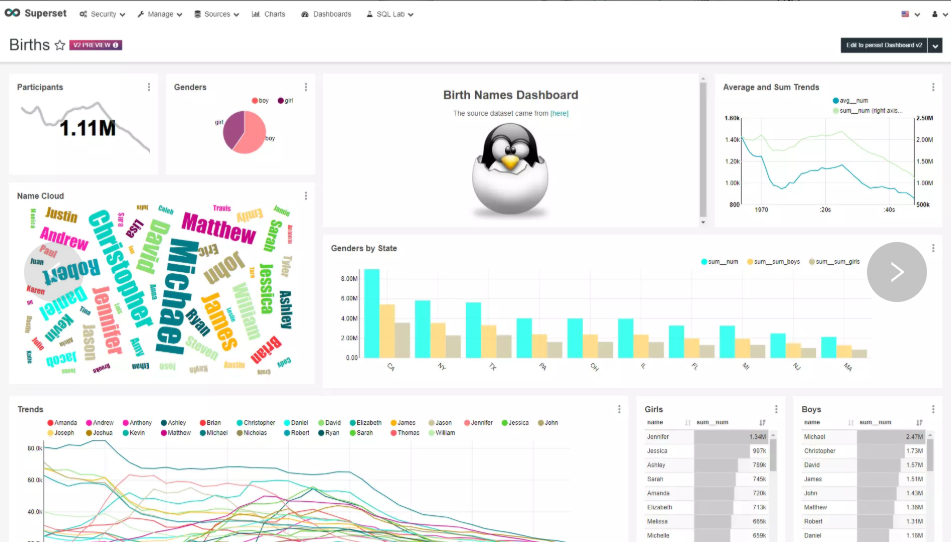
# Superset简介



[Superset](https://airbnb.io/projects/superset/) 是一款由 Airbnb 开源的“现代化的企业级 BI（商业智能） Web 应用程序”，其通过创建和分享 dashboard，为数据分析提供了轻量级的数据查询和可视化方案。  
  Superset 的前端主要用到了 [React](https://reactjs.org/) 和 [NVD3/D3](http://nvd3.org/)，而后端则基于 Python 的 [Flask](http://flask.pocoo.org/) 框架和 [Pandas](http://pandas.pydata.org/pandas-docs/stable/)、[SQLAlchemy](http://www.sqlalchemy.org/" \t "_blank) 等依赖库，主要提供了这几方面的功能：

* 集成数据查询功能，支持多种数据库，包括 MySQL、PostgresSQL、Oracle、SQL Server、SQLite、SparkSQL 等，并深度支持 [Druid](http://druid.io/)。
* 通过 NVD3/D3 预定义了多种可视化图表，满足大部分的数据展示功能。如果还有其他需求，也可以自开发更多的图表类型，或者嵌入其他的 JavaScript 图表库（如 HighCharts、ECharts）。
* 提供细粒度安全模型，可以在功能层面和数据层面进行访问控制。支持多种鉴权方式（如数据库、OpenID、LDAP、OAuth、REMOTE\_USER 等）。

  Superset 的搭建与使用非常简单，只需要一些 Python 基础，下面先从创建虚拟环境开始。

# Superset安装

## 创建虚拟环境

  Superset 的依赖包较多，为了避免冲突，需要先搭建虚拟环境，再进行安装，这里推荐使用 [Anaconda](https://www.anaconda.com/) 自带的 conda 工具创建虚拟环境：

conda create -n superset python=3.6

  创建虚拟环境成功后，启动虚拟环境：

activate superset

## 安装

  使用豆瓣源安装 Superset：

pip install superset -i https://pypi.douban.com/simple

  这里遇到了一个坑，某个依赖包没有被正确的安装，当时随手就解决了，没有记录下来。只大概记得报错信息提到 Microsoft Visual C++ 14.0 is required ，这是因为该依赖包需要安装 C++ 进行编译。  
  一般不用为此专门安装 Microsoft Visual C++ 14.0，而是去下载该依赖包的 whl 格式文件（需要对应虚拟环境的 python 版本），再进入 whl 文件所在的路径通过 pip install 安装即可。

## 初始化

  初始化的官方步骤如下：

# 创建管理员账号

fabmanager create-admin --app superset

# 初始化数据库

superset db upgrade

# 载入案例数据

superset load\_examples

# 初始化角色和权限

superset init

# 启动服务，端口号 8088，使用 -p 更改端口号

superset runserver

  但在命令行中直接运行 superset， 会提示“不是内部或外部命令”。要解决这个问题，可以直接通过 cd 命令进入 Superset 安装目录（ ...\Anaconda3\envs\superset\Lib\site-packages\superset\bin ）。然后运行如下命令：

python superset db upgrade

python superset load\_examples

python superset init

python superset runserver

  这里直接运行 python superset runserver 会出错，原因是 Superset 使用 gunicorn 作为应用程序服务器，而 gunicorn 不支持 Windows。需要在命令行中添加 -d，使用 development web server 运行。最终运行命令为：

python superset runserver -d

  但是这种部署方式，官方并不建议在生产环境中使用。在 Superset 的 [issues 922](https://github.com/airbnb/superset/issues/922)，有人提供了一种方法，使用 waitress。首先安装 waitress：

pip install waitress

  接着找到 superset/cil.py 的 debug\_run() 函数：

def debug\_run(app, port, use\_reloader):

app.run(

host='0.0.0.0',

port=int(port),

threaded=True,

debug=True,

use\_reloader=use\_reloader)

  将其改写成：

def debug\_run(app, port):

from waitress import serve #使用 waitress 解决 gunicorn 不支持 windows 问题

return serve(

app,

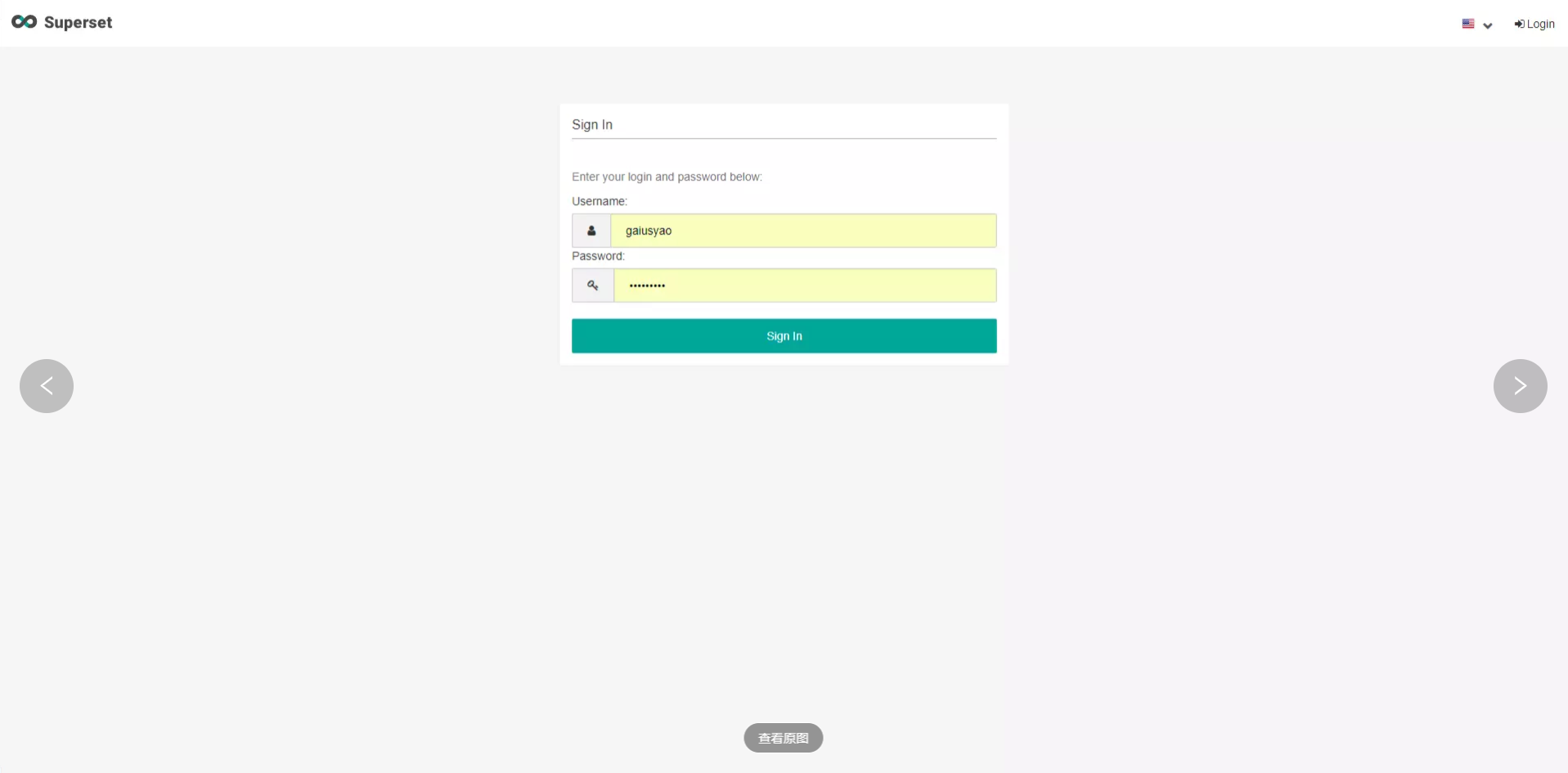
host='0.0.0.0',

port=int(port))

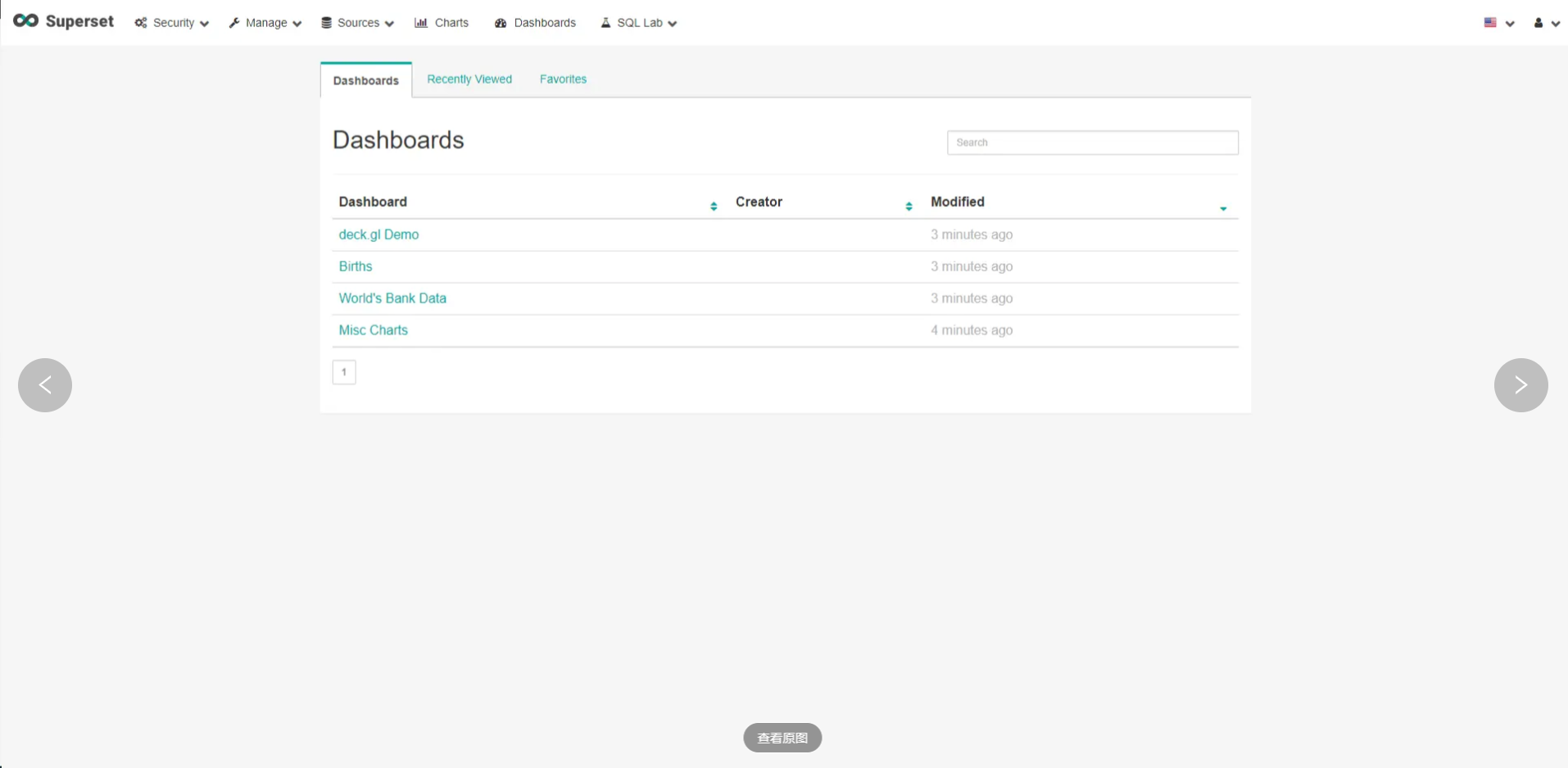
  再次运行 Superset:

python superset runserver -d -p 8079

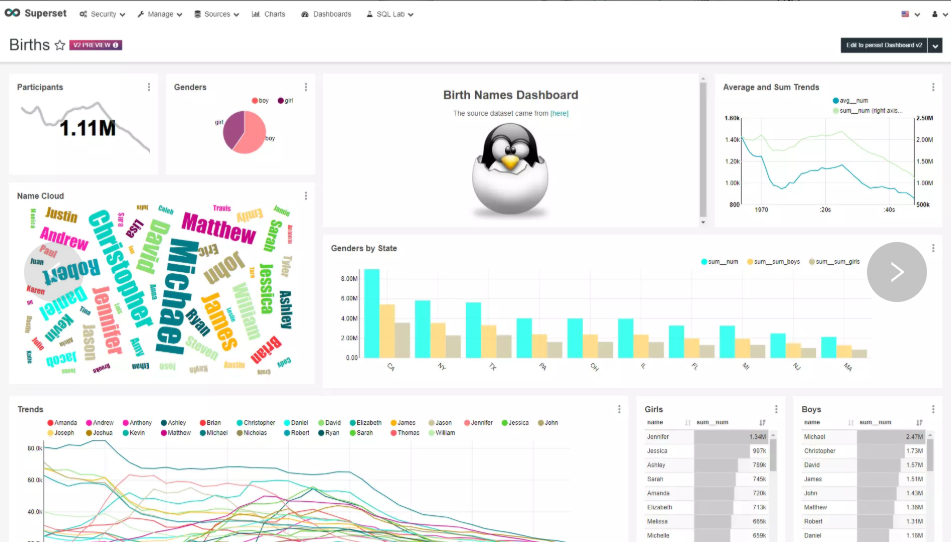
  最后在浏览器输入 http://localhost:8079 进入登录界面：



输入在 fabmanager create-admin --app superset 这步设置的账号和密码登录，进入 Superset 首页（*右上角国旗处可设置语言*）：



这里 Superset 已经用之前加载的示例数据，建立了 4 个 dashboard，选择其中一个进行查看（*即本文开始的那个 dashboard*）：

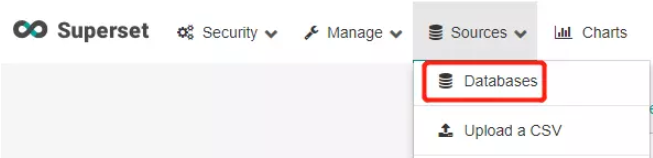


## 数据源

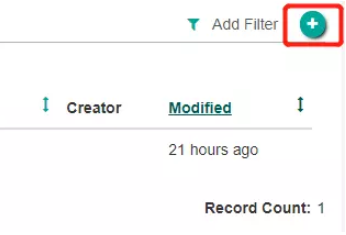
接下来，将演示如何连接数据库，以及怎样使用表和导入 CSV 到数据库。Superset 还深度支持 Druid（*一个高效的海量数据查询系统*），但这里不做介绍。

### 连接到数据库

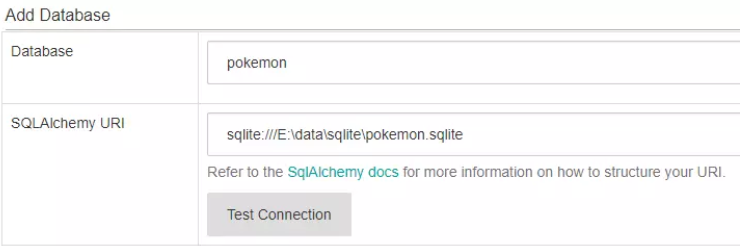
从顶部导航菜单的**Source — Databases** 进入数据库列表页：



点击在数据库列表右上角的绿色加号按钮：



点击按钮后，将进入添加数据库的表单页，因为这只是一次简单的演示，只需要填写两项：Database 和 SQLAlchemy URL，分别是数据库名称和 SQLAlchemy 的连接 URI （参阅为目标数据库创建连接 URI 的 *[SQLAlchemy 文档](http://docs.sqlalchemy.org/en/rel_1_0/core/engines.html" \l "database-urls" \t "_blank)*）。



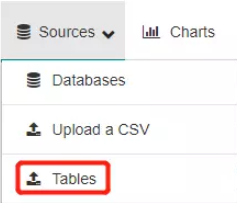
这里为了方便，用的是本地的 SQLite 数据库。也可以使用其他数据库，如官方文档推荐的 PostgreSQL 的一些[示例数据集](https://wiki.postgresql.org/wiki/Sample_Databases)或官方文档使用的[示例天气数据](https://github.com/dylburger/noaa-ghcn-weather-data)。

接着点击 Test Connection 按钮，测试是否能成功连接到数据库。

当连接测试成功后，可在页面底部看到该数据库下的数据表，点击 Save 按钮，完成创建。

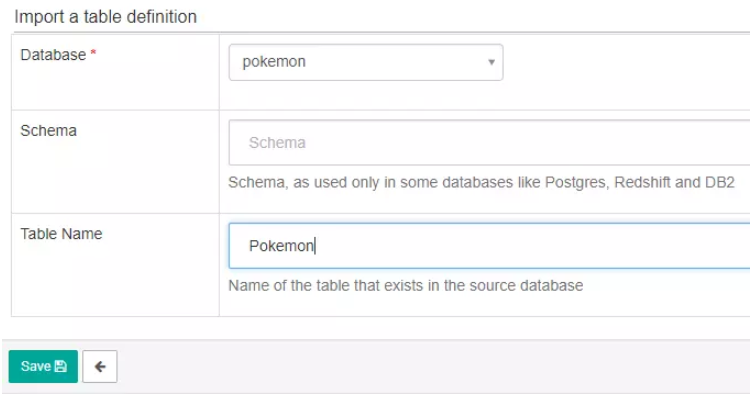
### 创建表

现在已经配置了数据库，接下来需要向 Superset 添加想要查询的特定表。从 **Sources — Tables** 进入到数据表列表页：

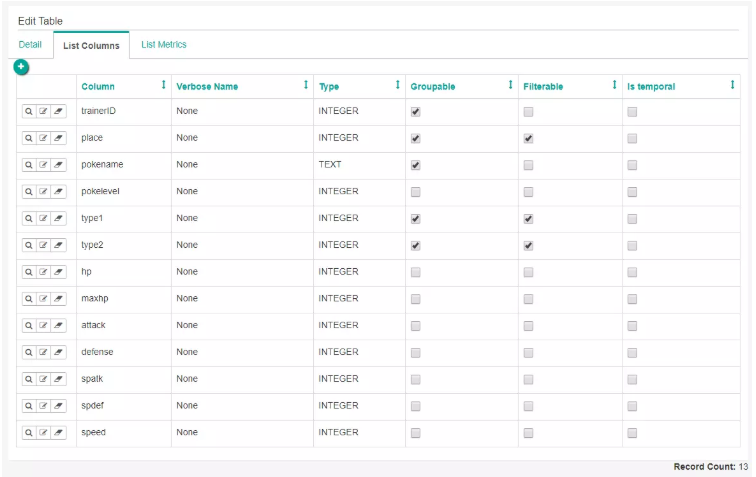


点击列表页右上角的绿色加号按钮。

点击按钮后，将进入添加数据表的表单页，在该页面依次填写目标数据库、数据表名称、数据库模式（可选），再点击 Save 按钮，即可完成创建。

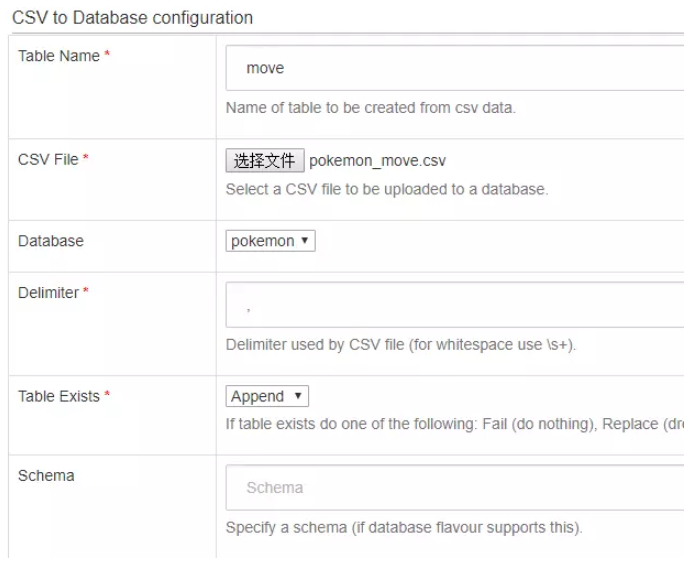


创建完成后，将重定向回到列表页，此时在页面顶部会出现一条消息提示指示表已创建，在列表页还可以对已添加的数据表进行编辑，点击编辑图标，可进入表的编辑页，对表的详细信息、字段、指标进行配置，这里演示对表字段的配置（*设置是否可对指定字段进行分组或过滤*）：



### 上传CSV

Superset 还可以导入 CSV 到数据库中，从 Sources — Upload a CSV 进入到导入 CSV 的表单页，依次填写表名称、导入 CSV 文件、选择要导入的数据库，再点击 Save 按钮完成导入：



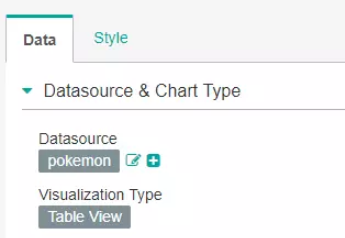
创导入完成后，将重定向回到数据表列表页，此时在页面顶部会出现一条消息提示指示表已创建。

## 创建dashboard

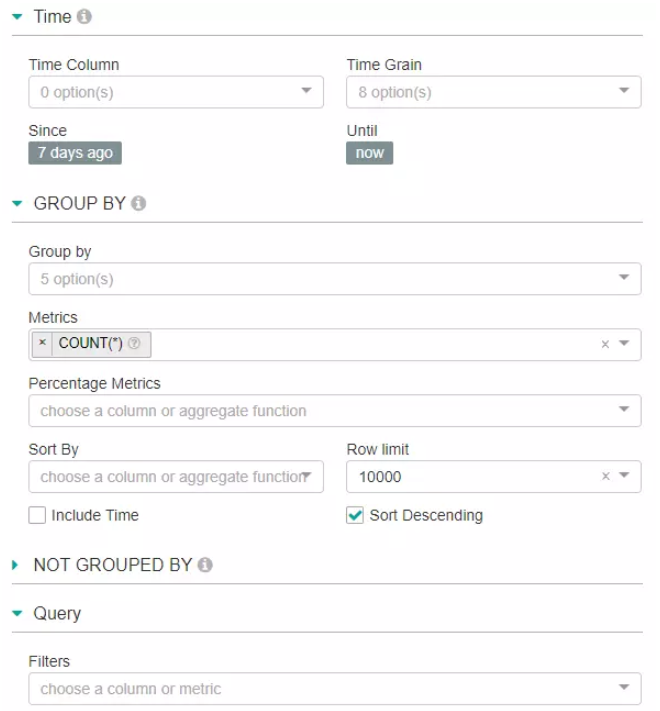
### 探索数据

要开始探索数据，只需在可用数据表列表中点击刚刚创建的表名。

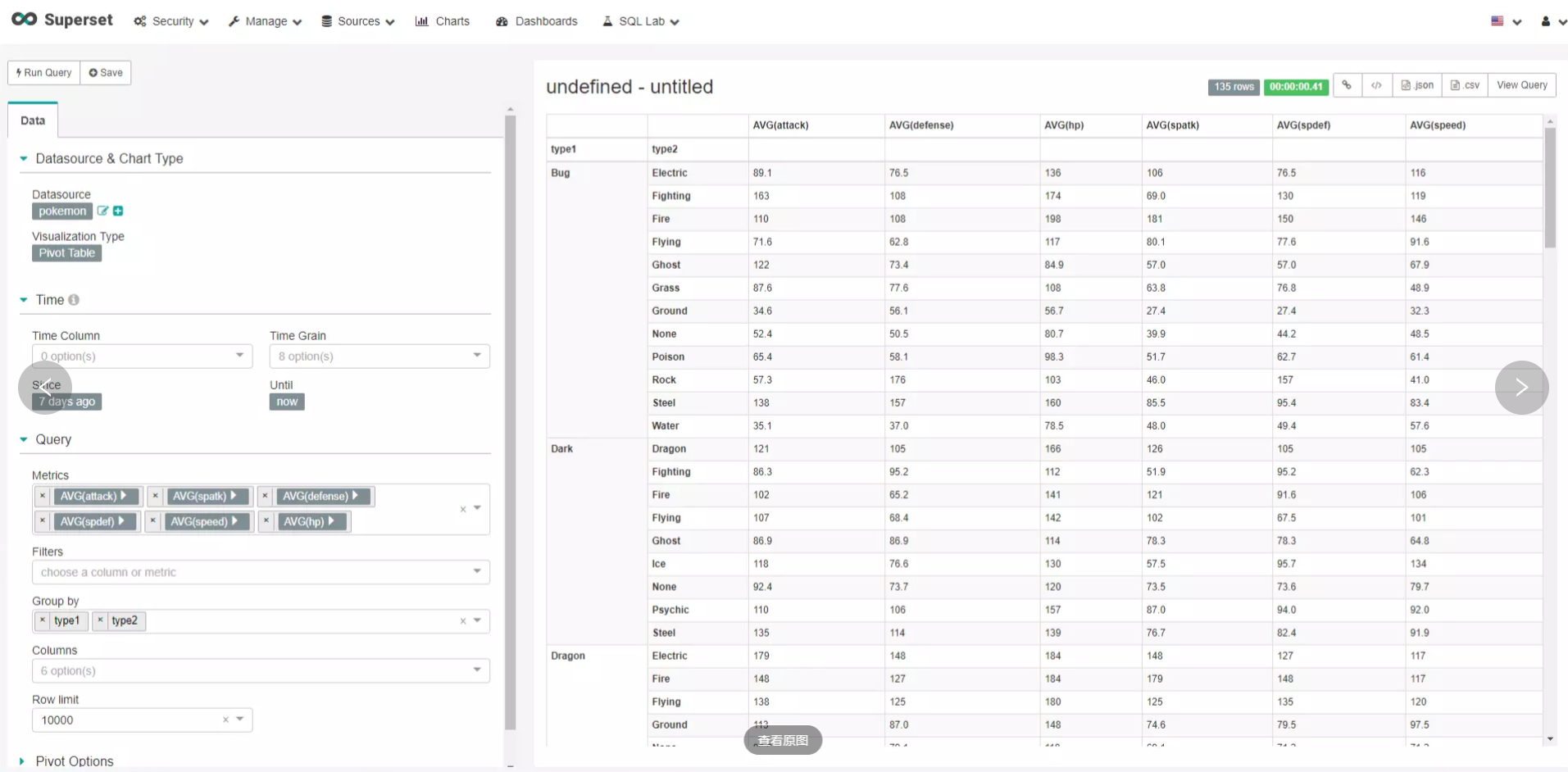
点击表名后进入表的可视化页面，默认的可视化类型为表视图。



在 Datasouce & Chart Type 下方，依次可以进行时间、Group By、Not Group By 以及字段过滤等设置。

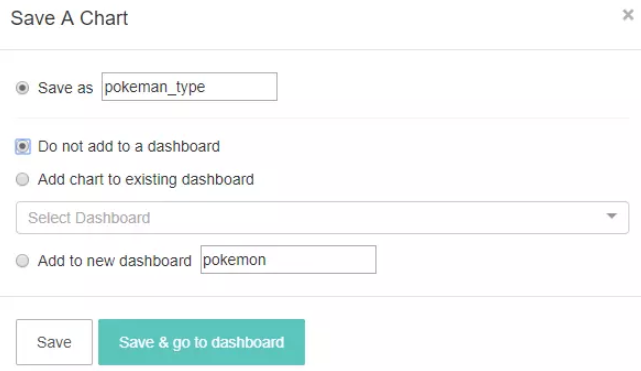


在页面左侧完成相关设置后，点击 Run Query 按钮，即可在右侧的可视化视图，查看数据的可视化展示：



### 创建图表

通过更改可视化类型和其他设置，可以很灵活地对数据进行探索性分析。若在探索的过程中发现某个有价值的点，可以点击左上角的 Save 按钮，在出现的弹出框中命名图表并选择是否将其添加到 dashboard，以将其保存为图表：



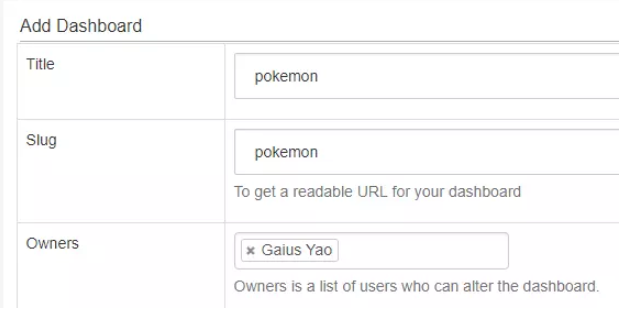
从顶部导航菜单的 **Charts** 进入到图表的列表页，找到刚刚创建的图表：

点击图表名称进入到刚才的可视化页面，对已保存的图表进行再次编辑。

### 创建dashboard

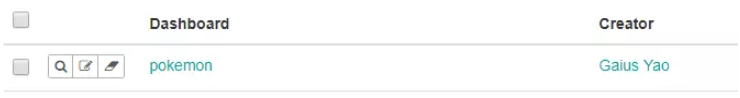
从顶部导航菜单的 **Dashboards** 进入到 dashboard 的列表页，点击右上角的绿色加号按钮。

点击按钮后，将进入添加 dashboard 的表单页，在该页面依次填写 dashboard 名称和拥有者，再点击 Save 按钮，即可完成创建：



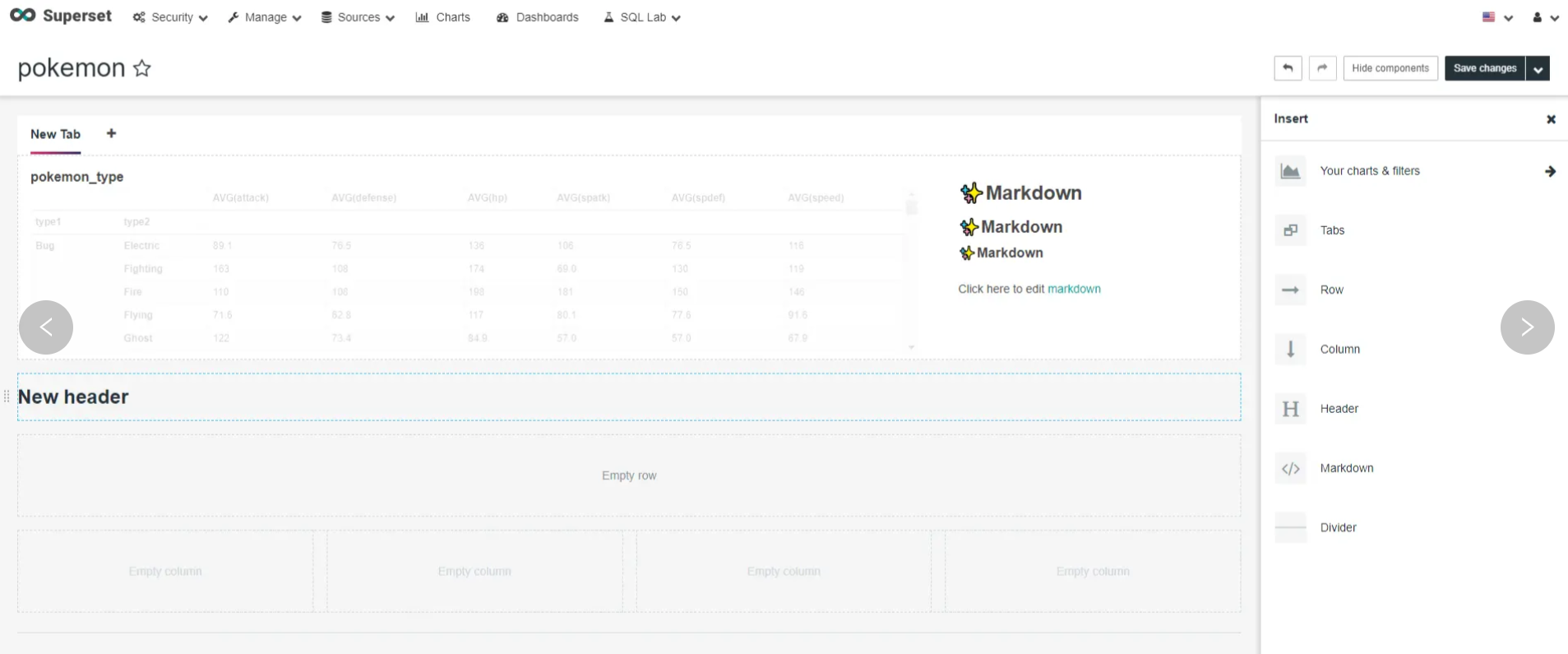
### 编辑dashboard

完成dashboard的创建后，将重定向到dashboard的列表页，找到刚创建的 dashboard：



点击 dashboard 的名称，进入 dashboard，此时 dashboard 还未配置任何图表，处于空值状态，点击右上角的 Edit dashboard 按钮进行编辑。

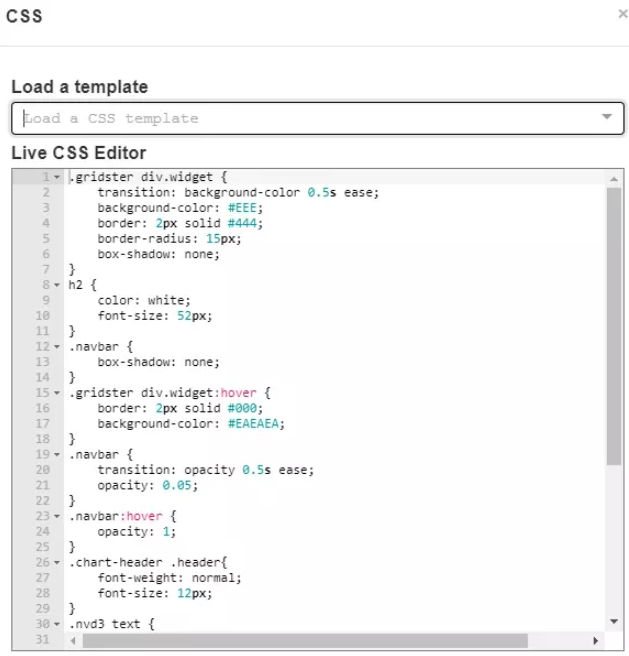
进入编辑状态后，可已导入图表（*仅限 dashboard 拥有者拥有的图表*）、标签页、行、列、标题、Markdown 和分割线等组件：



并可通过拖拽编辑 dashboard 的布局（在拖拽过程中，Superset 还提供了辅助线和栅格进行提示）：



Superset 还可以通过编辑 CSS 修改 dashboard 的样式：



在进行编辑之后，点击 Save changes 按钮，即可完成操作。

## SQL查询

在使用 SQL 查询前，需要进行两项设置：  
  首先，从顶部导航菜单的 **Sources — Databases** 进入数据库的列表页，选中数据库进行编辑，将 Expose in SQL Lab 和 Allow Run Sync 都勾选上，其余的不要勾选。

在勾选完上述两项之后，Windows 用户还会出现 “module" object has no attribute 'SIGALRM' 错误，这又是由于 Windows 环境下依赖包不兼容导致的 —— Python 的 signal 包只作用于 Linux 和 Mac ，在 Windows 下不启作用。解决方法很简单粗暴，在 superset/utils.py 下找到相关代码，把 signal 所在行都注释，然后再加上一个 pass （*这块代码的功能是在超时后将查询进程杀掉，注释后没大影响*）。

def \_\_enter\_\_(self):

try:

pass

#signal.signal(signal.SIGALRM, self.handle\_timeout)

#signal.alarm(self.seconds)

except ValueError as e:

logging.warning("timeout can't be used in the current context")

logging.exception(e)

def \_\_exit\_\_(self, type, value, traceback):

try:

pass

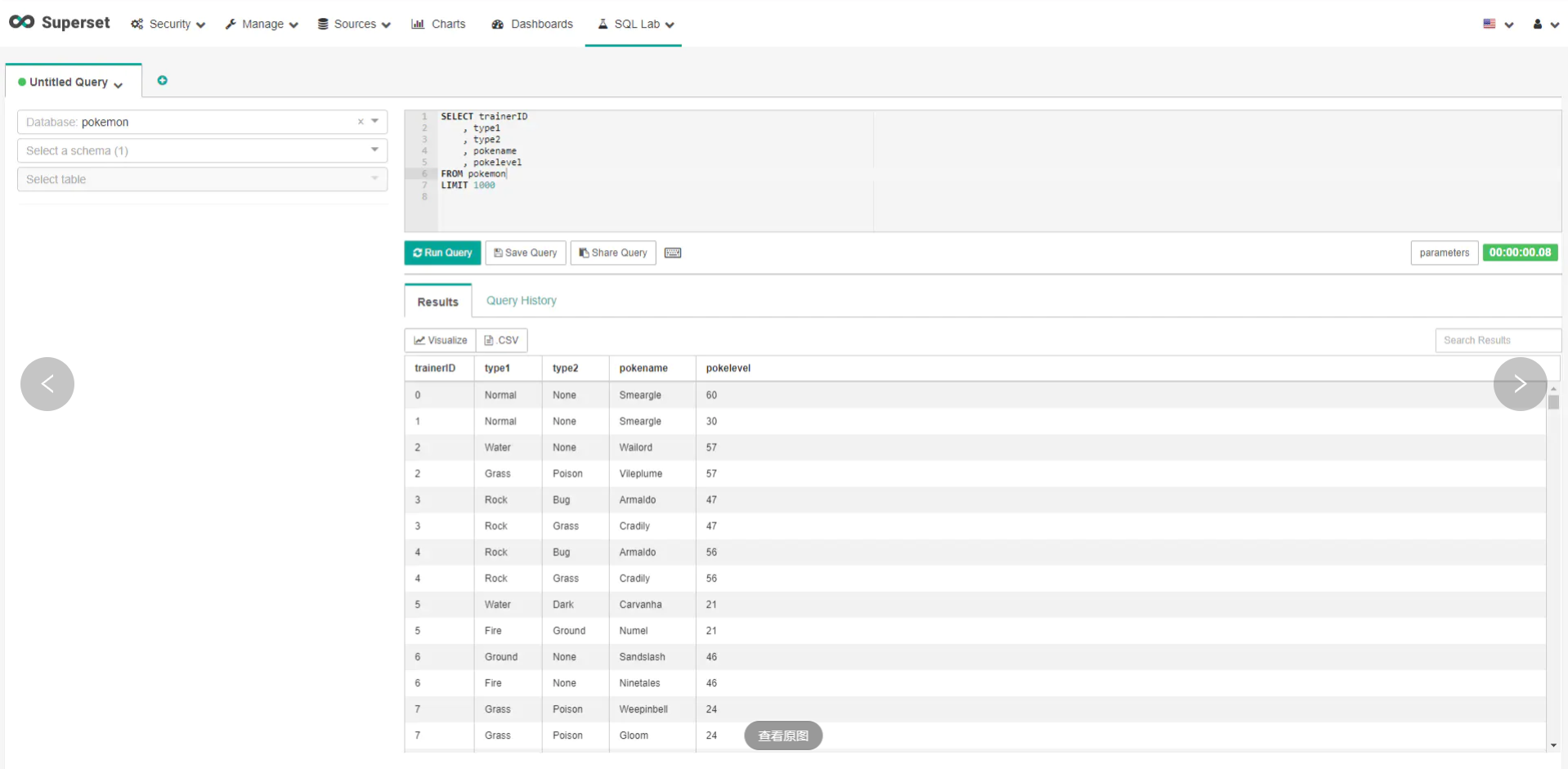
#signal.alarm(0)

except ValueError as e:

logging.warning("timeout can't be used in the current context")

logging.exception(e)

上面两步做完后，即可在**SQL Lab — SQL Editor**进行 SQL 查询操作：



可对查询语句进行执行、保存、分享（复制）操作，还可以对查询结果可以进行可视化和导出为 CSV 文件。

## 安全性

Superset 中的安全性由 Flask AppBuilder（FAB）处理（一个“简单快速的应用程序开发框架，构建在Flask之上”）。FAB 提供身份验证、用户管理、权限和角色，可参阅其[安全文档](http://flask-appbuilder.readthedocs.io/en/latest/security.html" \t "_blank)。角色由一组权限组成，不同的用户角色被赋予了不同的权限，Superset 本身提供了一组基本角色：

* **Admin：**拥有所有可能的权限，包括从其他用户授予或撤消权限以及更改其他人的切片和 dashboard。
* **Alpha：**可以访问所有数据源，但不能授予或撤消其他用户的访问权限。它们也仅限于改变它们拥有的对象。可以添加和更改数据源。
* **Gamma：**访问受限，只能使用通过另一个互补角色获得访问权限的数据源。他们只能查看由他们有权访问的数据源制作的切片和 dashboard。无法更改或添加数据源，但可以可以创建切片和 dashboard。
* **sql\_lab：**被授予对 SQL Lab 的访问权限。
* **public：**可以通过在 superset\config.py 设置 PUBLIC\_ROLE\_LIKE\_GAMMA = True，授予该角色与 Gamma 角色相同的权限集。

在执行 superset init 命令时，所有这些基本角色将重新同步到初始值，因此不建议通过授予或撤消权限来更改这些基本角色。

  这里不对 Superset 的安全机制做过多展开，如果想了解更多，包括如何为用户提供对特定数据集的访问权限，以及如何定制自己的角色，可以参阅官方的[安全文档](http://superset.apache.org/security.html)。