

# A New SQLAlchemyCollector for Emitting Airflow Lineage as DAGs Run

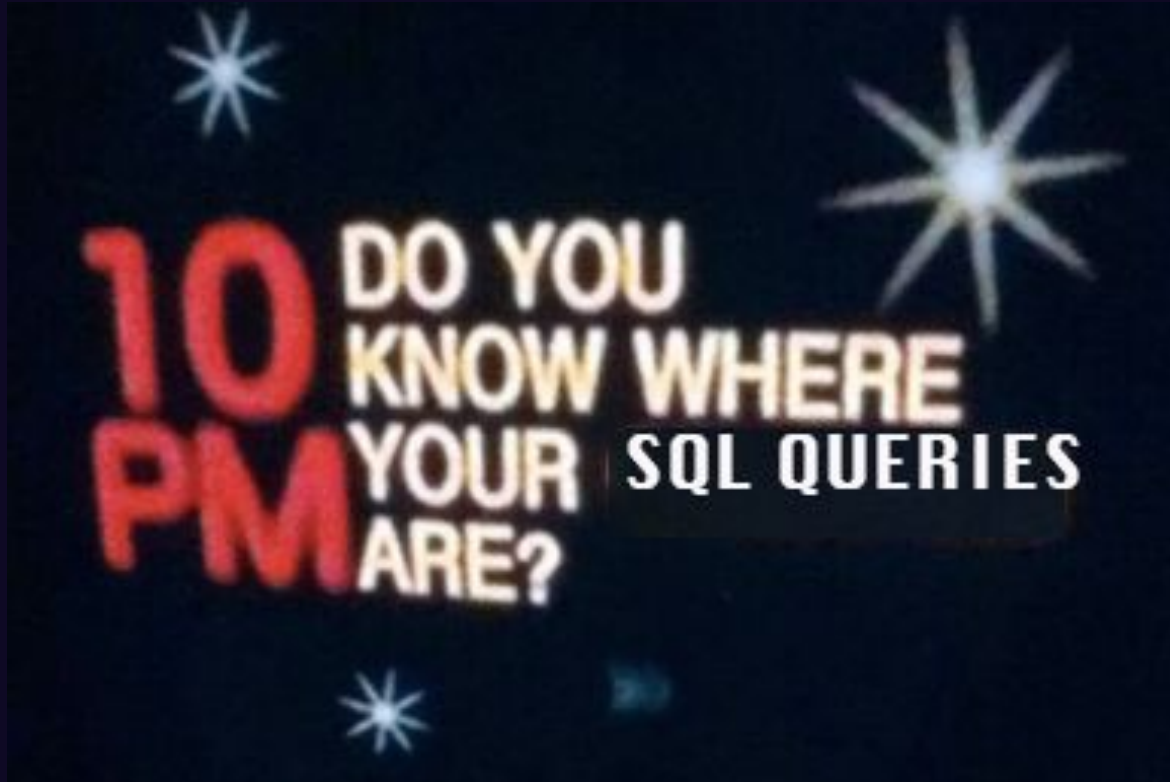
Michael Robinson, Community Manager at Astronomer

## Michael Robinson



- Email: [michael.robinson@astronomer.io](mailto:michael.robinson@astronomer.io)
- LinkedIn: <https://www.linkedin.com/in/michael-robinson/>
- GitHub: <https://github.com/merobi-hub>
- Project PR: <https://github.com/OpenLineage/OpenLineage/pull/2088>

1. Why build a SQLAlchemy integration in OpenLineage?
2. SQLAlchemy
  - Background & design
3. SQLAlchemy ORM
  - Available metadata from database operations
4. OpenLineage-SQLAlchemy Integration Prototype
  - Design
  - Use cases
  - Implementation
5. Integration Output Examples



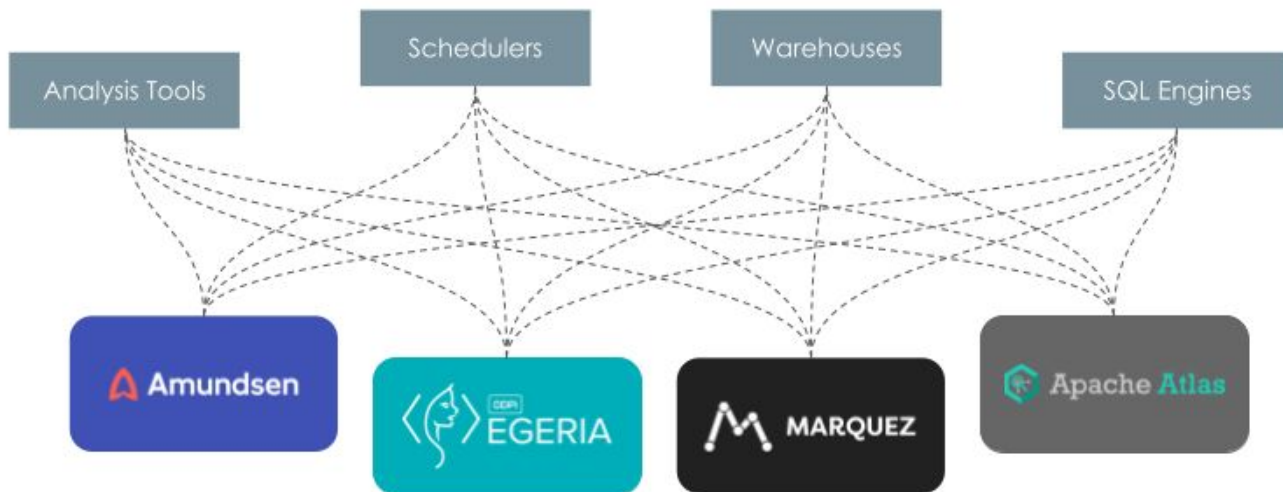
# OpenLineage

## Mission

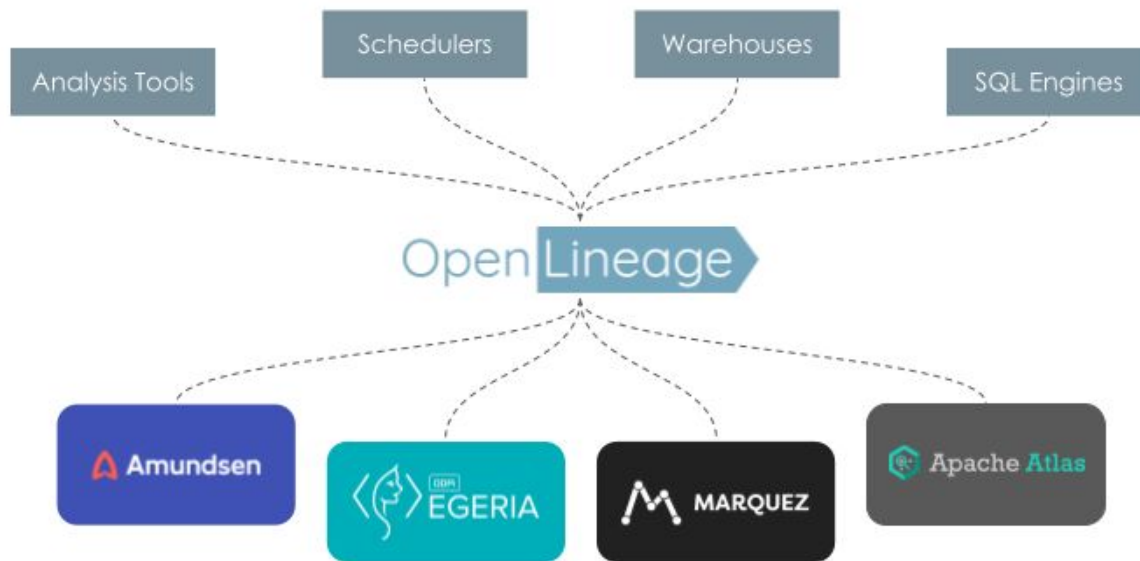
To define an **open standard** for the collection of lineage metadata from pipelines **as they are running**.



# Before OpenLineage



# With OpenLineage



# SQLAlchemy





# SQLAlchemy

- First release: **2/6/2006**
- Latest release: **20.0.20, 8/15/2023**
- License: **MIT**
- Closed issues: **7266**
- Closed PRs: **652**
- Stars: **7.7k**

*SQLAlchemy is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL.*

*It provides a full suite of well known enterprise-level persistence patterns, designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language.*

# SQLAlchemy

- **Architecture:** Core + separate ORM
- **Guiding philosophy:**
  - expose the “R” in ORM
  - support abstraction but don’t enable mystification
  - the ORM is intentionally redundant for most use cases

*A relational database provides **rich, set-based functionality that should be fully exposed**. SQLAlchemy’s ORM provides an open-ended set of patterns that allow a developer to construct **a custom mediation layer** between a domain model and a relational schema, turning the so-called “object relational impedance” issue into a distant memory.*

*With SQLAlchemy, there’s no such thing as “the ORM generated a bad query” - **you retain full control over the structure of queries**, including how joins are organized, how subqueries and correlation is used, what columns are requested. **Everything SQLAlchemy does is ultimately the result of a developer-initiated decision.***

***Don’t use an ORM if the problem doesn’t need one.** SQLAlchemy consists of a Core and separate ORM component. The Core offers a full SQL expression language that allows Pythonic construction of SQL constructs that render directly to SQL strings for a target database, returning result sets that are essentially enhanced DBAPI cursors.*

# SQLAlchemy ORM



- Since SQLAlchemy 1.4, all ORM (Object Relational Mapper) mappings derive from a registry of mapped classes.
- A common use case for the ORM is automating database operations.
- The ORM also offers event tracking capability through session **events**:
  - these Object Lifecycle Events track when objects are **added**, **deleted** or **persisted** in sessions.

## Object Lifecycle Events

- `after_bulk_update`
- `after_bulk_delete`
- `after_commit`
- `before_commit`
- `after_insert`
- ...

```
from sqlalchemy import event

@event.listens_for(SomeSessionClassOrObject, 'after_commit')
def receive_after_commit(session):
    "listen for the 'after_commit' event"

    # ... (event handling logic) ...
```

## Some of the ORM metadata available

The ORM's `execute_state` offers multiple hooks containing columns:

- `all_orm_descriptors.items()`
- `attrs.items()`
- `column_attrs.items()`
- `columns`

queries:

- `statement`

tables:

- `tables`

# OpenLineage- SQLAlchemy Prototype



**SQLAlchemyCollector** and **OpenLineageAdapter** classes for:

1. listening for SQLAlchemy events using the SQLAlchemy ORM module
  - `orm_execute_state.all_mappers` include:
    - `columns` object
    - `tables` object
2. adapting the events to the OpenLineage spec with the OpenLineage Common Integration

## OpenLineage Classes

- Dataset
- Run
- RunEvent
- RunState
- Job
- SqlJobFacet

## SQLAlchemy Hooks

`inspect()` allows us to get tables and columns from `orm.execute.state.all_mappers`

- **Apache Airflow** testing and debugging
  - Metadata about internal database operations on xcom, task\_instance, dag, and additional datasets
- **Web application** design and development
  - Web developer: new persona for OpenLineage+Marquez
  - Debugging: diagnose broken apps, identify compromised datasets, jobs
  - Database design? (enabled by static lineage capability, new in OpenLineage 1.0)



## Flask

A micro Web framework written in Python.  
Flask-SQLAlchemy is a commonly used backend.



## Bottle

A lightweight, simple WSGI framework in Python.  
Reportedly plays well with SQLAlchemy.

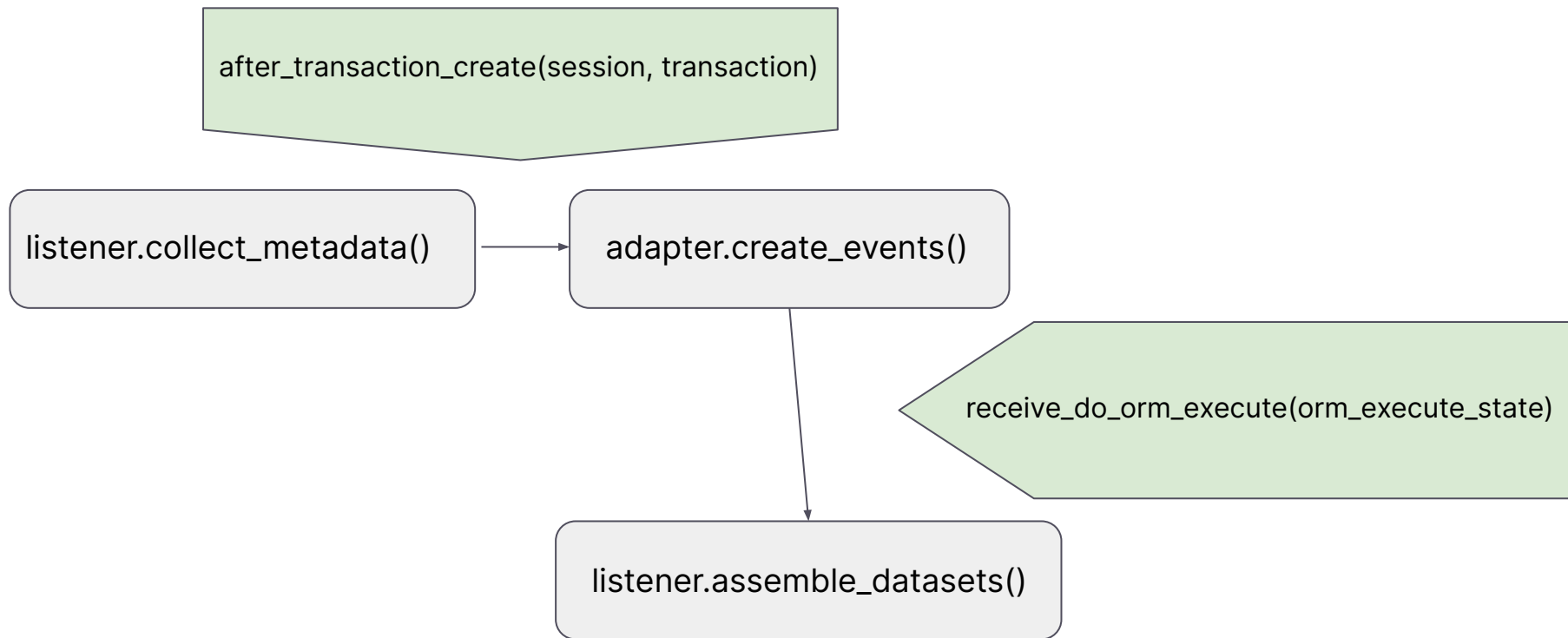


## Apache Airflow

Airflow™ is a platform to programmatically author, schedule and monitor workflows.



- Developing, testing and debugging Apache Airflow internals
  - Start time, completion time, SQL queries, runstate, more
  - Lineage of datasets and jobs (WIP)
  - Metadata emitted about:
    - xcom
    - task\_instance
    - rendered\_task\_instance
    - dag
    - serialized\_dag
    - dag\_owner\_attributes
    - dag\_run
    - dag\_warning
    - import\_error
    - job
    - log\_template
    - slot\_pool



Airflow runtime metadata available from the integration include **rendered SQL queries**:

```
SELECT task_instance.try_number AS task_instance_try_number, task_instance.task_id AS task_instance_task_id, task_instance.dag_id AS task_instance_dag_id, task_instance.run_id AS task_instance_run_id, task_instance.map_index AS task_instance_map_index, task_instance.start_date AS task_instance_start_date, task_instance.end_date AS task_instance_end_date, task_instance.duration AS task_instance_duration, task_instance.state AS task_instance_state, task_instance.max_tries AS task_instance_max_tries, task_instance.hostname AS task_instance_hostname, task_instance.unixname AS task_instance_unixname, task_instance.job_id AS task_instance_job_id, task_instance.pool AS task_instance_pool, task_instance.pool_slots AS task_instance_pool_slots, task_instance.queue AS task_instance_queue, task_instance.priority_weight AS task_instance_priority_weight, task_instance.operator AS task_instance_operator, task_instance.queued_dttm AS task_instance_queued_dttm, task_instance.queued_by_job_id AS task_instance_queued_by_job_id, task_instance.pid AS task_instance_pid, task_instance.executor_config AS task_instance_executor_config, task_instance.updated_at AS task_instance_updated_at, task_instance.external_executor_id AS task_instance_external_executor_id, task_instance.trigger_id AS task_instance_trigger_id, task_instance.trigger_timeout AS task_instance_trigger_timeout, task_instance.next_method AS task_instance_next_method, task_instance.next_kwargs AS task_instance_next_kwargs, dag_run_1.state AS dag_run_1_state, dag_run_1.id AS dag_run_1_id, dag_run_1.dag_id AS dag_run_1_dag_id, dag_run_1.queued_at AS dag_run_1_queued_at, dag_run_1.execution_date AS dag_run_1_execution_date, dag_run_1.start_date AS dag_run_1_start_date, dag_run_1.start_date AS dag_run_1_start_date, dag_run_1.end_date AS dag_run_1_end_date, dag_run_1.run_id AS dag_run_1_run_id, dag_run_1.creating_job_id AS dag_run_1_creating_job_id, dag_run_1.external_trigger AS dag_run_1_external_trigger, dag_run_1.run_type AS dag_run_1_run_type, dag_run_1.conf AS dag_run_1_conf, dag_run_1.data_interval_start AS dag_run_1_data_interval_start, dag_run_1.data_interval_end AS dag_run_1_data_interval_end, dag_run_1.last_scheduling_decision AS dag_run_1_last_scheduling_decision, dag_run_1.dag_hash AS dag_run_1_dag_hash, dag_run_1.log_template_id AS dag_run_1_log_template_id, dag_run_1.updated_at AS dag_run_1_updated_at ...  
FROM task_instance JOIN dag_run AS dag_run_1 ON dag_run_1.dag_id = task_instance.dag_id AND dag_run_1.run_id = task_instance.run_id  
WHERE task_instance.dag_id = :dag_id_1 AND task_instance.run_id = :run_id_1 AND task_instance.task_id = :task_id_1 AND task_instance.map_index = :map_index_1
```



**task\_instance**

## log\_template

```
SELECT log_template.id AS log_template_id, log_template.filename AS log_template_filename, log_template.elasticsearch_id AS log_template_elasticsearch_id, log_template.created_at AS log_template_created_at FROM log_template WHERE log_template.id = :pk_1
```

## dag\_run

```
SELECT dag_run.state AS dag_run_state, dag_run.id AS dag_run_id, dag_run.dag_id AS dag_run_dag_id, dag_run.queued_at AS dag_run_queued_at, dag_run.execution_date AS dag_run_execution_date, dag_run.start_date AS dag_run_start_date, dag_run.end_date AS dag_run_end_date, dag_run.run_id AS dag_run_run_id, dag_run.creating_job_id AS dag_run_creating_job_id, dag_run.external_trigger AS dag_run_external_trigger, dag_run.run_type AS dag_run_run_type, dag_run.conf AS dag_run_conf, dag_run.data_interval_start AS dag_run_data_interval_start, dag_run.data_interval_end AS dag_run_data_interval_end, dag_run.last_scheduling_decision AS dag_run_last_scheduling_decision, dag_run.dag_hash AS dag_run_dag_hash, dag_run.log_template_id AS dag_run_log_template_id, dag_run.updated_at AS dag_run_updated_at FROM dag_run WHERE dag_run.dag_id = :dag_id_1 AND dag_run.execution_date < :execution_date_1 AND dag_run.state = :state_1 ORDER BY dag_run.execution_date DESC LIMIT :param_1
```

## LogTemplate dataset columns

```
[('id', <ColumnProperty at 0xfffffa0766a40; id>), ('filename', <ColumnProperty at 0xfffffa05be640; filename>), ('elasticsearch_id', <ColumnProperty at 0xfffffa05be740; elasticsearch_id>), ('created_at', <ColumnProperty at 0xfffffa05be840; created_at>)]
```

## DagRun dataset columns

```
[('_state', <ColumnProperty at 0xfffffa05be940; _state>), ('task_instances', <RelationshipProperty at 0xfffffa06884c0; task_instances>), ('dag_model', <RelationshipProperty at 0xfffffa0688540; dag_model>), ('dag_run_note', <RelationshipProperty at 0xfffffa06885c0; dag_run_note>), ('state', <SynonymProperty at 0xfffffa05c29a0; state>), ('id', <ColumnProperty at 0xfffffa05bea40; id>), ('dag_id', <ColumnProperty at 0xfffffa05beb40; dag_id>), ('queued_at', <ColumnProperty at 0xfffffa05bec40; queued_at>), ('execution_date', <ColumnProperty at 0xfffffa05bed40; execution_date>), ('start_date', <ColumnProperty at 0xfffffa05bee40; start_date>), ('end_date', <ColumnProperty at 0xfffffa069c040; end_date>), ('run_id', <ColumnProperty at 0xfffffa069c140; run_id>), ('creating_job_id', <ColumnProperty at 0xfffffa069c240; creating_job_id>), ('external_trigger', <ColumnProperty at 0xfffffa069c340; external_trigger>), ('run_type', <ColumnProperty at 0xfffffa069c440; run_type>), ('conf', <ColumnProperty at 0xfffffa069c540; conf>), ('data_interval_start', <ColumnProperty at 0xfffffa069c640; data_interval_start>), ('data_interval_end', <ColumnProperty at 0xfffffa069c740; data_interval_end>), ('last_scheduling_decision', <ColumnProperty at 0xfffffa069c840; last_scheduling_decision>), ('dag_hash', <ColumnProperty at 0xfffffa069c940; dag_hash>), ('log_template_id', <ColumnProperty at 0xfffffa069ca40; log_template_id>), ('updated_at', <ColumnProperty at 0xfffffa069cb40; updated_at>), ('consumed_dataset_events', <RelationshipProperty at 0xffff9e2d49c0; consumed_dataset_events>), ('creating_job', <RelationshipProperty at 0xffff9e21fb40; creating_job>), ('serialized_dag', <RelationshipProperty at 0xffff9e21fbc0; serialized_dag>)]
```



Search Jobs and Datasets

ns airflow

[API Docs](#)






## DATASETS


Page: 1 (1 - 13)




NAME	NAMESPACE	SOURCE	UPDATED AT	STATUS
dag	airflow	airflow	Sep 06, 2023 07:33am	N/A
dag_code	airflow	airflow	Sep 06, 2023 07:33am	N/A
dag_owner_attributes	airflow	airflow	Sep 06, 2023 07:33am	N/A
dag_run	airflow	airflow	Sep 06, 2023 07:33am	N/A
dag_warning	airflow	airflow	Sep 06, 2023 07:33am	N/A
import_error	airflow	airflow	Sep 06, 2023 07:33am	N/A
job	airflow	airflow	Sep 06, 2023 07:33am	N/A
log_template	airflow	airflow	Sep 06, 2023 07:33am	N/A
rendered_task_instance_fields	airflow	airflow	Sep 06, 2023 07:33am	N/A
serialized_dag	airflow	airflow	Sep 06, 2023 07:33am	N/A
slot_pool	airflow	airflow	Sep 06, 2023 07:33am	N/A
task_instance	airflow	airflow	Sep 06, 2023 07:33am	N/A
xcom	airflow	airflow	Sep 06, 2023 07:33am	N/A





 **MARQUEZ**

ns airflow API Docs

 dag

Graph Depth

LATEST SCHEMA VERSION HISTORY COLUMN LINEAGE

DELETE X

### dag

NAME	TYPE	DESCRIPTION
dag_id	VARCHAR(250)	no description
root_dag_id	VARCHAR(250)	no description
is_paused	BOOLEAN	no description
is_subdag	BOOLEAN	no description
is_active	BOOLEAN	no description
last_parsed_time	TIMESTAMP	no description
last_pickled	TIMESTAMP	no description
last_expired	TIMESTAMP	no description
scheduler_lock	BOOLEAN	no description
pickle_id	INTEGER	no description
fileloc	VARCHAR(2000)	no description
processor_subdir	VARCHAR(2000)	no description
owners	VARCHAR(2000)	no description



Search Jobs and Datasets

ns airflow

API Docs



ID	STATE	NAME	NAMESPACE	TIME
950e3521-95ef-4a93-be67-ad10fa720197	COMPLETE	airflow.airflow.SELECT dag_warning.dag_id AS dag_warning_dag_id, dag_warning.warning_type AS dag_warning_warning_type, dag_warning.message AS dag_warning_message, dag_warning.timestamp AS dag_warning_timestamp FROM dag_warning WHERE dag_warning.dag_id IN (__[POSTCOMPILE_dag_id_1])	airflow	Sep 06, 2023 07:37am

```
"root" : { 8 items
  "eventType" : "COMPLETE"
  "eventTime" : "2023-09-06T11:37:26.88774Z"
  "run" : { 2 items
    "runId" : "950e3521-95ef-4a93-be67-ad10fa720197"
    "facets" : {...} 2 items
  }
  "job" : { 3 items
    "namespace" : "airflow"
    "name" :
      "airflow.airflow.SELECT dag_warning.dag_id AS dag_warning_dag_id, dag_warning.warning_type AS dag_warning_warning_type,
      dag_warning.message AS dag_warning_message, dag_warning.timestamp AS dag_warning_timestamp FROM dag_warning WHERE dag_warning.dag_id IN
      (__[POSTCOMPILE_dag_id_1])"
    "facets" : {...} 3 items
  }
  "inputs" : [ 1 item
    0 : { 5 items
      "namespace" : "airflow"
      "name" : "dag_warning"
      "facets" : {...} 7 items
      "inputFacets" : NULL
      "outputFacets" : NULL
    }
  ]
}
```





MARQUEZ



Search **Jobs** and **Datasets**




ns airflow




API Docs




```
"sourceCodeLocation" : NULL
  "sql" : { 3 items
    "_producer" : https://github.com/OpenLineage/OpenLineage/tree/0.29.2/integration/airflow
    "_schemaURL" : https://raw.githubusercontent.com/OpenLineage/OpenLineage/main/spec/OpenLineage.json#/definitions/SqlJobFacet
    "query" :
      "SELECT dag_warning.dag_id AS dag_warning_dag_id, dag_warning.warning_type AS dag_warning_warning_type,
      dag_warning.message AS dag_warning_message, dag_warning.timestamp AS dag_warning_timestamp FROM dag_warning WHERE
      dag_warning.dag_id IN (__[POSTCOMPILE_dag_id_1])"
  }
}
}
}
"inputs" : [ 1 item
  0 : { 5 items
    "namespace" : "airflow"
    "name" : "dag_warning"
    "facets" : {...} 7 items
    "inputFacets" : NULL
    "outputFacets" : NULL
  }
]
```



 **MARQUEZ**

ns library API Docs



user → library.bo...inventory

Depth

Full

LATEST RUN RUN HISTORY DELETE LOCATION ×

● library.book\_inventory

```
SELECT "user".id AS user_id, "user".email AS user_email, "user".password AS user_password
FROM "user"
WHERE "user".id = :pk_1
```

Sep 03, 2023 07:54am

FACETS

en



Search Jobs and Datasets

ns library

[API Docs](#)






## DATASETS


Page: 1 (1 - 4)



NAME	NAMESPACE	SOURCE	UPDATED AT	STATUS
<a href="#">book</a>	library	library	Sep 03, 2023 07:57am	N/A
<a href="#">book_history</a>	library	library	Sep 03, 2023 07:57am	N/A
<a href="#">post</a>	library	library	Sep 03, 2023 07:58am	N/A
<a href="#">user</a>	library	library	Sep 03, 2023 07:57am	N/A

en





Search **Jobs** and **Datasets**

ns library API Docs

Depth   
Full

**LATEST SCHEMA** | VERSION HISTORY | COLUMN LINEAGE

**DELETE** X

### book

NAME	TYPE	DESCRIPTION
id	VARCHAR	no description
author	VARCHAR(150)	no description
title	VARCHAR(300)	no description
publisher	VARCHAR(150)	no description
description	VARCHAR(500)	no description
genre	VARCHAR(50)	no description
image	VARCHAR(200)	no description
pub_date	VARCHAR(10)	no description
more_info	VARCHAR(200)	no description
user_id	VARCHAR	no description

en

FACETS



Search Jobs and Datasets

ns library

API Docs



post

Depth 5

Full

LATEST SCHEMA

VERSION HISTORY

COLUMN LINEAGE

DELETE



post

NAME	TYPE	DESCRIPTION
id	VARCHAR	no description
title	VARCHAR(100)	no description
content	VARCHAR(300)	no description
date_created	DATETIME	no description
email	VARCHAR	no description

FACETS




Search


▼ root: {} 2 keys

▶ schema: {} 3 keys

▶ dataSource: {} 4 keys

en





Search **Jobs** and **Datasets**

ns library API Docs

Depth  Full

[LATEST SCHEMA](#) [VERSION HISTORY](#) [COLUMN LINEAGE](#) DELETE ×

### user

NAME	TYPE	DESCRIPTION
id	VARCHAR	no description
email	VARCHAR(150)	no description
password	VARCHAR	no description

**FACETS**

Search

- ▼ root: {} 2 keys
  - ▶ schema: {} 3 keys
  - ▶ dataSource: {} 4 keys

en

## Michael Robinson

- Community Manager, OpenLineage+Marquez Communities, Astronomer
- OpenLineage+Marquez committer
- Airflow contributor
- GitHub: <https://github.com/merobi-hub>

For more information about the integration, see the PR in OpenLineage:

- Project PR: <https://github.com/OpenLineage/OpenLineage/pull/2088>



ASTRONOMER

# (After) Party Under the Stars

**Wednesday, September 20th**

6:30pm-10:00pm

**The Sheraton Centre**

123 Queen St W

(7 min walk)



RSVP Now



Let's flow together

**Workshop**

# Get Airflow Certified

**Thursday, September 21st**

12:00 pm in Trinity 4

Marc Lamberti

Head of Customer Education  
at Astronomer

