社 区

理解超集语义层

preset™

语义层简介

Apache Superset 已发展成为最流行的商业智能平台之一，因为可访问的开源软件包中提供了丰富的功能集。 Superset 中的主要功能区域之一是**语义层**。

语义层是对数据仓库或数据湖中的基础数据的抽象。语义模型将 SQL 映射为更人性化的隐喻。

当然，Superset 社区并没有发明语义层。据推测，它于1992 年首次由 BusinessObjects 申请了专利，并慢慢融入到 Cognos 和 Microstrategy 等其他传统数据科学工具中。最近，Looker （已被

Google Cloud 收购）基本上围绕语义层（LookML）构建了整个产品。 Looker 的 LookML 将语义层思想带到了 BI 讨论的

订阅我们的博客更新

接收新博客文章的每周摘要

订阅

经过 BI 领域十年的 工具之一。现在让我们深入了解 Superse

语义层薄还是厚：

Superset 中的语义层*设计*得很薄。为什么是这样？

Looker 等上一代 BI 工具鼓励最终用户投入*巨资*构建大量 LookML 模型来填充语义层。虽然这使 Looker 成为组织中业务指标的真实来源，但它也对 BI 工具产生了巨大的**锁定**。如果您决定更换 BI 工具，则无法随身携带 LookML 模型并将其与其他工具一起使用。 Google 对 Looker 的收购加剧了组织对这种锁定的焦虑，以至于 Google 实际上被迫将 LookML 与 Tableau 集成。这样，组织就可以使用 LookML 进行转换（Looker 的优势），使用 Tableau 进行可视化（Tableaus 的优势）。

然而，未来看起来与过去有很大不同。现在有完整的开源项目和产品构建统一的语义层，位于数据库和 BI 层之间：

- 立方体
- 米特里QL
- 转换
- *还有很多其他的！*

薄语义层的目标主要是实现**最后一英里数据转换**，以实现 BI 工具中可视化的明确目的。我将在这篇文章中展示一些具体的例子。

现在我已经了解了一些背景信息，让我们深入了解具体细节。 Superset 中的语义层由三个主要隐喻组成：

- **虚拟数据集**
- **指标**
- **计算列**

As you traverse through this blog post, it's helpful to remember that all of these features exist **to help you craft better and more complex SQL queries.** BI tools provide convenient UI interfaces and affordances so you can avoid having to manually write very long SQL queries from scratch.

Physical vs Virtual Datasets

COOKIE 设置

The data layer in Superset falls into two buckets: **physical datasets** and **virtual datasets**.

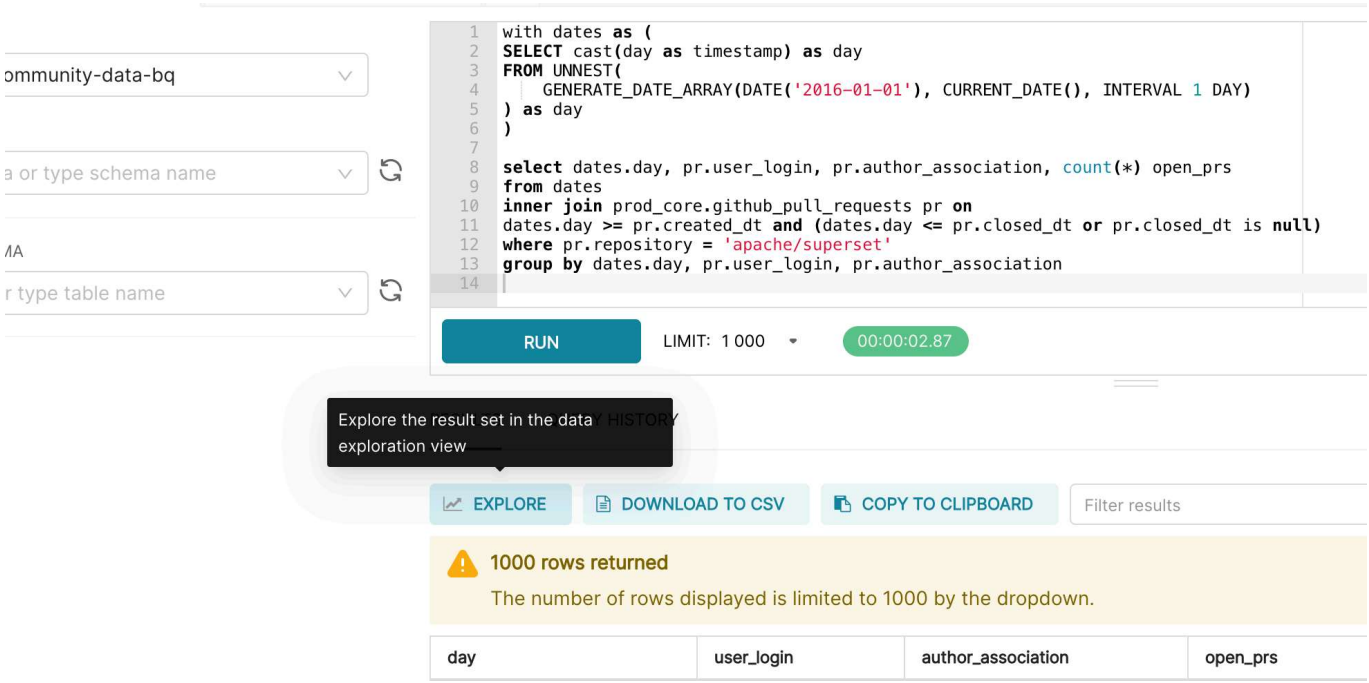
A physical dataset in ause a physical dataset reflects a re elevant information from the database (l Superset's metadata database. n click **Sync Columns from Sourc**

Virtual datasets enable you to elevate a freeform SQL query against your database into a dataset entity in Superset. Virtual datasets inherit most of the same superpowers as physical datasets:

- column types (inferred from results of running the query)
- ability to define metrics
- ability to define calculated columns
- ability to certify metrics or calculated columns
- setting a cache timeout

Using Virtual Datasets

The fastest way to create a virtual dataset is to write and run your query in SQL Lab. Then, you can click **Explore** near the results tray and you'll be asked to name the virtual dataset:



So when should you use a virtual dataset? Here are some use cases with some concrete examples.

1. Joining multiple tables (or self-joining against the same table)

- To visualize total revenue by customer persona, we need to JOIN between a custom[illegible]s virtual dataset.

2. Ad-hoc, one-[illegible]
    - If you w[illegible] of a custom SQL query y[illegible]irtual dataset.
3. Transforming data in more nuanced ways than currently what Explore
    - No-code UI's can't really replace the need to write custom SQL entirely because of the complexity, but Superset's Explore view attempts to augment common slice-and-dice workflows common in analytics. For example, if you want to heavily transform the underlying data using window functions, virtual datasets are a great way to prep the data before visualizing in Explore.

You'll notice that some of these workflows have a "temporary" framing attached to them. This is because populating the semantic layer with hundreds or even thousands of virtual datasets makes it more difficult for a data platform / governance team to help keep the backend data systems (databases, data pipelines, caching layers, etc.) highly performant and reliable. In addition, in larger organizations, this can lead to data drift & metric inconsistency.

But as with most advice, it's contextual to your organization! If you're a small, nimble team like us at Preset, virtual datasets are powerful at unblocking end-user analysts in the short run. Commonly used transformations in the semantic layer can then be methodically be migrated to the data pipeline over time, in a more agile way.

Metrics

Superset's origins are in fast, slice & dice, exploratory analytics specifically for Apache Druid. In this workflow, it's natural to alternate between:

- prototyping visualizations quickly using different metrics
- defining a set of commonly used metrics for wider use in an organization

In more specific terms, a metric in Superset is any valid, aggregating SQL snippet that can be included in a SELECT clause. Each line within the SELECT clause below are valid metrics in Superset:

```
SELECT
    COUNT(*),
    SUM(CASE WHEN action='deal_closed' THEN 1 ELSE 0 END),
```

```
MAX('revenue'),
SUM('deals_oper') / SUM('deals_closed')
MAX('revenue'
```


Here are some use c

1. Converting a t

```
SUM(CASE WHEN action='deal_closed' THEN 1 ELSE 0 END)
```

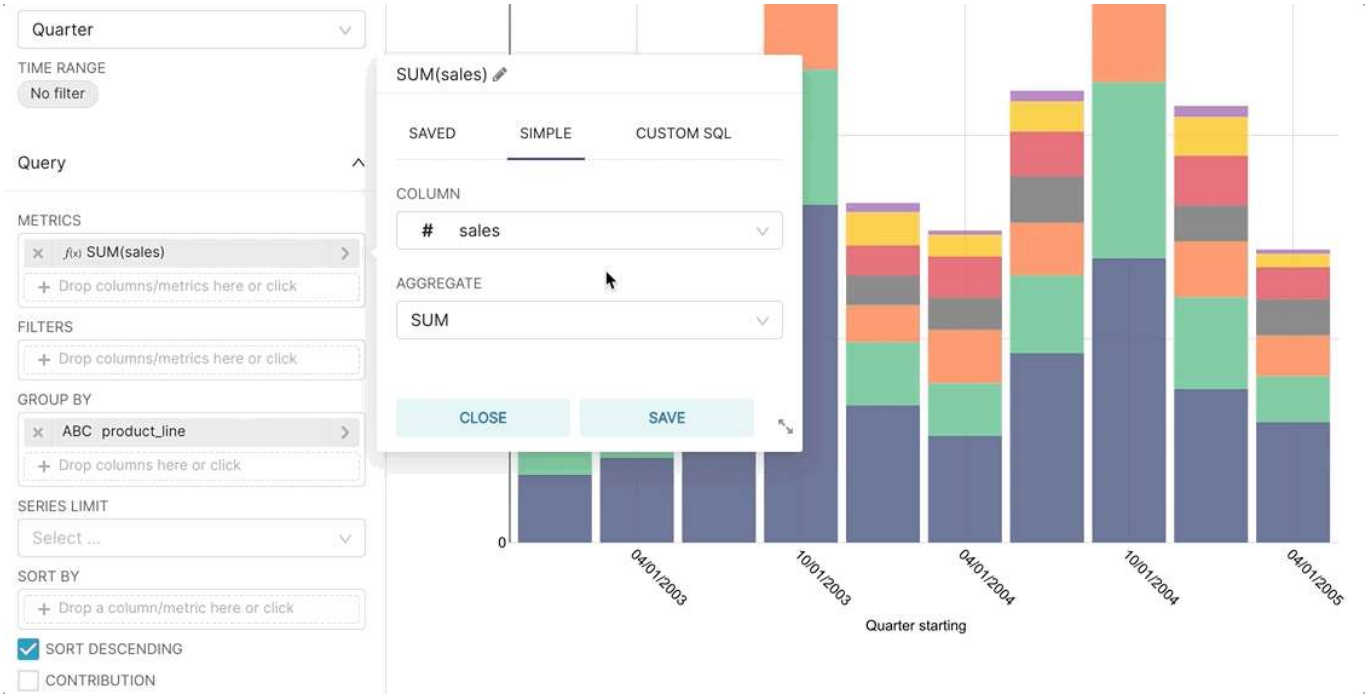
2. Computing a ratio of deals open to deals closed:

```
SUM('deals_open') / SUM('deals_closed')
```

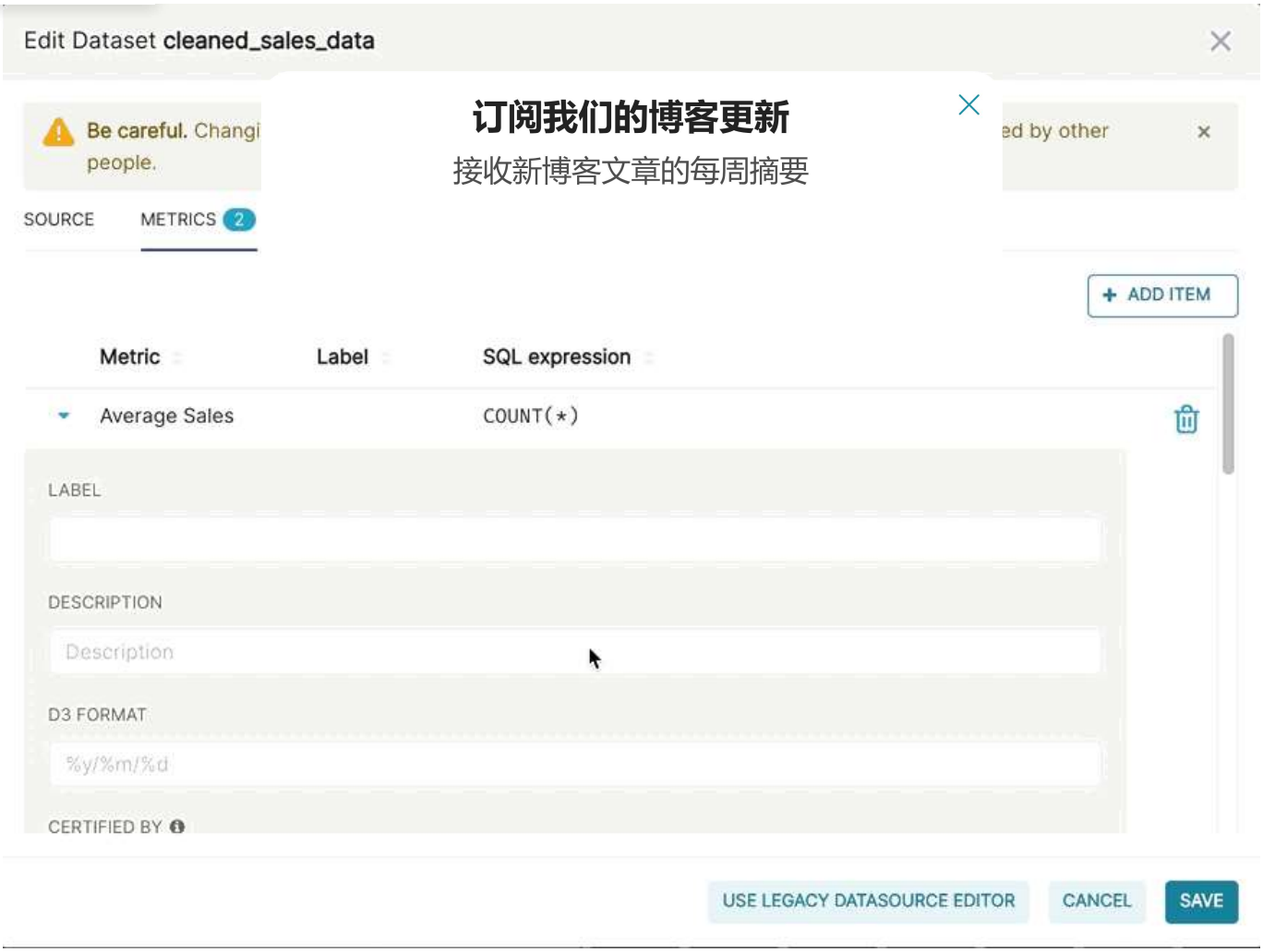
3. Calculating the range for a column:

```
MAX('revenue') - MIN('revenue')
```

The easiest way to get started with metrics is to select a Time-series visualization type while in the Explore workflow. Then, you can quickly try a few different metrics.



Then, if you want to publish a metric for more common use, you can persist the metric in the **Edit Dataset** view.



Calculated Columns

Calculated columns let you define simple transformations (as SELECT statements) for quick, last-mile data preparation.

You can define any valid, non-aggregate SQL snippets that can be included in a SELECT clause. Note that this means you can reference multiple columns in your calculated column queries.

Here are some examples:

1. To create a Table chart with clickable links, you can use a calculated column to augment the underlying data with HTML. The following snippet performs string concatenation to generate HTML using each row's value for `repo`, `parent_id`, and `title`:

```
CONCAT('<a href="https://github.com/", repo, '/issues', parent_id, ">', title, '</a>
```

2. For a more human-friendly presentation in visualizations, you often want to re-label group names in your data.

```
Case
    When is
    When is
    Else 'N
End
```


3. Converting / casting column types

```
CAST(sales_cts) as int)
```

4. Calculating number of days between two date columns

```
DATE_DIFF(DATE '2010-07-07', DATE '2008-12-25', DAY)
```

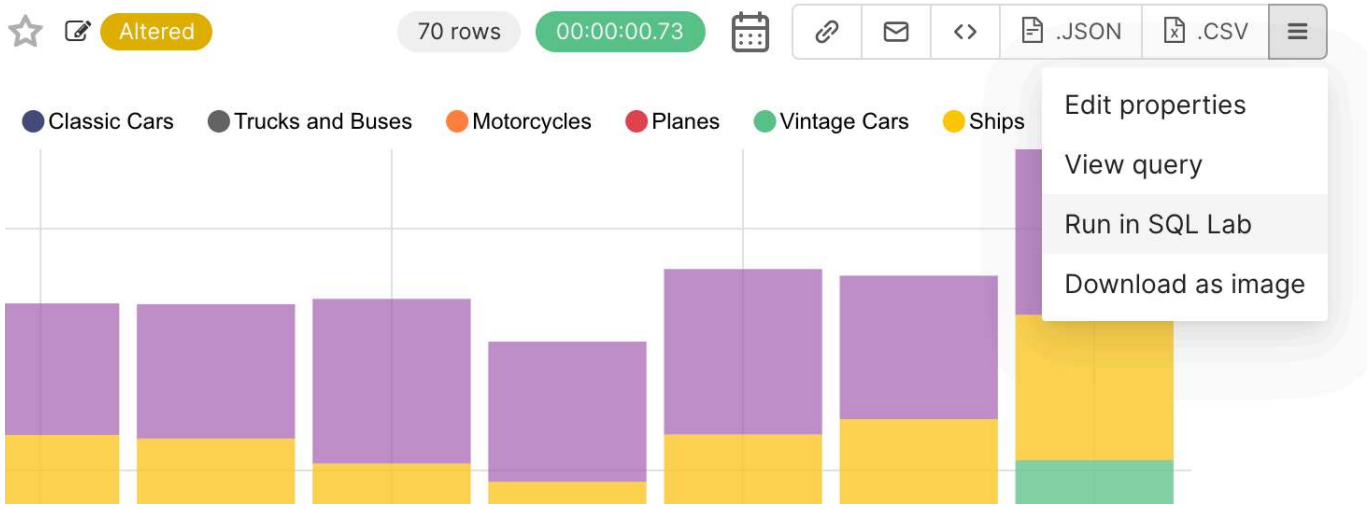
Debugging Queries

Ultimately, the end-user facing features of Superset (like the ones discussed in this post) help you generate more complex SQL queries.

- **Virtual datasets**, **metrics**, and **calculated columns** provide useful abstractions and superpowers to augment your queries.
- **Explore**提供了无代码、拖放式 UI，可快速生成支持可视化的查询。

由于 Superset 中工作流的最终工件是 SQL，因此您可以使用**SQL Lab**来调试所编写的任何查询。在“探索”中，选择右上角的汉堡菜单，然后选择**“查看查询”**以观察生成的查询超集。然后，您可以在 SQL Lab 中复制、粘贴并运行查询（确保您在 SQL Lab 中从下拉菜单中选择了正确的数据库上下文）。

更快的方法是从汉堡菜单中单击“**在 SQL Lab 中运行**”。



从这里，您可以：

- 检查并内化复杂数据的形状，以便您可以发现可能需要对假设进行哪些更改
- 调试 SQL 查询问题并更新相关虚拟数据集、指标或计算列

我们将通过提及预设文档中的语义层文档来结束这篇文章，它可以作为本文的可靠参考和补充。

让我们直观地了解一下。今天尝试预设。

订阅我们的博客更新

接收新博客文章的每周摘要

548 Market St, PMB 51897
旧金山, CA 94104-5401

产品

预设云
托管私有云
嵌入式仪表板
应用程序编程接口
信任与安全

用例

商业智能（BI）
内部模具
面向客户的应用程序

价钱

我们的定价
联系销售人员

资源

博客
文档
活动
播客
什么是超级组?
顾客

关于我们

职业机会

推荐奖金

订阅我们的博客更新

接收新博客文章的每周摘要

COOKIE 设置