in the future easier
    - Separated into components: separate backend package for integration with language bindings with new Java interface
    - Components
        - `openlineage_sql`: main implementation with table + column lineage extraction
        - `openlineage_sql_python`: Python bindings, uses the `pyo3` create, produces a Python wheel
        - `openlineage_sql_java`: Java bindings, using JNI, produces a jar
    - Changes
        - switch to a visitor pattern to traverse the AST
        - introduce Context Frames (like scopes) to resolve aliases, implicit contexts and shadowing
        - column lineage is a synthesized attribute over the tree – easy to compute with a visitor
    - Demo
    - Shout outs
        - Maciej Obuchowski (@mobuchowski)
        - Will Johnson (@wjohnson)
        - Hannah Moazam (@hmoazam)
- Open discussion
    - Spark implementation: where do deps need to be added? [Will]
        - it depends on which sub-project you want to modify
        - if you want to modify all, import the dependency in `shared`
    - Implementing the spec discussion [Sheeri]
        - 100% compliance is not required – it's a spec, after all, just like "standard" SQL
        - bottom line: compatibility between producers and consumers
        - minimum viable lineage
            - at least one circle
            - zero or more lines
            - associated information
        - data model: event runs a job on a dataset
        - What's required by the spec?
            - run: UUID
            - run state: transition, event time
            - job: namespace, job name
            - datasets: namespace, dataset name
        - But what is a run?
            - all the events for one UUID
        - Necessary per run:
            - at least one box
            - at least one line
            - everything else is optional
                - eventTime, etc.
        - OL query example:
            - run ID required for a run (but not a job, which can/should be a view)
            - inputs
            - outputs
            - producer
            - schemaURL
            - start event
            - complete event
        - Needed: discussion of what it means to be compliant with the spec, perhaps a test/self-test
            - maybe the test outputs categories (e.g., "design lineage") for compatibility between producers and consumers
        - Following up on main threads here [Julien]:
            - create Slack channel, Google docs
                - Sheeri will take the lead
                - we'll write a proposal that we eventually add to the spec

November 10, 2022 (10am PT)

Attendees:

- TSC:
    - Mike Collado, Staff Software Engineer, Astronomer
    - Julien Le Dem, OpenLineage Project lead
    - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage contributor
    - Mandy Chessell, Egeria Project Lead
    - Willy Lulciuc, Co-creator of Marquez
    - Pawe Leszczyski, Software Engineer, GetInData
    - Ross Turk, Senior Director of Community, Astronomer
    - Howard Yoo, Staff Product Manager, Astronomer
    - Tomasz Nazarewicz, Software Engineer, GetInData
    - Michael Robinson, Software Engineer, Dev. Rel., Astronomer
- And:
    - Ann Mary Justine, Research Engineer, HP Enterprise
    - Martin Foltin, Master Technologist, HP Enterprise
    - Sam Holmberg, Software Engineer, Astronomer
    - Aalap Tripathy, Principal Research Engineer, HP Enterprise
    - Petr Hajek, Information Management Professional, Profinit
    - Harel Shein, Director of Engineering, Astronomer
    - Minkyu Park, Senior Software Engineer, Astronomer
    - Benji Lampel, Ecosystem Engineer, Astronomer

- Suparna Bhattacharya, Distinguished Technologist, HP Enterprise
- John Thomas, Software Engineer, Dev. Rel., Astronomer
- Sergey Serebryakov, Research Engineer, HP Enterprise
- Glyn Bowden, Chief Technologist, HP Enterprise, CMF
- Nigel Jones, Maintainer, Egeria/IBM
- Tomasz Nazarewicz, Software Engineer, GetInData
- Sheeri Cabral, Technical Product Manager, Lineage, Collibra
- Prachi Mishra, Senior Software Engineer, Astronomer

Agenda:

- Recent release overview
- Update on LFAI & Data Foundation progress
- Implementing OpenLineage proposal and discussion
- Update from MANTA
- Linking CMF (a common ML metadata framework) and OpenLineage
- Open discussion

Meeting:

Notes:

- Announcements [Julien]
  - OpenLineage earned the OSSF Core Infrastructure Silver Badge!
  - Happening soon: OpenLineage to apply formally for Incubation status with the LFAI
  - Blog: a post by Ernie Ostic about MANTA's OpenLineage integration
  - Website: a new Ecosystem page
  - Workshops repo: An Intro to Dataset Lineage with Jupyter and Spark
  - Airflow docs: guidance on creating custom extractors to support external operators
  - Spark docs: improved documentation of column lineage facets and extensions
- Recent release 0.16.1 [Michael R.]
  - Added
    - Airflow: add dag_run information to Airflow version run facet #1133 @fm100
      Adds the Airflow DAG run ID to the taskInfo facet, making this additional information available to the integration.
    - Airflow: add LoggingMixin to extractors #1149 @JDarDagran
      Adds a LoggingMixin class to the custom extractor to make the output consistent with general Airflow and OpenLineage logging settings.
    - Airflow: add default extractor #1162 @mobuchowski
      Adds a DefaultExtractor to support the default implementation of OpenLineage for external operators without the need for custom extractors.
    - Airflow: add on_complete argument in DefaultExtractor #1188 @JDarDagran
      Adds support for running another method on extract_on_complete.
    - SQL: reorganize the library into multiple packages #1167 @StarostaGit @mobuchowski
      Splits the SQL library into a Rust implementation and foreign language bindings, easing the process of adding language interfaces. Also contains a CI fix.

    Changed

    - Airflow: move get_connection_uri as extractor's classmethod #1169 @JDarDagran
      The get_connection_uri method allowed for too many params, resulting in unnecessarily long URIs. This changes the logic to whitelisting per extractor.
    - Airflow: change get_openlineage_facets_on_start/complete behavior #1201 @JDarDagran
      Splits up the method for greater legibility and easier maintenance.
  - Removed
    - Airflow: remove support for Airflow 1.10 #1128 @mobuchowski
      Removes the code structures and tests enabling support for Airflow 1.10.
  - Bug fixes and more details
    - https://github.com/OpenLineage/OpenLineage/blob/main/CHANGELOG.md
- Update on LFAI & Data progress [Michael R.]
  - LFAI & Data: *a single funding effort to support technical projects hosted under the [Linux] foundation*
  - Current status: applying soon for Incubation, will be ready to apply for Graduation soon (dates TBD).
  - **Incubation** stage requirements:

2+ organizations actively contributing to the project	**23 organizations**
A sponsor who is an existing LFAI & Data member	**To do**
300+ stars on GitHub	**1.1K GitHub stars**
A Core Infrastructure Initiative Best Practices Silver Badge	**Silver Badge earned on November 2**
Affirmative vote of the TAC and Governing Board	**Pending**
A defined TSC with a chairperson	**TSC with chairperson: Julien Le Dem**

Graduation stage requirements:

5+ organizations actively contributing to the project	**23 organizations**
Substantial flow of commits for 12 months	**Commit growth rate (12 mo.): 155.53%** **Avg commits pushed by active contributors (12 mo.): 2.18K**
1000+ stars on GitHub	**1.1K GitHub stars**
Core Infrastructure Initiative Best Practices Gold Badge	**Gold Badge in progress (57%)**
Affirmative vote of the TAC and Governing Board	**Pending**
1+ collaboration with another LFAI project	**Marquez, Egeria, Amundsen**
Technical lead appointed on the TAC	**To do**

- Implementing OpenLineage proposal and discussion [Julien]
  - Procedure for implementing OpenLineage is under-documented
  - Goal: provide a better guide on the multiple approaches that exist
  - Contributions are welcome
  - Expect more information about this at the next meeting
- MANTA integration update [Petr]
  - Project: MANTA OpenLineage Connector
  - Straightforward solution:
    - Agent installed on customer side to setup an API endpoint for MANTA
    - MANTA Agent will hand over OpenLineage events to the MANTA OpenLineage Extractor, which will save the data in a MANTA OpenLineage Event Repository
    - Use the MANTA Admin UI to run/schedule the MANTA OpenLineage Reader to generator an OpenLineage Graph and produce the final MANTA Graph using a MANTA OpenLineage Generator
    - The whole process will be parameterized
  - Demo:
    - Example dataset produced by Keboola integration
    - All dependencies visualized in UI
    - Some information about columns is available, but not true column lineage
    - Possible to draw lineage across range of tools
  - Looking for volunteers willing to test the integration
  - Q&A
    - Are you using the Column-level Lineage Facet from OpenLineage?
      - Not yet, but we would like to test it
      - Find a good example of this in the OpenLineage/workshops/Spark GitHub repo
      - What would be great would be a real example/real environment for testing
- Linking CMF (a common ML metadata framework) and OpenLineage [Suparna & Ann Mary]
  - https://github.com/HewlettPackard/cmf
  - Where CMF will fit in the OpenLineage ecosystem
    - linkage needed between forms of metadata for conducting AI experiments
    - concept: "git for AI metadata" consumable by tools such as Marquez and Egeria after publication by an OpenLineage-CMF publisher
    - challenges:
      - multiple stages with interlinked dependencies
      - executing asynchronously
      - data centricity requires artifact lineage and tracking influence of different artifacts and data slices on model performance
      - pipelines should be Reproducible, Auditable and Traceable
      - end-to-end visibility is necessary to identify biases, etc.
    - AI for Science example:
      - training loop in complex pipeline with multiple models optimized concurrently
        - e.g., an embedding model, edge selection model and graph neural model in same pipeline
        - CMF used to capture metadata across pipeline stages
    - Manufacturing quality monitoring pipeline
      - iterative retraining with new samples added to the dataset every iteration
      - CMF tracks lineage across training and deployment stages
      - Q: is the recording of metadata automatic, or does the data scientist have control over it?
        - there both explicit (e.g., APIs) and implicit modes of tracking
        - the data scientist can choose which "branches" to "push" a la Git
    - 3 columns of reproducibility
      - metadata store (MLMD/MLFlow)
      - Artifact Store (DVC/Others)
      - Query Cache Layer (Graph Database)
      - GIT
      - optimization
    - Comparison with other AI metadata infrastructure
      - Git-like support and ability to collaborate across teams distinguish CMF from alternatives
      - Metrics and lineage also make CMF comparable to model-centric and pipeline-centric tools
    - Lineage tracking and decentralized usage model
      - complete view of data model lineage for reproducibility, optimization, explainability
      - decentralized usage model, easily cloned in any environment
    - What does it look like?
      - explicit tracking via Python library
      - tracking of dataset, model and metrics
      - offers end-to-end visibility

        - API
            - abstractions: pipeline state, context/stage of execution, execution
        - Automated logging, heterogeneous SQ stand distributed teams
            - enables collaboration of distributed teams of scientists using a diverse set of libraries
            - automatic logging in command line interface
        - POC implementations
            - allows for integration with existing frameworks
            - compatible with ML/DL frameworks and ML tracking platforms
        - Translation between CMF and OpenLineage
            - export of metadata in OpenLineage format
            - mapping of abstractions onto OpenLineage
            - Run ~ Execution with Run facet
            - Job ~ Context with Job facet
            - Dataset ~ Dataset with Dataset facet
            - Namespace ~ Pipeline
        - Q&A
            - Pipeline might map to Job name
            - Context might map to Pipeline as Parent job
            - Model could map to a Dataset as well as Dataset
            - Metric as a model could map to a Dataset facet
            - 2 levels of dataset facet, one static and one tied to Job Runs

October 13, 2022 (10am PT)

Attendees:

- TSC:
    - Mike Collado, Staff Software Engineer, Astronomer
    - Julien Le Dem, OpenLineage Project lead
    - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage contributor
- And:
    - Petr Hajek, Software Engineer, MANTA
    - Harel Shein, Director of Engineering, Astronomer
    - Minkyu Park, Senior Software Engineer, Astronomer
    - Michael Robinson, Software Engineer, Dev. Rel., Astronomer
    - Howard Yoo, Staff Product Manager, Astronomer
    - Tomasz Nazarewicz, Software Engineer, GetInData
    - Sheeri Cabral: Technical Product Manager, Lineage, Collibra
    - Hanna Moazam, Cloud Solution Architect, Microsoft

Agenda:

- Recent release 0.15.1
- Project roadmap review
- Column-level lineage workshop using Jupyter + Spark

Meeting:

Notes:

- Announcements:
    - We recently removed support for Airflow 1.x
    - Ross gave a talk on OpenLineage at ApacheCon in New Orleans last week
    - Upcoming opportunities to give talks about OpenLineage:
        - Data Teams Summit (January 2023)
        - Subsurface Live (January 2023)
        - Data Council Austin (March 2023)
    - Giving a talk on data lineage soon? Ping Michael R. on Slack to let us know.
- Recent release 0.15.1 [Michael R.]
    - Added
        - Airflow: improve development experience #1101 @JDarDagran
        - Documentation: update issue templates for proposal & add new integration template #1116 @rossturk
        - Spark: add description for URL parameters in readme, change overwriteName to appName #1130 @tnazarew

      Changed
        - Airflow: lazy load BigQuery client #1119 @mobuchowski

      Fixed
        - Spark: fix column lineage #1069 @pawel-big-lebowski
        - Spark: set log level of Init OpenLineageContext to DEBUG #1064 **new contributor @varuntestaz**
        - Java client: update version of SnakeYAML #1090 **new contributor Lukáš AKA @TheSpeedding**
        - CI: build macos release package on medium resource class #1131 @mobuchowski

      Additional bug fixes and more details: https://github.com/OpenLineage/OpenLineage/blob/main/CHANGELOG.md
- Project roadmap review [Harel]
    - Improved understanding of Airflow

        - Track DAG runs
        - Native lineage in operators
    - Increased adoption of OpenLineage consumers
        - Collaborate with data catalogs
    - Coverage by event producers
        - Increased support for Snowflake access history using tags
        - Data quality frameworks
        - Start thinking about data consumption integrations (e.g., on the BI layer)
    - Continue experimenting with a Flink integration, streaming in general
    - Increased support of column level lineage (e.g., SQL operators)
- Column-level lineage workshop [Howard]
    - Tutorial by Pawel Leszczynski available in the OpenLineage/workshops GitHub repo
    - Uses Jupyter and Spark
    - Covers:
        - Installing Marquez and Jupyter
        - Using column lineage feature in a Jupyter notebook
    - Requires:
        - Docker 17.05+
        - Docker Compose 1.29.1+
        - Git (preinstalled on most versions of MacOS; verify with `git version`)
        - 4 GB of available memory (the minimum for Docker — more is strongly recommended)
    - Preconfigured, including a token for Jupyter
    - Notebook contains scripts to set up environment, run Marquez, start Spark session
    - Allows you to see Marquez in action and understand how the APIs work
        - scripts return the JSON payloads
    - Other features are also well-suited to Jupyter notebooks, so more tutorials will be forthcoming
    - We welcome your contribution of additional tutorials!

September 8, 2022 (10am PT)

Attendees:

- TSC:
    - Mandy Chessel, Egeria Project Lead
    - Willy Lulciuc, Co-creator of Marquez
    - Mike Collado, Staff Software Engineer, Astronomer
    - Julien Le Dem, OpenLineage Project lead
- And:
    - Petr Hajek, Information Management Professional, Profinit
    - Harel Shein, Director of Engineering, Astronomer
    - Minkyu Park, Senior Software Engineer, Astronomer
    - Srikanth Venkat, Product Manager, Privacera
    - Peter Hicks, Senior Software Engineer, Astronomer
    - Michael Robinson, Software Engineer, Dev. Rel., Astronomer
    - Ross Turk, Senior Director of Community, Astronomer
    - Will Johnson, Senior Cloud Solution Architect, Azure Cloud, Microsoft
    - Ann Mary Justine, Expert Technologist, HP Enterprise
    - Benji Lampel, Ecosystem Engineer, Astronomer
    - Ernie Ostic, SVP of Product, MANTA
    - Howard Yoo, Staff Product Manager, Astronomer
    - Jakub Moravec, Software Architect, MANTA
    - Suparna Bhattacharya, Distinguished Technologist, HP Enterprise
    - John Thomas, Software Engineer, Dev. Rel., Astronomer

Agenda:

- Recent releases (0.13.0, 0.13.1, 0.14.0, 0.14.1)
- Native data quality in Airflow with OpenLineage
- MANTA integrations using OpenLineage

Meeting:

Notes:

- Recent releases (0.13.0, 0.13.1, 0.14.0, 0.14.1) [Michael R.]
    - 0.13.0
        - Added
            - Add BigQuery check support `#960` @denimalpaca
            - Add `RUNNING` `EventType` in spec and Python client `#972` @mzareba382
            - Use databases & schemas in SQL Extractors `#974` @JDarDagran
            - Implement Event forwarding feature via HTTP protocol `#995` @howardyoo
            - Introduce `SymlinksDatasetFacet` to spec `#936` @pawel-big-lebowski
            - Add Azure Cosmos Handler to Spark integration `#983` @hmoazam
            - Support OL Datasets in manual lineage inputs/outputs `#1015` @conorbev
            - Create ownership facets `#996` @julienledem

Changed

- Use `RUNNING` EventType in Flink integration for currently running jobs #985 @mzareba382
- Convert task object into JSON encodable when creating Airflow version facet #1018 @fm100

Fixed

- Add support for custom SQL queries in v3 Great Expectations API #1025 @collado-mike

  - 0.13.1
    - Fixed
      - Rename all parentRun occurrences to parent from Airflow integration #1037 @fm100
      - Do not change task instance during on_running event #1028 @JDarDagran
  - 0.14.0
    - Added
      - Support ABFSS and Hadoop Logical Relation in Column-level lineage #1008 @wjohnson
      - Add Kusto relation visitor #939 @hmoazam
      - Add ColumnLevelLineage facet doc #1020 @julienledem
      - Include symlinks dataset facet #935 @pawel-big-lebowski
      - Add support for dbt 1.3 beta's metadata changes #1051 @mobuchowski
      - Support Flink 1.15 #1009 @mzareba382
      - Add Redshift dialect to the SQL integration #1066 @mobuchowski

      Changed

      - Make the timeout configurable in the Spark integration #1050 @tnazarew

      Fixed

      - Add a dialect parameter to Great Expectations SQL parser calls #1049 @collado-mike
      - Fix Delta 2.1.0 with Spark 3.3.0 #1065 @pawel-big-lebowski
  - 0.14.1
    - Fixed
      - Fix Spark integration issues including error when no `openlineage.timeout` #1069 @pawel-big-lebowski
  - Notes:
    - Thank you to all the contributors! And a special shout out to new contributor Hanna Moazam!

- Native data quality in Airflow with OpenLineage [Benji]
  - Related webinar: https://www.astronomer.io/events/webinars/implementing-data-quality-checks-in-airflow/
  - Why Airflow?
    - In-pipeline checks
    - Immediate alerts
    - Lineage support
  - Use case
    - static checks
      - typed values
      - data ranges
      - temporal intervals
  - Two providers
    - SQL column check operator
      - "On Rails operator"
      - supports tolerance
      - supports partitioning with parameter
      - available checks:
        - min
        - max
        - unique check
        - distinct check
        - null check
      - qualifiers:
        - greater_than
        - geq_to
        - less_than
        - leq_to
        - equal_to
    - SQL table check operator
      - flexible
      - supports static checks
      - supports partitioning with parameter
      - uses cases:
        - checks that include aggregate values using the whole table
        - row count checks
        - schema checks
        - comparisons between multiple columns, both aggregated and not aggregated
  - Innovation: operators can now give data quality data directly to a lineage consumer (e.g., Marquez)
  - Note: the UI in the demo is part of the Datakin product
  - Can you talk about the OL packets?
    - the existing OL data quality facets are being used

- MANTA integrations using OpenLineage [Petr]
    - MANTA & MANTA Flow tools
        - unique column-level lineage parser of most data technologies
        - parses code to create database and reconstruct detailed column-level based on static analysis
        - represents end-to-end dependencies across technologies on enterprise level (indirect and direct)
        - challenge: integrating runtime lineage
        - MANTA connectors
            - reverse-engineer code
        - integration gets lineage from OpenLineage producers
            - e.g., Keboola, dbt, Airflow, Snowflake, Spark
            - converts the OpenLineage json files to MANTA objects
            - currently limited to the table level
            - for some technologies, Marquez libraries were used
        - MANTA repository model
            - underlying graph database
            - nodes: hierarchically organized objects
            - edges: relations
            - layers: physical, logical, runtime...
            - resources: all integration OL metadata sources
                - used to distinguish the sources of metadata
        - column-level project
            - we currently can get it if provided in facets
            - idea: extend the OpenLineage model for facet extensions which MANTA then analyzes statically
            - passes code, encoded using BASE64, in artifacts in job facets
        - status: in testing, beginning with Keboola
        - hope: to use the integration to increase number of producers we can consumer lineage from
    - Q & A
        - Have you used json files for metadata in the past?
        - No, but we are now and also using API calls
        - Egeria was in a similar situation
- Open Discussion
    - common metadata framework project at HP Enterprise will be added to agenda for a future meeting

August 11, 2022 (10am PT)

Attendees:

- TSC:
    - Mandy Chessel, Egeria Project Lead
    - Maciej Obuchowski, Software Engineer, GetInData, OpenLineage contributor
    - Willy Lulciuc, Co-creator of Marquez
    - Mike Collado, Staff Software Engineer, Astronomer
- And:
    - Petr Hajek, Information Management Professional, Profinit
    - Harel Shein, Director of Engineering, Astronomer
    - Minkyu Park, Senior Software Engineer, Astronomer
    - Sandeep Adwankar, Senior Technical Product Manager, AWS
    - Srikanth Venkat, Product Manager, Privacera
    - Peter Hicks, Senior Software Engineer, Astronomer
    - Michael Robinson, Software Engineer, Dev. Rel., Astronomer
    - Ross Turk, Senior Director of Community, Astronomer

Agenda:

- Docs site update
- Release 0.11.0 and 0.12.0 overview
- Extractors: examples and how to write them
- Open discussion

Meeting:

- Slides

Notes:

- Docs Site Update [Ross]
    - Lots of activity:
        - 19 closed PRs!
    - Infrastructure is becoming robust but not ready to launch yet
    - URL: openlineage.io/docs
    - Needed:
        - additions to About, Getting Started
        - additions to Object Model section
        - Completion of the Integration landing page
    - Stretch goal for next month: put it in production
- Recent releases [Michael R.]
    - 0.11.0

        - Added:
            - PMD to Java and Spark builds in CI #898 @merobi-hub
            - HTTP option to override timeout and properly close connections in openlineage-java lib. #909 @mobuchowski
            - Dynamic mapped tasks support to Airflow integration #906 @JDarDagran
            - SqlExtractor to Airflow integration #907 @JDarDagran
        - Changed:
            - Render templates as start of integration tests for TaskListener in the Airflow integration #870 @mobuchowski
            - When testing extractors in the Airflow integration, set the extractor length assertion dynamic #882 @denimalpaca
        - Fixed:
            - Spark casting error and session catalog support for iceberg in Spark integration #856 @wslulciuc
            - Dependencies bundled with openlineage-java lib. #855 @collado-mike
            - PMD reported issues #891 @pawel-big-lebowski
    - 0.12.0
        - Added:
            - Spark 3.3.0 support #950 @pawel-big-lebowski
            - Apache Flink integration #951 @mobuchowski
            - Ability to extend column level lineage mechanism #922 @pawel-big-lebowski
            - ErrorMessageRunFacet #897 @mobuchowski
            - SQLCheckExtractors #717 @denimalpaca
            - RedshiftSQLExtractor & RedshiftDataExtractor #930 @JDarDagran
            - Dataset builder for AlterTableCommand #927 @tnazarew
        - Changed:
            - Airflow integration: allow lineage metadata to flow through inlets and outlets #914 @fenil25
            - Limit Delta events #905 @pawel-big-lebowski
        - Fixed:
            - Fix noclassdef error #942 @pawel-big-lebowski
            - Limit size of serialized plan #917 @pawel-big-lebowski
- Extractors: example and tutorial [Maciej]
    - Airflow: defined tasks composed of pieces of code executed by operators (which number in the hundreds)
    - Extraction of data
        - Operator example
            - accesses operator object
            - processes it in customizable way
            - runtime information can also be extracted
                - additional method (`extract_on_complete`)
        - Metadata matches the structure of the OpenLineage spec
            - supplemented by facets (`job_facets`)
        - How to expose:
            - set up env vars supplying full paths to extractor classes (separated by commas)
        - Help available from OpenLineage side:
            - SQL parser
            - common library covering a few systems
            - community help on Slack and Github (please contribute your custom extractors!)
    - Typical problems
        - incorrect path provided
            - more debugging info would help in this case – help welcome!
        - Imports from Airflow
            - Python prevents import cycles, leading to extractor failure
            - use local imports instead, with type checking
    - What's the future?
        - debugability
        - additional coverage – PythonOperator, TaskFlow
            - watching AIP-44 in Airflow to make it more data-aware
        - covering hooks
            - e.g., with PythonOperator
    - See also: new doc about this on the forthcoming docs site
    - Q & A
        - Does the documentation link out to the extractors currently in the Airflow library? Helpful for examples
            - we need to add links to the doc
- Open Discussion
    - Mandy: presenting at Open Source Summit, Dublin, 9/15
    - Ross: talking at ApacheCon in New Orleans
    - Ross: should we create a calendar of events?
    - Maciej: we're looking for feedback on the Flink integration
        - let us know if it solves your problems, etc.
    - Mandy: Egeria running a hackathon as part of the Grace Hopper Open Source Day event on 9/16; theme: sustainability

July 14, 2022 (10am PT)

Attendees:

- TSC:
    - Willy Lulciuc: Co-creator of Marquez
    - Mike Collado: Staff Software Engineer, Astronomer
    - Julien Le Dem: OpenLineage Project lead
- And:
    - Ernie Ostic, SVP of Product, Manta
    - Ross Turk, Senior Director of Community, Astronomer

    - Minkyu Park, Senior Software Engineer, Astronomer
    - Peter Hicks, Senior Software Engineer, Astronomer
    - Michael Robinson, Software Engineer, Dev. Rel., Astronomer
    - Sandeep Adwankar: Senior Technical Product Manager, AWS
    - Will Johnson: Senior Cloud Solution Architect, Azure Cloud, Microsoft
    - John Thomas: Software Engineer, Dev. Rel., Astronomer
    - Chandru Sugunan: Product Manager, Azure Cloud, Microsoft
    - Petr Hajek, Information Management Professional, Profinit
    - Colin Schaub, Lead API Engineer, API Platform Lead, Cargill
    - Mark Chiarelli, Senior Consultant, MarkLogic
    - Sam Holmberg, Software Engineer, Astronomer
    - Pawe Leszczyski, Software Engineer, GetInData

Agenda:

- Recent talks [Julien]
- Recent release: 0.10.0 [Michael R.]
- Flink integration [Pawe, Maciej]
- New docs site [Ross]
- Discuss: streaming services in Flink integration [Will]
- Open discussion
    - OL philosophy for streaming in general

Meeting:

Slides: https://bit.ly/3c9o1U1

Notes:

- Recent talks
    - Ross, “What Is Data Lineage and Why Should I Care?”
    - Maciej & Pawe, “OpenLineage & Airflow: Data Lineage has never been Easier”
    - Willy, “Automating Airflow Backfills with Marquez”
    - Michael C., “Data Lineage with Apache Airflow and Apache Spark”
    - Ross & Michael R., “An Introduction to Data Lineage with Airflow and Marquez”
    - Julien, “Observability for Data Pipelines with OpenLineage”
    - Michael C., “Cross-platform Lineage with OpenLineage"
- Release 0.10.0
    - Added:
        - Extend SaveIntoDataSourceCommandVisitor to extract schema from LocalRelation and LogicalRdd in Spark integration (#794) @pawel-big-lebowski
        - Add InMemoryRelationInputDatasetBuilder for InMemory datasets to Spark integration (#818) @pawel-big-lebowski
        - Add SnowflakeOperatorAsync extractor support to Airflow integration (#869) @denimalpaca
        - Add PMD analysis to proxy project (#889) @howardyoo
        - Add static code analysis tool mypy to run in CI against all Python modules (#802) @howardyoo
        - Add copyright to source files (#755) @merobi-hub

      Changed:
        - Skip FunctionRegistry.class serialization in Spark integration (#828) @mobuchowski
        - Reduce OL event payload size by excluding local data and including output node in start events (#881) @collado-mike
        - Install new rust-based SQL parser by default in Airflow integration (#835) @mobuchowski
        - Improve overall pytest and integration tests for Airflow integration (#851, #858) @denimalpaca
        - Split Spark integration into submodules (#834, #890) @tnazarew @mobuchowski
- Flink integration
    - Entry point: built Flink example app to find out if metadata, schema extractable
    - Maciej also successfully read data from Iceberg
    - Flink provides two APIs
    - Created integration tests for all use cases, added them to CircleCI
    - New Java client: different configs for HTTP, Kafka endpoints
    - Missing feature: make sure crashing integration doesn't kill a Flink job
    - Coming soon: experimental version
        - not focused on streaming currently
        - focus: how to extract info from Flink
        - feedback from community desired
    - Q & A
        - Will: is the code an extension of OL or an integration?
            - an integration akin to the dbt integration
        - Willy: any changes to the spec/schema? Is the state part of the payload?
            - new state should be added (currently "other")
- New docs site
    - Up until today, docs have been on the website and spread throughout READMEs
    - Docusaurus deployment now available
    - Changes to structure as well as content welcome
    - Not currently live but will be soon
    - Can be hosted at docs.openlineage.io
    - Everything is in Markdown
    - Another motivation: Keboola use case not part of the codebase, so a docs site could describe it

        - Next milestone: we all decide to publish it
        - Q & A
            - Willy: let's add a section on defining custom facets
            - Ross: feel free to add another page stub
            - Ross: also need a FAQ
            - Julien: we could autogenerate some docs
            - Ross: there are downsides to such an approach
            - Julien: let's open issues when answers aren't good enough
            - Willy: descriptions of facets could be improved
            - Julien: we could version them
            - Ross: I'll look for signs that people are not finding docs on the version they are using
- Discussion: streaming in Flink integration
    - Has there been any evolution in the thinking on support for streaming?
        - Julien: start event, complete event, snapshots in between limited to certain number per time interval
        - Pawe: we can make the snapshot volume configurable
    - Does Flink support sending data to multiple tables like Spark?
        - Yes, multiple outputs supported by OpenLineage model
        - Marquez, the reference implementation of OL, combines the outputs
    - Looking forward to seeing this documented on the new docs site
- Open discussion
    - What's the logical approach to avoid overloading the backend with lineage events? [Colin]
        - Pawe: we only send events when checkpoints change; configurable for more events
        - Will: at Microsoft we're working on a fix that caches and consolidates OL events
    - It'd be awesome to see example payloads for streaming in docs [Colin]
        - Ross: they're currently spread out; it'd be nice to have them in one place
    - How can we create custom facets? [Sandeep]
        - Julien: two options; anyone can create a custom facet without asking permission, or open a proposal/issue

June 9th, 2022 (10am PT)

Attendees:

- TSC:
    - Mandy Chessel: Egeria Project Lead
    - Maciej Obuchowski: Software Engineer, GetInData, OpenLineage contributor
    - Willy Lulciuc: Co-creator of Marquez
    - Mike Collado: Staff Software Engineer, Datakin
- And:
    - Ernie Ostic, SVP of Product, Manta
    - Šimon Rajan, Senior Business Intelligence Consultant, Profinit
    - Sheeri Cabral: Technical Product Manager, Lineage, Collibra
    - Ross Turk, Senior Director of Community, Astronomer
    - Howard Yoo, Staff Product Manager, Astronomer
    - Minkyu Park, Senior Software Engineer, Astronomer
    - Peter Hicks, Senior Software Engineer, Astronomer
    - Jakub Moravec, Software Architect, Manta
    - Michael Robinson, Software Engineer, Dev. Rel., Astronomer

Agenda:

- Release: 0.9.0 [Michael R.]
- A recent blog post about Snowflake [Ross T.]
- Great Expectations integration [Michael C.]
- dbt integration [Willy]
- Open discussion

Meeting:

Notes:

- Release 0.9.0 [Michael R.]
    - We added:
        - Spark: Column-level lineage introduced for Spark integration (#698, #645) @pawel-big-lebowski
        - Java: Spark to use Java client directly (#774) @mobuchowski
        - Clients: Add OPENLINEAGE_DISABLED environment variable which overrides config to NoopTransport (#780) @mobuchowski
    - For the bug fixes and more information, see the Github repo.
    - Shout out to new contributor Jakub Dardziski, who contributed a bug fix to this release!
- Snowflake Blog Post [Ross]
    - topic: a new integration between OL and Snowflake
    - integration is the first OL extractor to process query logs
    - design:
        - an Airflow pipeline processes queries against Snowflake
        - separate job: pulls access history and assembles lineage metadata
        - two angles: Airflow sees it, Snowflake records it
    - the meat of the integration: a view that does untold SQL madness to emit JSON to send to OL
    - result: you can study the transformation by asking Snowflake AND Airflow about it
    - required: having access history enabled in your Snowflake account (which requires special access level)

    - Q & A
        - Howard: is the access history task part of the DAG?
        - Ross: yes, there's a separate DAG that pulls the view and emits the events
        - Howard: what's the scope of the metadata?
        - Ross: the account level
        - Michael C: in Airflow integration, there's a parent/child relationship; is this captured?
        - Ross: there are 2 jobs/runs, and there's work ongoing to emit metadata from Airflow (task name)
- Great Expectations integration [Michael C.]
    - validation actions in GE execute after validation code does
    - metadata extracted from these and transformed into facets
    - recent update: the integration now supports version 3 of the GE API
    - some configuration ongoing: currently you need to set up validation actions in GE
    - Q & A
        - Willy: is the metadata emitted as facets?
        - Michael C.: yes, two
- dbt integration [Willy]
    - a demo on getting started with the OL-dbt library
        - pip install the integration library and dbt
        - configure the dbt profile
        - run seed command and run command in dbt
        - the integration extracts metadata from the different views
        - in Marquez, the UI displays the input/output datasets, job history, and the SQL
- Open discussion
    - Howard: what is the process for becoming a committer?
        - Maciej: nomination by a committer then a vote
        - Sheeri: is coding beforehand recommended?
        - Maciej: contribution to the project is expected
        - Willy: no timeline on the process, but we are going to try to hold a regular vote
        - Ross: project documentation covers this but is incomplete
        - Michael C.: is this process defined by the LFAI?
    - Ross: contributions to the website, workshops are welcome!
    - Michael R.: we're in the process of moving the meeting recordings to our YouTube channel

May 19th, 2022 (10am PT)

Agenda:

- Releases: 0.7.1, 0.8.1, 0.8.2 preview [Michael R.]
- Column-level lineage [Pawe]
- Open discussion

Attendees:

- TSC:
    - Mike Collado: Staff Software Engineer, Datakin
    - Maciej Obuchowski: Software Engineer, GetInData, OpenLineage contributor
    - Julien Le Dem: OpenLineage Project lead
    - Willy Lulciuc: Co-creator of Marquez
- And:
    - Ernie Ostic: SVP of Product, Manta
    - Sandeep Adwankar: Senior Technical Product Manager, AWS
    - Pawe Leszczyski, Software Engineer, GetinData
    - Howard Yoo: Staff Product Manager, Astronomer
    - Michael Robinson: Developer Relations Engineer, Astronomer
    - Ross Turk: Senior Director of Community, Astronomer
    - Minkyu Park: Senior Software Engineer, Astronomer
    - Will Johnson: Senior Cloud Solution Architect, Azure Cloud, Microsoft

Meeting:

Notes:

- Releases
    - 0.8.2
        - Added

            - openlineage-airflow now supports getting credentials from Airflows secrets backend (#723) @mobuchowski
            - openlineage-spark now supports Azure Databricks Credential Passthrough (#595) @wjohnson
            - openlineage-spark detects datasets wrapped by ExternalRDDs (#746) @collado-mike

        Fixed

            - PostgresOperator fails to retrieve host and conn during extraction (#705) @sekikn
            - SQL parser accepts lists of sql statements (#734) @mobuchowski
    - 0.8.1
        - Added

            - Airflow integration uses new TaskInstance listener API for Airflow 2.3+ (#508) @mobuchowski

- Support for HiveTableRelation as input source in Spark integration (#683) @collado-mike
- Add HTTP and Kafka Client to openlineage-java lib (#480) @wslulciuc, @mobuchowski
- New SQL parser, used by Postgres, Snowflake, Great Expectations integrations (#644) @mobuchowski

Fixed

GreatExpectations: Fixed bug when invoking GreatExpectations using v3 API (#683) @collado-mike

- 0.7.1
  - Added
    - Python implements Transport interface - HTTP and Kafka transports are available (#530) @mobuchowski
    - Add UnknownOperatorAttributeRunFacet and support in lineage backend (#547) @collado-mike
    - Support Spark 3.2.1 (#607) @pawel-big-lebowski
    - Add StorageDatasetFacet to spec (#620) @pawel-big-lebowski
    - README.md created at OpenLineage/integrations for compatibility matrix (#663) @howardyoo

    Fixed

    - Airflow: custom extractors lookup uses only get_operator_classnames method (#656) @mobuchowski
    - Dagster: handle updated PipelineRun in OpenLineage sensor unit test (#624) @dominiquetipton
    - Delta improvements (#626) @collado-mike
    - Fix SqlDwDatabricksVisitor for Spark2 (#630) @wjohnson
    - Airflow: remove redundant logging from GE import (#657) @mobuchowski
    - Fix Shebang issue in Spark's wait-for-it.sh (#658) @mobuchowski
    - Update parent_run_id to be a uuid from the dag name and run_id (#664) @collado-mike
    - Spark: fix time zone inconsistency in testSerializeRunEvent (#681) @sekikn

- Communication reminders [Julien]
- Agenda [Julien]
- Column-level lineage [Pawe]
  - Linked to 4 PRs, the first being a proposal
  - The second has been merged, but the core mechanism is turned off
  - 3 requirements:
    - Outputs labeled with expression IDs
    - Inputs with expression IDs
    - Dependencies
  - Once it is turned on, each OL event will receive a new JSON field
  - It would be great to be able to extend this API (currently on the roadmap)
  - Q & A
    - Will: handling user-defined functions: is the solution already generic enough?
      - The answer will depend on testing, but I suspect that the answer is yes
      - The team at Microsoft would be excited to learn that the solution will handle UDFs
    - Julien: the next challenge will be to ensure that all the integrations support column-level lineage
- Open discussion
  - Willy: in Mqz we need to start handling col-level lineage, and has anyone thought about how this might work?
    - Julien: lineage endpoint for col-level lineage to layer on top of what already exists
    - Willy: this makes sense – we could use the method for input and output datasets as a model
    - Michael C.: I don't know that we need to add an endpoint – we could augment the existing one to do something with the data
    - Willy: how do we expect this to be visualized?
      - Julien: not quite sure
      - Michael C.: there are a number of different ways we could do this, including isolating relevant dataset fields

Apr 13th, 2022 (9am PT)

Attendees:

- TSC:
  - Maciej Obuchowski: Software Engineer, GetInData, OpenLineage contributor
  - Julien Le Dem: OpenLineage Project lead
  - Mandy Chessel: Egeria Project Lead
  - Willy Lulciuc: Co-creator of Marquez
- And:
  - Sheeri Cabral: Technical Product Manager, Lineage, Collibra
  - Michael Robinson: Software Engineer, Developer Relations, Astronomer
  - John Thomas: Support Engineer, Astronomer
  - Ross Turk: Senior Director of Community, Astronomer
  - Minkyu Park: Senior Software Engineer, Astronomer
  - Ernie Ostic: SVP of Product, Manta
  - Kelsy Brennan: Lead Developer, Environmental Intelligence Group
  - Dalin Kim: Data Engineer, Northwestern Mutual
  - Will Johnson: Microsoft, OL contributor
  - Jorge
  - Jakub Moravec: Software Architect, Manta
  - Chandru Sugunan: Product Manager, Azure Cloud, Microsoft

Agenda:

- 0.6.2 release overview [Michael R.]
- Transports in OpenLineage clients [Maciej]
- Airflow integration update [Maciej]

- Dagster integration retrospective [Dalin]
- Open discussion

Meeting info:

Notes:

- Introductions
- Communication channels overview [Julien]
- Agenda overview [Julien]
- 0.6.2 release overview [Michael R.]

  Added

  - CI: add integration tests for Airflow's SnowflakeOperator and dbt-Snowflake @mobuchowski
    - #611
    - Workaround necessitated by the fact we have only 1 schema in the Snowflake db
    - This creates conflicts between different Airflow versions
    - By contrast: in BigQuery, different schemas are prefixed with Airflow versions
  - Introduce DatasetVersion facet in spec @pawel-big-lebowski
    - #580
    - Problem: the spec did not support dataset versioning (which is needed for providers like Iceberg, Delta)
    - Solution: this change introduced a DatasetVersionFacet in spec
  - Airflow: add external query ID facet @mobuchowski
    - #546
    - Issue: jobs that ran on external systems like BigQuery or Snowflake were identified by their query IDs.
    - This change added a facet that exposes this collected query ID, so that an OpenLineage job run can be associated with that external job.

  Fixed

  - Complete Fix of Snowflake Extractor get_hook() Bug @denimalpaca
    - #589
    - In #507, an incorrect fix was made to the Snowflake Extractor to allow for the operator's new get_db_hook() method.
    - Solution: this change checks for the existence of the get_db_hook() method in the underlying Operator, then get_hook() calls the correct version of the underlying method, enabling it
  - Update artwork @rossturk
    - #605
    - This change updated artwork in the README.md with the latest versions from recent presentations and other sources.

- Transports in OpenLineage clients [Maciej]
  - Currently, OL clients can only read HTTP data
  - Common request: ability to read Kafka
  - This feature will offer a language-independent solution
  - Status: Python client implementation merged, Java implementation close to being merged
  - Timeline: next release (0.7.0)
- Airflow integration [Maciej]
  - TaskInstance listener-based plugin not ready yet
  - Status: waiting for Airflow 2.3 to be merged (due by April 18, 2022)
  - Ready upon Airflow 2.3 release
  - New SQL parser
    - Used in Snowflake, Postgres, GE integrations
    - Missing: API for SQL queries
    - Formerly had a SQL parser but based on guesswork and fragile reliance on language patterns
    - Solution: AST (abstract syntax trees), not guesswork
    - Features strong typing, Enums, encapsulation
    - Language: Rust
      - Disadvantages: additional language, distribution
      - Advantages: high-quality libraries, possible new applications, e.g. Spark
    - Unified API: previous implementation still exists for users of older architectures
    - Utilizable in Java
    - Makes all tasks using SQL easier
    - Will J.: can I inject a different SQL parser that I want to use?
      - Unified API would make this possible
      - Goal is to work with different dialects, implementations
- Dagster integration [Dalin]
  - Initial proposal: use custom OL executor as thin wrapper over existing executors
  - Challenges:
    - OL handling tightly coupled with actual job runs
    - Requires multiple custom executors to main flexibility
    - Incomplete events (only op-level)
  - Solution: use Dagster's OL sensor that tails Dagster event logs for tracking metadata
  - Lessons learned:
    - Non-sharded event log storage must be used for sensor to access all event logs across runs
    - Sensor's cursor does not get updated on an exception. Typical use of cursors is to submit a run request while tracking some state. To guarantee atomic operation with the cursor, the cursor update gets processed only after the sensor function exits.
  - Event type conversion

        - Dagster event types converted to OpenLineage events
    - Architecture
        - Sensor defined under a repository then converted and sent to the OL backend
    - Lineage collected at job level only; dataset tracking being explored
        - Currently datasets being stored as Dagster assets
        - This a manual/custom solution
    - 3M event logs processed, used as part of published telemetry report
    - Will J.: what's been the timeline since inception of the idea to now?
        - December 2021; integrated within ~1 month's time
        - Bulk of time was spent on understanding Dagster
        - OL sensor is configurable and can be started late while still catching the first events
    - Willy: do you remember the issue # or title you were waiting for?
    - Julien: Dalin reached out on Slack initially. We started a new channel, my small contribution was to reach out to the Dagster community to facilitate collaboration; we can support new integrations in this way. Thanks to Sandy from the Dagster community for help with this.
        - Don't hesitate to reach out for help!
- Open discussion
    - Mandy: where do I submit my blog? Two website repos are a source of confusion.
    - Julien: Ross and Michael R. can help.
    - Ross: branching could solve this problem. We welcome blog posts from anyone in the community.
    - Will J.: parent/child relationships in OL. Problem in Azure: Databricks connector has a parent execution inside Spark and a child execution that is not connected. Spark issues a parent ID that's not being caught. Currently using a workaround. What's the right way to emit a parent/child relationship?
        - Julien: this is relevant to the ParentRunFacet in OL. Michael C. is working on this in Marquez. Recommended: create an issue about this and ping Michael C.
        - Maciej: this functional in the Airflow integration for Spark jobs.
        - Julien: this issue could be documented better.

Mar 9th, 2022 (9am PT)

Attendees:

- TSC:
    - Mike Collado: Staff Software Engineer, Datakin
    - Maciej Obuchowski: Software Engineer, GetInData, OpenLineage contributor
    - Julien Le Dem: OpenLineage Project lead
    - Mandy Chessel: Egeria Project Lead
    - Willy Lulciuc: Co-creator of Marquez
- And:
    - Michael Robinson: Dev Rel Engineer
    - Ross Turk: VP of Marketing, Datakin
    - Minkyu Park: Senior Software Engineer, Datakin
    - Srikanth Venkat: Product Manager, Privacera
    - John Thomas: Support Engineer, Datakin
    - Will Johnson: Senior Cloud Solution Architect, Azure Cloud, Microsoft
    - Pawe Leszczyski, Software Engineer, GetinData
    - Sheeri Cabral, Technical Product Manager, Lineage, Collibra
    - Michal Bartos, Software Engineer, MANTA
    - Chandru Sugunan, Product Manager, Azure Cloud, Microsoft
    - Caroline Fahrenkrog, Product Manager, MANTA Scanners
    - John Montroy, Backend Engineer

Agenda:

- New committers [Julien]
- Release overview (0.6.0-0.6.1) [Michael R.]
- Process for blog posts [Ross]
- Retrospective: Spark integration [Willy et al.]
- Open discussion

Meeting:

Notes:

- New committers [Julien]
    - 4 new committers were voted in last week
    - We had fallen behind
    - Congratulations to all
- Release overview (0.6.0-0.6.1) [Michael R.]
    - Added
        - Extract source code of PythonOperator code similar to SQL facet @mobuchowski (0.6.0)
        - Airflow: extract source code from BashOperator @mobuchowski (0.6.0)
            - These first two additions are similar to SQL facet
            - Offer the ability to see top-level code
        - Add DatasetLifecycleStateDatasetFacet to spec @pawel-big-lebowski (0.6.0)
            - Captures when someone is conducting dataset operations (overwrite, create, etc.)
        - Add generic facet to collect environmental properties (EnvironmentFacet) @harishsune (0.6.0)
            - Collects environment variables

                - Depends on Databricks runtime but can be reused in other environments
            - OpenLineage sensor for OpenLineage-Dagster integration @dalinkim (0.6.0)
                - The first iteration of the Dagster integration to get lineage from Dagster
            - Java-client: make generator generate enums as well @pawel-big-lebowski (0.6.0)
                - Small addition to Java client feat. better types; was string
        - Fixed
            - Airflow: increase import timeout in tests, fix exit from integration @mobuchowski (0.6.0)
                - The former was a particular issue with the Great Expectations integration
            - Reduce logging level for import errors to info @rossturk (0.6.0)
                - Airflow users were seeing warnings about missing packages if they weren't using a part of an integration
                - This fix reduced the level to Info
            - Remove AWS secret keys and extraneous Snowflake parameters from connection URI @collado-mike (0.6.0)
                - Parses Snowflake connection URIs to exclude some parameters that broke lineage or posed security concerns (e.g., login data)
                - Some keys are Snowflake-specific, but more can be added from other data sources
            - Convert to LifecycleStateChangeDatasetFacet @pawel-big-lebowski (0.6.0)
                - Mandates the LifecycleStateChange facet from the global spec rather than the custom tableStateChange facet used in the past
            - Catch possible failures when emitting events and log them @mobuchowski (0.6.1)
                - Previously when an OL event failed to emit, this could break an integration
                - This fix catches possible failures and logs them
- Process for blog posts [Ross]
    - Moving the process to Github Issues
    - Follow release tracker there
    - Go to https://github.com/OpenLineage/website/tree/main/contents/blog to create posts
    - No one will have a monopoly
    - Proposals for blog posts also welcome and we can support your efforts with outlines, feedback
    - Throw your ideas on the issue tracker on Github
- Retrospective: Spark integration [Willy et al.]
    - Willy: originally this part of Marquez – the inspiration behind OL
        - OL was prototyped in Marquez with a few integrations, one of which was Spark (other: Airflow)
        - Donated the integration to OL
    - Srikanth: #559 very helpful to Azure
    - Pawel: is anything missing from the Spark integration? E.g., column-level lineage?
    - Will: yes to column-level; also, delta tables are an issue due to complexity; Spark 3.2 support also welcome
    - Maciej: should be more active about tracking projects we have integrations with; add to test matrix
    - Julien: let's open some issues to address these
- Open Discussion
    - Flink updates? [Julien]
        - Maciej: initial exploration is done
            - challenge: Flink has 4 APIs
            - prioritizing Kafka lineage currently because most jobs are writing to/from Kafka
            - track this on Github milestones, contribute, ask questions there
        - Will: can you share thoughts on the data model? How would this show up in MZ? How often are you emitting lineage?
        - Maciej: trying to model entire Flink run as one event
        - Srikanth: proposed two separate streams, one for data updates and one for metadata
        - Julien: do we have an issue on this topic in the repo?
        - Michael C.: only a general proposal doc, not one on the overall strategy; this worth a proposal doc
        - Julien: see notes for ticket number; MC will create the ticket
            - https://github.com/OpenLineage/OpenLineage/issues/596
        - Srikanth: we can collaborate offline

Feb 9th 2022 (9am PT)

Attendees:

- TSC:
    - Mike Collado: Staff Software Engineer, Datakin
    - Maciej Obuchowski: Software Engineer, GetInData, OpenLineage contributor
    - Julien Le Dem: OpenLineage Project lead
- And:
    - Michael Robinson: Dev Rel Engineer
    - Ross Turk: VP of Marketing, Datakin
    - Minkyu Park: Senior Software Engineer, Datakin
    - Srikanth Venkat: Product Manager, Privacera
    - John Thomas: Support Engineer, Datakin
    - Peter Scharling: EI Group
    - Peter Hicks: Senior Software Engineer, Datakin
    - Dalin Kim: Data Engineer, Northwestern Mutual
    - Kevin Mellott: Data Engineer, Northwestern Mutual
    - Will Johnson: Senior Cloud Solution Architect, Azure Cloud, Microsoft
    - Kelsy Brennan: EI Group
    - Aaron Colcord: Data Engineer, Northwestern Mutual

Agenda:

- OpenLineage recent release overview (0.5.1) [Julien]
- TaskInstanceListener now official way to integrate with Airflow [Julien]
- Apache Flink integration [Julien]
- Dagster integration demo [Dalin]
- Open Discussion

Meeting:

Slides

Notes:

- OpenLineage recent release overview (0.5.1) [Julien]
  - No 0.5.0 due to bug
  - Support for dbt-spark adapter
  - New backend to proxy OL events
  - Support for custom facets
- TaskInstanceListener now official way to integrate with Airflow [Julien]
  - Integration runs on worker side
  - Will be in next OL release of airflow (2.3)
  - Thanks to Maciej for his work on this
- Apache Flink integration [Julien]
  - Ticket for discussion available
  - Integration test setup
  - Early stages
- Dagster integration demo [Dalin]
  - Initiated by Dalin Kim
  - OL used with Dagster on orchestration layer
  - Utilizes Dagster sensor
  - Introduces OL sensor that can be added to Dagster repo definition
  - Uses cursor to keep track of ID
  - Looking for feedback after review complete
  - Discussion:
    - Dalin: needed: way to interpret Dagster asset for OL
    - Julien: common code from Great Expectations/Dagster integrations
    - Michael C: do you pass parent run ID in child job when sending the job to MZ?
    - Hierarchy can be extended indefinitely – parent/child relationship can be modeled
    - Maciej: the sensor kept failing – does this mean the events persisted despite being down?
    - Dalin: yes - the sensor's cursor is tracked, so even if repo goes down it should be able to pick up from last cursor
    - Dalin: hoping for more feedback
    - Julien: slides will be posted on slack channel, also tickets
- Open discussion
  - Will: how is OL ensuring consistency of datasets across integrations?
  - Julien: (jokingly) Read the docs! Naming conventions for datasets can be found there
  - Julien: need for tutorial on creating integrations
  - Srikanth: have done some of this work in Atlas
  - Kevin: are there libraries on the horizon to play this role? (Julien: yes)
  - Srikanth: it would be good to have model spec to provide enforceable standard
  - Julien: agreed; currently models are based on the JSON schema spec
  - Julien: contributions welcome; opening a ticket about this makes sense
  - Will: Flink integration: MZ focused on batch jobs
  - Julien: we want to make sure we need to add checkpointing
  - Julien: there will be discussion in OLMZ communities about this
    - In MZ, there are questions about what counts as a version or not
  - Julien: a consistent model is needed
  - Julien: one solution being looked into is Arrow
  - Julien: everyone should feel welcome to propose agenda items (even old projects)
  - Srikanth: who are you working with on the Flink comms side? Will get back to you.

Jan 12th 2022 (9am PT)

Attendees:

- TSC:
  - Mike Collado: Eng, Datakin
  - Mandy Chessel: Lead Egeria project
  - Maciej Obuchowski: Eng GetInData, OpenLineage contributor
  - Willy Lulciuc: Co-creator of Marquez
  - Julien: OpenLineage Project lead

- And:
  - Michael Robinson: Dev Rel
  - Ross Turk: VP Marketing Datakin
  - Minkyu Park: Dev at Datakin
  - Conor Beverland: Senior Dir of Product, Astronomer

- Srikanth Venkat, Product Management, Privacera
- Mark Taylor, Technical P.M., Microsoft
- Harish Sune, Technical Architect, NE Analytics
- Joshua Wankowski, Associate Data Engineer, Northwestern Mutual
- Arpita Grange, Senior Technical Lead for Business Intelligence Solutions, Asurion

Agenda:

- OpenLineage recent releases overview [Julien]
  - OpenLineage 0.4 release overview: https://github.com/OpenLineage/OpenLineage/releases/tag/0.4.0
    - Databricks install README and init scripts (by Will)
    - Iceberg integration (by Pawel)
    - Kafka read and write support (by Olek and Mike)
    - Arbitrary parameters supported in HTTP URL construction (by Will)
    - Increased coverage (Pawel/Maciej)
  - OpenLineage 0.5 release overview
    - https://github.com/OpenLineage/OpenLineage/compare/0.4.0...main
- Egeria support for OpenLineage [Mandy]
  - https://odpi.github.io/egeria-docs/features/lineage-management/overview/#integrating-with-the-openlineage-standard
- Airflow TaskListener for OpenLineage integration [Maciej]
- Open discussion

Meeting:

Slides

Notes:

0.4 release [Willy]:

- Databricks install README and init scripts (by Will)
- Iceberg integration (Pawel)
  - Iceberg adoption already strong
- Kafka read and write support (Olek and Mike)
- Arbitrary parameters supported in HTTP URL construction (Will)
- Increased coverage (Pawel and Maciej)

0.5 preview [Willy]:

- Add Spark support to openlineage-dbt lib. (by Maciej)
- New extensible API to handle Spark events for openlineage-spark lib (Mike)
- New proxy HTTP backend to route events to event streams (Mandy and Willy)
- Increase coverage of sparkV2 cmds for openlineage-spark lib. (Pawel)
- Added HTTP client to openlineage-java lib. (Willy)
- Thanks go to Mike Collado for work on PRs, proposal; also to Mandy for work on HTTP backend over last two months
- HTTP client will decrease confusion about how to capture metadata

Tasklistener for OL Integration [Maciej]:

1.10 required modifying each DAG, which was cumbersome and not compatible with 2.1

2.1: lineage backend comparable to Apache Atlas' old backend

- benefit: provides all info about events
- downside: cannot notify about task starts/failures

2.3: Airflow Event Listener

- Status: not merged yet, in final reviews for deployment with 0.6
- Improvements: transparent, less exposure, enables pull model using queue, enables Egeria and other projects in the future (e.g., DataHub)
- Discussion [Julien, Maciej, Willy, Mike]:
  - generic: supports additional functionality
  - extendable to different kinds of events, e.g., scheduling
  - makes more data available
  - much less brittle because depends on public API
  - requires little configuration
  - will not do away with registration of listeners/extractors
  - entry point mechanism comparable to service loaded in Java, requires env variables
  - theoretically possible to back port it to earlier versions of Airflow (as far as 1.10)
  - possibly helpful to document that we have 3 approaches but are not recommending older ones, mention that this changes only how we collate
  - older approaches can be deprecated; it will be important to monitor the community to determine timing of this

Egeria Support for OpenLineage [Mandy]:

- Monthly releases
- OpenLineage support ready in recent release
- Metaphor: Lego blocks
  - OL events can be brought in through API or proxy backend with Kafka

    - events augmentable in Egeria, storable or publishable in Marquez or Kafka for distribution or to log store (e.g., file system)
- Can validate that a process is running correctly
- See documentation in Egeria about proxy backend and extensions, API mechanism
- Diagram in documentation illustrates capabilities
- Discussion [Julien, Mandy, Srikanth, Mike]:
    - Egeria sees value of OpenLineage
    - Engine is uncoupled from receivers
    - Endpoint is simple, allowing independent management of processes
    - Some transformation of payload during storage
    - Kafka integration coming in 0.5
    - Customers expect ability to filter data
    - Varying granularity of metadata already possible through versioning with Marquez

Open Discussion:

Proposal to convert licenses to SPDX [Michael]: no objections

Dec 8th 2021 (9am PT)

Attendees:

TSC:

- Mike Collado, Staff Engineer, Datakin
- Willy Lulciuc, Co-creator of Marquez, Datakin
- Mandy Chessel, Egeria Project Lead
- Julian Le Dem, OpenLineage Project Lead, CTO Datakin

And:

- Peter Hicks, Software Engineer, Datakin
- Srikanth Venkat, Product Management, Microsoft
- Ross Turk, VP Marketing, Datakin
- Maciej Obuchowski: Engineer GetInData, OpenLineage contributor
- John Thomas, Support Engineer, Datakin
- Minkyu Park, Engineer, Datakin
- Michael Robinson, Dev Rel Engineer
- Will Johnson, Senior Cloud Solution Architect, Azure Cloud, Microsoft
- Mark Taylor, Principal Technical PM, Microsoft
- Travis Hilbert, Associate Consultant, Microsoft

Agenda:

- SPDX headers [Mandy]
- Azure Purview + OpenLineage [Will and Mark]
- Logging backend (OpenTelemetry) [Julien]
- Open discussion

Meeting recording:

Slides

Notes:

Software Package Data Exchange (SPDX) Tags [Mandy]

- Open standard for creating software bill of materials
- Includes set of short identifiers for open source licenses
    - both human readable and machine processable
    - easy to maintain and validate
- Full license added in License file at top of git repository
- Each file includes the SPDX-License-Identifier tag
- Proposed: we use this approach in OpenLineage
- Becoming a best practice in open source development
- Julien: "a no brainer"
- Next question: how to integrate (implement going forward or add tags throughout project?)
- Willy: throughout existing; should also do with Marquez
- Mike: update build check to check for tags in new source files?
- Julien: must find right build plugins, two passes might be necessary
- Julien: all agreed?; adopted; someone should create issue
- Julien: Maven plugins exist to check and add tag if missing

Azure Purview Integration [Srikanth, Will]

- Overview of Azure Purview
    - Metadata and governance platform across MS, new
    - End-to-end governance practices
    - Goal is to fill gaps in lineage

- Database Lineage in Azure Purview
    - Began as hackathon project at Microsoft
    - Sought way to send lineage data directly to Purview (rather than use architecture of Marquez)
    - Azure Functions used to send data from Databricks through serverless compute and event hub to Purview
    - Required adapter pattern to make emissions conform to Atlas
    - Challenges:
        - automating getting most recent OL jar into Databricks; created PR for this with emit script
        - needed to use API key passed in URL parameter; support for this integrated with PR
    - Have goal of extending use of OpenLineage inside of Spark further
    - Motivation: didn't want to be dependent on catalog API, particular flavor of Spark
    - Plans include other integrations, including dbt
    - Want to be respectful of OpenLineage's global scope, even if it means metadata on Purview side not real-time
    - Want to incorporate filtering capability, make it customizable based on particular connector
    - Interest extends beyond Databricks (e.g., Snowflake)
    - Eager to see issue #181 addressed: ability to tack on a MS jar to installation where OpenLineage is
    - Possible PR in future: emit metadata outside a run (e.g., as dataset facets); would meet need at MS

Logging backends [Julien]

- Open suggestion: add ability to send events to a logging aggregator (e.g., Datadog)
- Mandy: needed in addition to proxy backend?
- Proxy backend could be distribution endpoint, first location for this
- Use case: experimentation
- Proposed: open a ticket

Discussion

- Azure PRs, other merged PRs will be in 0.4

Nov 10th 2021 (9am PT)

Attendees:

- TSC
    - Mike Collado: Eng
    - Ryan Blue: Tabular, Apache Iceberg
    - Mandy Chessel: Lead Egeria project
    - Maciej Obuchowski: Eng GetInData, OpenLineage contributor
    - Willy Lulciuc: Co-creator of Marquez
    - Julien: OpenLineage Project lead
- And:
    - Michael Robinson: dev rel
    - Peter Hicks: Marquez contributor
    - Ross Turk: VP marketting Datakin
    - John Thomas: Support eng at Datakin
    - Minkyu Park: Dev at Datakin, learning about MQZ and OL.

Agenda:

- OL Client use cases for Apache Iceberg [Ryan]
- Proxy Backend and Egeria integration progress update (Issue #152) [Mandy]
- OpenLineage last release overview (0.3.1)
    - Facet versioning
    - Airflow 2 / Spark 3 support, dbt improvements
- OpenLineage 0.4 scope review
    - Proxy Backend (Issue #152)
    - Spark, Airflow, dbt improvements (documentation, coverage, ...)
    - improvements to the OpenLineage model
- Open discussion

Meeting recording:

Slides

Notes:

SPDX tags:
shorter license headers => makes things easier.
https://spdx.org/licenses/
TODO: Mandy will propose something next time

Iceberg requirements:

- ability for Iceberg to add facets without having to depend on the context it's running in.
- Avoid depending on allowing the Sources to expose facets in the Spark API as it would be a hard change to get into Spark.

Ryan:
Proposal to have a logger style API.

- similar to SLF4J or dropwizard metrics => Create a logging/metrics object. Independent of logging backend.
- Facets can be emitted and the backend can be configured independently whether those facets are picked up or not.

Example: Have an OpenLineage API to add facets in a given context:
create facet for some context: Read datasets x, ... write dataset Y

=> broad agreement on principle

Open Questions:

- when facets are sent?
    - preference to sending events as they go.
    - does that it fit with the OpenLineage view of the world? => yes
    - do we send them immediately? Do we wait?
    - iceberg not creating a facet until Spark asks for the splits
- Spark, bound to a context thread:
    - the "logger backend can grab the sql execution id"
    - loggers depend on thread
    - listener is on different thread
    - Report for a given job run
    - Ryan: runcontext is threadlocal: sets the executionid.
- The client side should be able to send an event immediately vs sent when you get a chance.
    - Who needs to do this?
    - Need to have a guide to defining a facet.
- Michael C.: TODO: Design Doc on logging
- Willy: Do we need a "RUNNING" event?

Flink:

- how to handle long running job
- [Ryan] [Mandy] long running jobs need to be defined
- TODO: Julien, post a ticket for long running jobs

Also need for OSS trino integration, tabular might contribute

Proxy Backend update [Mandy]

- draft PR #500: Thanks Willy for the initial setup.
  Looking for feedback
  Issues:
  Initial implementation was using the provided beans to deserialize but it didn't quite work (TODO: ticket)
  Instead just pass through. faster, but no validation
- OL is the dynamic lineage solution for Egeria
  used postman for 3rd party
  released in a few weeks
  https://odpi.github.io/egeria-docs/features/lineage-management/overview/#the-openlineage-standard
- proposal for new facets.
  RequestFacet => should be a runfacet, maps to the run args in Marquez

https://github.com/OpenLineage/OpenLineage/issues/256

Does the last version of a facet win? => yes
Need to document size constraint in OL (name length...) TODO: ticket

Oct 13th 2021

Attendees:

- TSC:
    - Michael Collado: Datakin
    - Julien Le Dem: OpenLineage Project Lead, Datakin
    - Maciej Obuchowski: GetInData, OpenLineage
    - Willy Lulciuc: Marquez, OpenLineage
    - Mandy Chessel: Egeria Project Lead, working on OpenLineage
- And:
    - Ross Turk: VP marketing at Datakin talk about the website
    - Minkyu Park: interested in contributing to Datakin
    - Peter Hicks: Marquez contributor, OpenLineage user

- Meeting recording:

Slides

- Notes:
  - OpenLineage website: https://openlineage.io/
    - Gatsby based (markdown) in OpenLineage/website repo
    - generates a static site hosted in github pages. OpenLineage/OpenLineage.github.io
    - deployment is currently manual. Automation in progress
    - Please open PRs on /website to contribute a blog posts.
      - Getting started with Egeria?
    - Suggestions:
      - Add page on open governance and how to join the project.
      - Add LFAI & data banner to the website?
      - Egeria is using MKdocs: very nice to navigate documentation.
  - upcoming 0.3.0:
    - Facet versioning:
      - each facet schema is versioned individually.
      - client/server code generation to facilitate producing/consuming openlineage events
    - Spark 3.x support
    - new mechanism for airflow 2.x
      - working with airflow maintainer to improve that.
  - Proxy Backend update (planned for OL 0.4.0):
    - mapping to egeria backend
    - planning to release for the Egeria webinar on the 8th of November
    - Willy provided a base module for ProxyBackend
  - Monthly release is a good cadence
  - Open discussions:
    - Azure purview team hackathon ongoing to consumer OpenLineage events
    - Design docs discussion:
      - proposal to add design doc for proposal.
      - goal:
        - Similar to the process of projects like Kafka, Flink: for specs and bigger features
        - not for bug fixes.
      - options:
        - proposal directory for docs as markdown
        - Open PRs against wiki pages: proposals wiki.
      - Manage status:
        - list of designs that are implemented vs pending.
        - table of open proposals.
      - vote for prioritization:
        - Every proposal design doc has an issue opened and link back to it.
      - good start for the blog talking about that feature
    - New committee on data ops: Mandy will be speaking about Egeria and OpenLineage
      - Scope:
        - How the foundation projects should work together around the topic.
        - Establish OpenLineage is important.
        - https://wiki.lfaidata.foundation/display/DL/DataOps+Committee

Sept 8th 2021

- Attendees:
  - TSC:
    - Mandy Chessell: Egeria Lead. Integrating OpenLineage in Egeria
    - Michael Collado: Datakin, OpenLineage
    - Maciej Obuchowski: GetInData. OpenLineage integrations
    - Willy Lulciuc: Marquez co-creator.
    - Ryan Blue: Tabular, Iceberg. Interested in collecting lineage across iceberg user with OpenLineage
  - And:
    - Venkatesh Tadinada: BMC workflow automation looking to integrate with Marquez
    - Minkyu Park: Datakin. learning about OpenLineage
    - Arthur Wiedmer: Apple, lineage for Siri and AI ML. Interested in implementing Marquez and OpenLineage

- Meeting recording:

- Meeting notes:
  - agenda:
    - Update on OpenLineage latest release (0.2.1)
      - dbt integration demo

        - OpenLineage 0.3 scope discussion
            - Facet versioning mechanism (Issue #153)
            - OpenLineage Proxy Backend (Issue #152)
            - OpenLineage implementer test data and validation
            - Kafka client
        - Roadmap
            - Iceberg integration
        - Open discussion
    - Slides
    - Discussions:
        - added to the agenda a Discussion of Iceberg requirements for OpenLineage.
    - Demo of dbt:
        - really easy to try
        - when running from airflow, we can use the wrapper 'dbt-ol run' instead of 'dbt run'
    - Presentation of Proxy Backend design:
        - summary of discussions in Egeria
            - Egeria is less interested in instances (runs) and will keep track of OpenLineage events separately as Operational lineage
            - Two ways to use Egeria with OpenLineage
                - receives HTTP events and forwards to Kafka
                - A consumer receives the Kafka events in Egeria
        - Proxy Backend in OpenLineage:
            - direct HTTP endpoint implementation in Egeria
        - Depending on the user they might pick one or the other and we'll document
    - Use a direct OpenLineage endpoint (like Marquez)
        - Deploy the Proxy Backend to write to a queue (ex: Kafka)
        - Follow up items:
    - The transport abstraction (Backend interface) could be usable directly from the client or from the Proxy Backend. The user can decide if they want the intermediate proxy. See #269
        - We should add a distribution client symmetric to the Proxy Backend. It reads from Kafka and sends event to an OpenLineage HTTP endpoint. Marquez would use it, for example to consume OpenLineage events produced by Egeria. See #270
- Iceberg integration:
    - presentation of Iceberg model
        - Manifest and manifest list: 2-level tree structure tracking data files.
        - root metadata version file. Points to manifest list (It knows all of the previous versions of the dataset that we want to keep)
    - Iceberg collect various metadata about the scans and data being produced and wants to expose it through OpenLineage. It can already expose metadata but there is no listener yet.
        - Ryan: added the metadata list presented to the Iceberg ticket: See #167

Aug 11th 2021

- Attendees:
    - TSC:
        - Ryan Blue
        - Maciej Obuchowski
        - Michael Collado
        - Daniel Henneberger
        - Willy Lulciuc
        - Mandy Chessell
        - Julien Le Dem
    - And:
        - Peter Hicks
        - Minkyu Park
        - Daniel Avancini
- Meeting recording:

- Meeting notes:
    - Agenda:
        - Coming in OpenLineage 0.1
            - OpenLineage spec versioning
            - Clients
            - Marquez integrations imported in OpenLineage
                - Apache Airflow:
                    - BigQuery
                    - Postgres
                    - Snowflake
                    - Redshift
                    - Great Expectations

        - Apache Spark
        - dbt
    - OpenLineage 0.2 scope discussion
        - Facet versioning mechanism (Issue #153)
        - OpenLineage Proxy Backend (Issue #152)
        - Kafka client
    - Roadmap
    - Open discussion
- Slides: https://docs.google.com/presentation/d/1Lxp2NB9xk8sTXOnT0_gTXicKX5FsktWa/edit#slide=id.ge80fbcb367_0_14
- Notes:
    - OpenLineage 0.1 is being published
    - Coming in OpenLineage 0.1
        - OpenLineage spec versioning
        - Clients (Java, Python)
        - Marquez integrations imported in OpenLineage
            - Apache Airflow:
                - BigQuery
                - Postgres
                - Snowflake
                - Redshift
                - Great Expectations
            - Apache Spark
            - dbt
            - Question: How is airflow capturing openlineage events?
                - openlineage-airflow installed on the airflow instance
                - adapters per operator
    - OpenLineage 0.2 scope discussion
        - Facet versioning mechanism (Issue #153)
        - OpenLineage Proxy Backend (Issue #152)
            - Questions:
                - What is the advantage of the proxy backend?
                    - The consumer does not need to implement an endpoint and can consume from kafka
                    - can configure what to do with events independently of various integrations
                    - first step to having a routing mechanism:
                        - to send events to multiple consumer
                        - to have rule-based routing
                        - to enable archiving the event in addition to sending them
                - Is it included in OpenLineage?
                    - Yes (Otherwise it would have to be in Egeria)
                - Does it include error management or retry policy? What if the proxy dies? Do we care about durability?
                    - Yes we care about durability
                    - first implementation to be synchronous. single transaction to Kafka per event.
                    - future might be configurable to adjust depending on context (guaranteed delivery vs performance batching)
                - What technology should we use?
                    - Proposed: Java + spring boot (like Egeria)
                    - discussion to use Java + dropwizard like Marquez
                    - general consensus on using java. (framework TBD)
                    - In the future, might have a go implementation to enable lightweight sidecar pattern
        - Kafka client
    - Roadmap
        - https://github.com/OpenLineage/OpenLineage/projects
    - Open discussion
        - How do we define extension points for integrations? For example hooks, spark and airflow for the user to add adapters /facets without having to modify OL.
            - TODO: create a ticket to track this
        - Apache Iceberg interest in OpenLineage:
            - Would want to add additional notifications
                - how many files read or written
                - How long a commit took.
                - How many attempts to commit were needed?
            - TODO: create ticket to enable Iceberg facets to be added to OpenLineage events
                - => #166 Enabling facets to be added from Iceberg
            - Iceberg needs to send events independently of where the library is used. (example: plain java process or other)
                - TODO: need ticket for this => #167 Iceberg integration
            - TODO: ticket for PrestoDB/Trino integrations
                - => #164 Trino and #165 PrestoDB
        - Egeria has a weekly community call
            - September 1st will be about OpenLineage
            - Also an incoming webinar

July 14th 2021

- Attendees:
    - TSC:

        - Julien Le Dem
        - Mandy Chessel
        - Michael Collado
        - Willy Lulciuc
- Meeting recording:

- Meeting notes
    - Agenda:
        - Finalize the OpenLineage Mission Statement
        - Review OpenLineage 0.1 scope
        - Roadmap
        - Open discussion
        - Slides: https://docs.google.com/presentation/d/1fD_TBUykuAbOqm51Idn7GeGqDnuhSd7f/edit#slide=id.ge4b57c6942_0_46
    - Notes:
      Mission statement:

        - https://github.com/OpenLineage/OpenLineage/issues/84
        - Overall consensus on the statement.
        - TODO: vote by commenting on the ticket

      Spec versioning mechanism:

        - The goal is to commit to compatible changes once 0.1 is published
        - We need a follow up to separate core facet versioning

      => TODO: create a separate github ticket.
        - The lineage event should have a field that identifies what version of the spec it was produced with
            - => TODO: create a github issue for this
        - TODO: Add issue to document version number semantics (SCHEMAVER)

      Extend Event State notion:

        - where do we capture more precise state transitions like RESTART?
            - Discussion should happen here: https://github.com/OpenLineage/OpenLineage/issues/9

      OpenLineage 0.1:

        - finalize a few spec details for 0.1 : a few items left to discuss.
            - In particular job naming
            - parent job model
        - Importing Marquez integrations in OpenLineage

      Open Discussion:

        - connecting the consumer and producer

            - TODO: ticket to track distribution mechanism
            - options:

                - Would we need a consumption client to make it easy for consumers to get events from Kafka for example?
                - OpenLineage provides client libraries to serialize/deserialize events as well as sending them.
                - proxy similar to OpenTelemetry Collector.
                - Send to Kafka: https://github.com/OpenLineage/OpenLineage/issues/70
            - We can have documentation on how to send to backends that are not Marquez using HTTP and existing gateway mechanism to queues.
            - Do we have a mutual third party or the client know where to send?
        - Source code location finalization
        - job naming convention

            - you don't always have a nested execution
                - can call a parent
            - parent job
            - You can have a job calling another one.
            - always distinguish a job and its run
        - need a separate notion for job dependencies
        - need to capture event driven: TODO: create ticket.

      TODO(Julien): update job naming ticket to have the discussion.

June 9th 2021

- Attendees:
    - TSC:
      Julien Le Dem: Marquez, Datakin
      Drew Banin: dbt, CPO at fishtown analytics
      Maciej Obuchowski: Marquez, GetIndata consulting company
      Zhamak Dehghani: Datamesh, Open protocol of observability for data ecosystem is a big piece of Datamesh
      Daniel Henneberger: building a database, interested in lineage
      Mandy Chessel: Lead of Egeria, metadata exchange. lineage is a great extension that volunteers lineage

Willy Lulciuc: co-creator of Marquez
Michael Collado: Datakin, OpenLineage end-to-end holistic approach.
- And:
  Kedar Rajwade: consulting on distributed systems.
  Barr Yaron: dbt, PM at Fishtown analytics on metadata.
  Victor Shafran: co-founder at databand.ai pipeline monitoring company. lineage is a common issue
- Excused: Ryan Blue, James Campbell

- Meeting recording:

- Meeting notes:

  Agenda:

  - project communication
  - Technical charter review
  - medium term roadmap discussion

  Notes:

  - project communication
    - github: for specs, designs, reviews and building consensus (issues and PRs)
    - email: for announcements, notes, etc
    - Slack: transient discussions, does not maintain history. Any decision making or notes should go to persistent medium (email and github)
    - monthly meeting: recorded, notes and recording published on the wiki
  - Technical Charter review:
    - https://docs.google.com/document/d/11xo2cPtuYHmqRLnR-vt9ln4GToe0y60H/edit#
    - TODO: Finalize the mission statement. TSC members to comment in the doc.
  - Roadmap discussion:
    - https://docs.google.com/document/d/1ANXLKON3TN55XuNxYuTWe_CusfV66Gh0FyN8C2x9ayA/edit
    - TODO: please comment in the doc. Julien to update the OpenLineage project in github: https://github.com/OpenLineage/OpenLineage/projects/1