LEGACY POWERS LEGENDARY

Apache Superset Technical Whitepaper

Contents

1 Configuring Apache Superset.. 2

1 Configuring Apache Superset

This blueprint describes how to deploy Apache Superset as a Docker image and connect it to databases like PostgreSQL, Oracle, and SQL Server.

Event Analytics service writes consumed audit events to database tables as JSON data. Customers can easily integrate with third-party Business Intelligence tool like Apache Superset that can analyze JSON data and generate good visualizations to showcase the analysis on audit events. The following sections describe the integration with Apache Superset.

Overview

Apache Superset is an open-source data visualization tool that can be used to access auditing data captured by the Rocket Audit and Enterprise Search Service. The following sections describe how to deploy Apache Superset as a Docker image and connect it to databases like PostgreSQL, Oracle, and SQL Server. Information relating to the Docker image can be found in docker hub at Docker Hub documentation on Apache Superset. Further information relating to Apache Superset can be found at Apache Superset/.

Prerequisites

1. Deploy Superset and start a Superset instance as explained in Docker Hub documentation on Apache Superset.
2. Install database drivers, if required:

   Oracle

   a. Install Oracle driver. Superset requires cx_Oracle driver for establishing the connection to Oracle database and the following command installs the driver.

```
docker exec -it superset pip install cx_Oracle
```

   b. Oracle driver requires Oracle Client and other libraries to be installed. The free Oracle Instant Client “Basic” can be downloaded from Oracle Instant Client Downloads for Linux x86-64 (64-bit). Download Basic Package (ZIP) file, `instantclient-basic-linux.x64-21.6.0.0.0dbru.zip`(the zip file name may be different based on the latest version available). Once downloaded, copy the zip file to the docker container using the following command:

```
docker cp instantclient-basic-linux.x64-21.6.0.0.0dbru.zip superset:/opt
```

   c. Create a `/opt/oracleclient` directory in the docker container for installing the client library using the following command:

```
docker exec -it -u root superset mkdir /opt/oracleclient
```

   d. Unzip the client library contents into the `/opt/oracleclient` directory using the following command:

```
docker exec -it -u root superset unzip -j /opt/instantclient-basic-
linux.x64-21.6.0.0.0dbru.zip -d /opt/oracleclient
```

e. Install libaio library using the following commands:

```
docker exec -it -u root superset apt-get update
docker exec -it -u root superset apt-get install libaio1 libaio-dev
```

SQL Server

a. Install SQL Server driver. Superset requires pymssql driver for establishing the connection to SQL Server database and the following command installs the driver:

```
docker exec -it superset pip install pymssql
```

3. Initialize local Superset instance as explained in Docker Hub documentation on Apache Super. Follow steps to create local admin account, migrate local database to latest and set up roles.

Connecting Superset to Audit and Enterprise Search database

1. Log in to Superset.
2. Navigate to the **Data** > **Databases** and create a new database connection.
3. Click on **+Database** button.

   PostgreSQL

   a. Select the PostgreSQL icon.
   b. Enter the hostname of the deployed Audit Service, PORT: 2101, Database: eventanalytics, Username: mobius, Password: postgres, Display Name: Demo Database.
   c. Click on **Connect** button.

   Oracle

   a. Select Oracle from the **Choose a Database**drop-down list.
   b. Provide the correct SQL ALCHEMY URI using the syntax `oracle://username:password@host:port/database`
   c. Click on **TEST CONNECTION** button.
   d. Once successful, click on **Connect**button.

   SQL Server

   a. Select Microsoft SQL Server from the **Choose a Database** drop-down list.
   b. Provide the correct SQL ALCHEMY URI using the syntax `mssql+pymssql://username:password@host:port/database`.
   c. Click on **TEST CONNECTION** button.
   d. Once successful, click on **Connect**button.
4. Navigate to the **SQL Lab** tab and check **Expose database in SQL Lab**, **Allow Create VIEW AS**,

and **Allow DML** check boxes.

5. Navigate to the security section and check the **Allow Data Upload**check box and click on **Finish** button.

Creating view to access data

To work more conveniently in Superset, a view will be created to map JSONB field data items to regular relational columns. These columns will be referenced in charts instead of the JSONB field format.

1. Navigate to **SQL Lab** > **SQL Editor**.
2. Select the Audit and Enterprise Search database and run the following SQL script to create the views. Choose the script corresponding to your database engine.
   PostgreSQL

```
-- View: public.supersetbetaview

-- DROP VIEW public.supersetbetaview;

CREATE OR REPLACE VIEW public.supersetbetaview
AS
SELECT events.data ->> 'id'::text AS id,
   (events.data ->> 'timestamp'::text)::timestamp without time zone AS
 timestamp,
  events.data ->> 'type'::text AS type,
  events.data ->> 'user'::text AS user,
  events.data ->> 'action'::text AS action,
  events.data ->> 'description'::text AS description,
   ((events.data ->> 'mobius'::text)::json) ->> 'server'::text AS
 mobius_server,
   ((events.data ->> 'mobius'::text)::json) ->> 'recipient'::text AS
 mobius_recipient,
   ((events.data ->> 'mobius'::text)::json) ->> 'section'::text AS
 mobius_section,
   ((events.data ->> 'mobius'::text)::json) ->> 'contentClass'::text AS
 mobius_contentclass,
   ((events.data ->> 'mobius'::text)::json) ->> 'contentIngestion'::text
 AS mobius_contentingestion
FROM events;

ALTER TABLE public.supersetbetaview
  OWNER TO mobius;

-- View: public.supersetbetadocumentview
-- DROP VIEW public.supersetbetadocumentview;

CREATE OR REPLACE VIEW public.supersetbetadocumentview
AS
SELECT documentevents.data ->> 'id'::text AS id,
  (documentevents.data ->> 'timestamp'::text)::timestamp without time
 zone AS timestamp,
  documentevents.data ->> 'type'::text AS type,
  documentevents.data ->> 'user'::text AS user,
  documentevents.data ->> 'action'::text AS action,
  documentevents.data ->> 'description'::text AS description,
  ((documentevents.data ->> 'mobius'::text)::json) ->> 'server'::text AS
 mobius_server,
  ((documentevents.data ->> 'mobius'::text)::json) ->> 'recipient'::text
 AS mobius_recipient,
  ((documentevents.data ->> 'mobius'::text)::json) ->> 'section'::text
 AS mobius_section,
  ((documentevents.data ->> 'mobius'::text)::json) ->>
 'contentClass'::text AS mobius_contentclass,
  ((documentevents.data ->> 'mobius'::text)::json) ->>
 'contentIngestion'::text AS   mobius_contentingestion,
  documentevents.documentdata ->> 'page_count'::text AS pgcount,
  documentevents.documentdata ->> 'page_of_doc'::text AS docpage,
  documentevents.documentdata ->> 'section_id'::text AS docname,
  documentevents.documentdata ->> 'ProcessID'::text AS processid
  FROM documentevents;

ALTER TABLE public.supersetbetadocumentview
  OWNER TO mobius;
```

SQL Server

```
-- View: dbo.supersetbetaview
CREATE VIEW dbo.supersetbetaview AS
  SELECT JSON_VALUE(data, '$.id') AS id,
    JSON_VALUE(data, '$.timestamp') AS timestamp,
    JSON_VALUE(data, '$.type') AS type,
    JSON_VALUE(data, '$.user') AS userid,
    JSON_VALUE(data, '$.action') AS action,
    JSON_VALUE(data, '$.description') AS description,
    JSON_VALUE(data, '$.mobius.server') AS mobius_server,
    JSON_VALUE(data, '$.mobius.recipient') AS mobius_recipient,
    JSON_VALUE(data, '$.mobius.section') AS mobius_section,
    JSON_VALUE(data, '$.mobius.contentClass') AS mobius_contentclass,
    JSON_VALUE(data, '$.mobius.contentIngestion') AS
 mobius_contentingestion
  FROM events;
GO
-- View: dbo.supersetbetadocumentview
CREATE VIEW dbo.supersetbetadocumentview
  AS
  SELECT JSON_VALUE(documentdata, '$.id') AS id,
    JSON_VALUE(documentdata, '$.timestamp') AS timestamp,
    JSON_VALUE(documentdata, '$.type') AS type,
    JSON_VALUE(documentdata, '$.user') AS userid,
    JSON_VALUE(documentdata, '$.action') AS action,
    JSON_VALUE(documentdata, '$.description') AS description,
    JSON_VALUE(documentdata, '$.mobius.server') AS mobius_server,
    JSON_VALUE(documentdata, '$.mobius.recipient') AS mobius_recipient,
    JSON_VALUE(documentdata, '$.mobius.section') AS mobius_section,
    JSON_VALUE(documentdata, '$.mobius.contentClass') AS
 mobius_contentclass,
    JSON_VALUE(documentdata, '$.mobius.contentIngestion') AS
 mobius_contentingestion,
    JSON_VALUE(documentdata, '$.page_count') AS pgcount,
    JSON_VALUE(documentdata, '$.page_of_doc') AS docpage,
    JSON_VALUE(documentdata, '$.section_id') AS docname,
    JSON_VALUE(documentdata, '$.ProcessID') AS processid
  FROM documentevents;
GO
```

Oracle

```
-- View: mobius.supersetbetaview
-- DROP VIEW mobius.supersetbetaview;
CREATE OR REPLACE VIEW mobius.supersetbetaview
AS
SELECT tbl."data"."id" AS "id",
  tbl."data"."timestamp" AS "timestamp",
  tbl."data"."type" AS "type",
  tbl."data"."user" AS "user",
  tbl."data"."action" AS "action",
  tbl."data"."description" AS "description",
  tbl."data"."mobius.server" AS "mobius_server",
  tbl."data"."mobius.recipient" AS "mobius_recipient",
  tbl."data"."mobius.section" AS "mobius_section",
  tbl."data"."mobius.contentClass" AS "mobius_contentclass",
  tbl."data"."mobius.contentIngestion" AS "mobius_contentingestion"
FROM "events" tbl;
-- View: mobius.supersetbetadocumentview
-- DROP VIEW mobius.supersetbetadocumentview;
CREATE OR REPLACE VIEW mobius.supersetbetadocumentview
AS
SELECT tbl."documentdata"."id" AS "id",
  tbl."documentdata"."timestamp" AS "timestamp",
  tbl."documentdata"."type" AS "type",
  tbl."documentdata"."user" AS "user",
  tbl."documentdata"."action" AS "action",
  tbl."documentdata"."description" AS "description",
  tbl."documentdata"."mobius.server" AS "_mobius_server",
  tbl."documentdata"."mobius.recipient" AS "mobius_recipient",
  tbl."documentdata"."mobius.section" AS "mobius_section",
  tbl."documentdata"."mobius.contentClass" AS "mobius_contentclass",
  tbl."documentdata"."mobius.contentIngestion" AS
 "mobius_contentingestion",
  tbl."documentdata"."page_count" AS "pgcount",
  tbl."documentdata"."page_of_doc" AS "docpage",
  tbl."documentdata"."section_id" AS "docname",
  tbl."documentdata"."ProcessID" AS "processid"
FROM "documentevents" tbl;
```

Create datasets to generate charts

Using the two views **supersetbetaview** & **supersetbetadocumentview**, you have to create datasets in Superset so that charts can be created.

1. Navigate to the `Data` > **Datasets** menu item and create a new dataset.
2. Click on **+Dataset** button.
3. Select the correct items from the drop-downs.

   PostgreSQL

   a. Enter Database (Demo Database), Schema (public) & Table Schema.

   b. Click on **Add** button.

Oracle

  a. Enter Database (Oracle), Schema (mobius) & Table Schema.
  b. Click on **Add** button.

SQL Server

  a. Enter Database (Microsoft SQL Server), Schema (dbo) & Table Schema.
  b. Click on **Add** button.

4. Add views supersetbetaview and supersetbetadocumentview as datasets.
5. For Oracle and SQL Server, edit both datasets **Is temporal** and **Default datetime** in the timestamp column.