

Visualizing Real Time Data Using ClickHouse and Superset

Srini Kadamati -- Preset
Robert Hodges -- Altinity

Presenter Bios



Robert Hodges - CEO at Altinity

30+ years on DBMS plus
virtualization and security.
ClickHouse is DBMS #20



Srini Kadamati - Senior Developer
Advocate at Preset.io

6+ years in data science & data
science education. Committer to
Apache Superset.

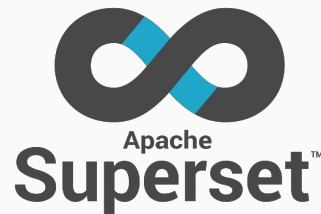
Introducing ClickHouse and Superset

ClickHouse SQL Data Warehouse



- C++ Binary
- Apache 2.0 License
- Active community
- Column storage with compression
- Efficient parallel processing
- Sharding and replication

Superset Data Visualization and Exploration Platform



- Python + Javascript (Browser)
- Apache 2.0 License
- Active community
- Supports nearly any SQL database
- Dozens of chart types
- Preset Cloud managed service

Visualization and data analysis

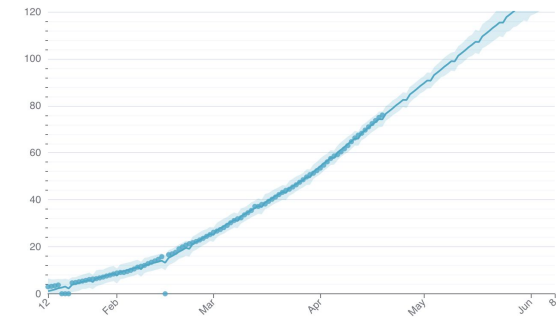
The data revolution

The main purpose
of data science is
insight

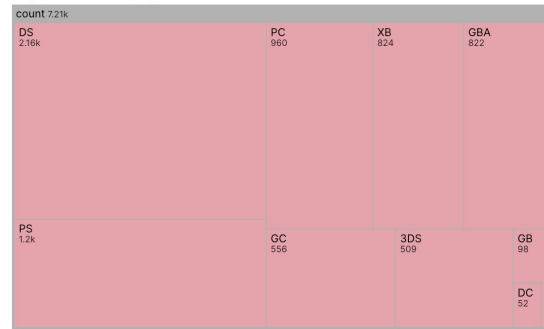
Data + Visualization!

Effective data visualization is one of the best ways to get to **insight** from large, complex amounts of data

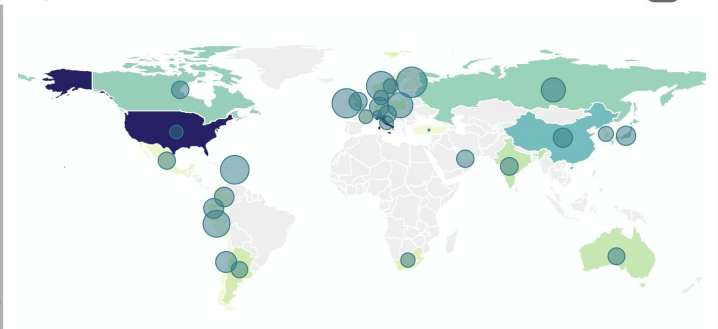
Massachusetts Total Vaccinations (per 100 people)



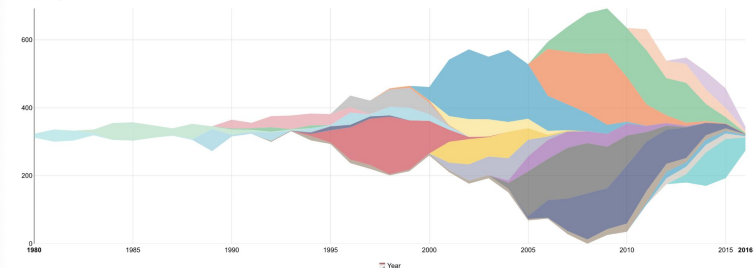
Top 10 Consoles, by # of Hit Games



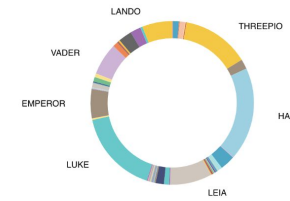
[Flights] World Destination/Price



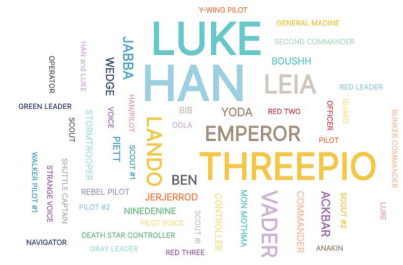
Global Sales per Console



Episode 6

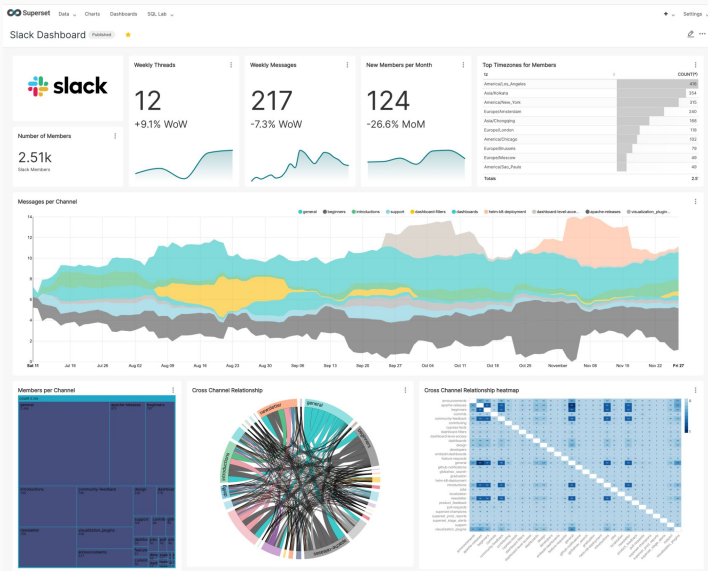


Episode 6

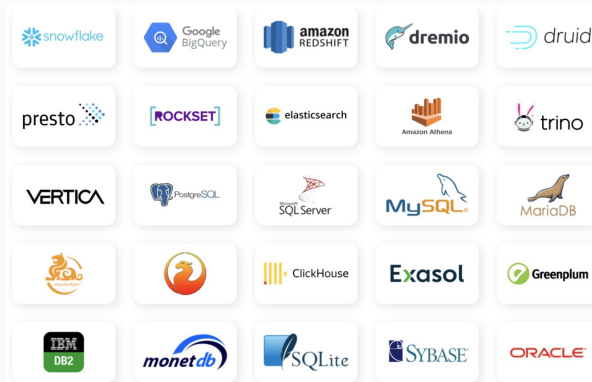


What is Superset & how does it work?

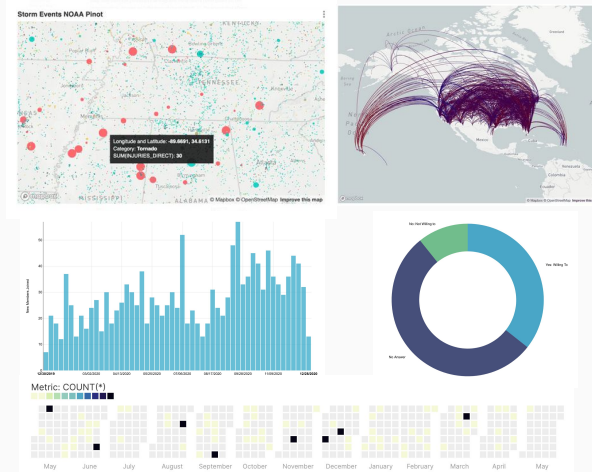
What is Apache Superset?



Modern open source BI platform



Works with nearly any SQL speaking data engine



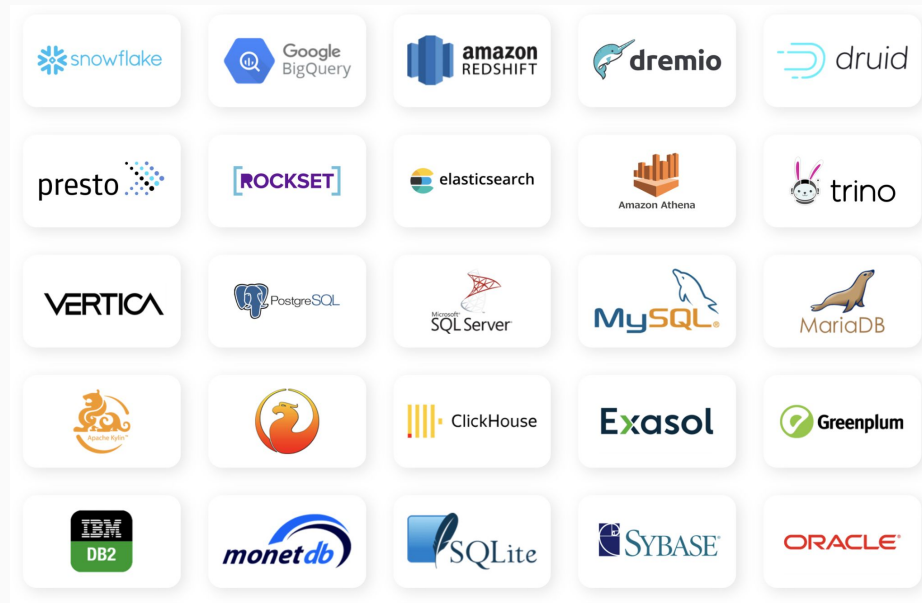
Large diversity of charts

Support for SQL Speaking Databases

Superset can query data from nearly any SQL speaking database

Requirements:

- Python DB-API 2.0 driver
 - <https://www.python.org/dev/peps/pep-0249/>
- SQLAlchemy Dialect
 - Incomplete list here: <https://docs.sqlalchemy.org/en/14/dialects/>



[Blog Post: Building New Database Connectors for Superset](#)

SQL Lab

Browser based SQL IDE (focused on writing analytical queries)

- Explore data
- Sculpt data for visualization
- Save transformed data as virtual datasets, db VIEWS, or db TABLEs

The screenshot displays the SQL Lab interface. At the top, the database is set to 'postgresql' and the schema to 'demo'. The query editor contains the SQL statement: `SELECT * FROM demo."Original Trilogy Scripts"`. Below the editor, there are buttons for 'RUN', 'LIMIT: 100', a timer showing '00:00:00.64', 'SAVE AS', and 'COPY LINK'. The results section shows a table with columns 'LineNumber', 'Character', and 'Line'. A warning message indicates '100 rows returned'. The table contains the following data:

LineNumber	Character	Line
0	THREEPIO	Did you hear that? They've shut down the main reactor. We'll be destroyed for sure.
1	THREEPIO	We're doomed!
2	THREEPIO	There'll be no escape for the Princess this time.
3	THREEPIO	What's that?
4	THREEPIO	I should have known better than to trust the logic of a half-sized thermocapsulary c
5	LUKE	Hurry up! Come with me! What are you waiting for?! Get in gear!
6	THREEPIO	Artoo! Artoo-Detoo, where are you?
7	THREEPIO	At last! Where have you been?
8	THREEPIO	They're heading in this direction. What are we going to do? We'll be sent to the spic
9	THREEPIO	Wait a minute, where are you going?

Superset Semantic Layer

```
1 SELECT unnest(string_to_array("Line", ' ')) as Tokens, *
2 FROM demo."Original Trilogy Scripts"
```

RUN LIMIT: 100 00:00:00.00 SAVE AS

RESULTS QUERY HISTORY PREVIEW: 'ORIGINAL TRILOGY SCRIPTS'

EXPLORE DOWNLOAD TO CSV COPY TO CLIPBOARD Filter results

100 rows returned

tokens	LineNumber	Character	Line
Did	0	THREEPIO	Did you hear that? They've shu
you	0	THREEPIO	Did you hear that? They've shu
hear	0	THREEPIO	Did you hear that? They've shu
that?	0	THREEPIO	Did you hear that? They've shu



Edit Dataset Split Tokens

Be careful. Changing these settings will affect all charts using this dataset, including charts owned by other people.

SOURCE METRICS (1) COLUMNS (5) CALCULATED COLUMNS (0) SETTINGS

Physical (table or view) Virtual (SQL)

VIRTUAL

Database: postgresql community-data-rds

Schema: demo

DATASET NAME

Split Tokens

SQL

```
1 SELECT unnest(string_to_array("Line", ' ')) as Tokens, *
2 FROM demo."Original Trilogy Scripts"
```

Click the lock to prevent further changes.

USE LEGACY DATASOURCE EDITOR CANCEL SAVE

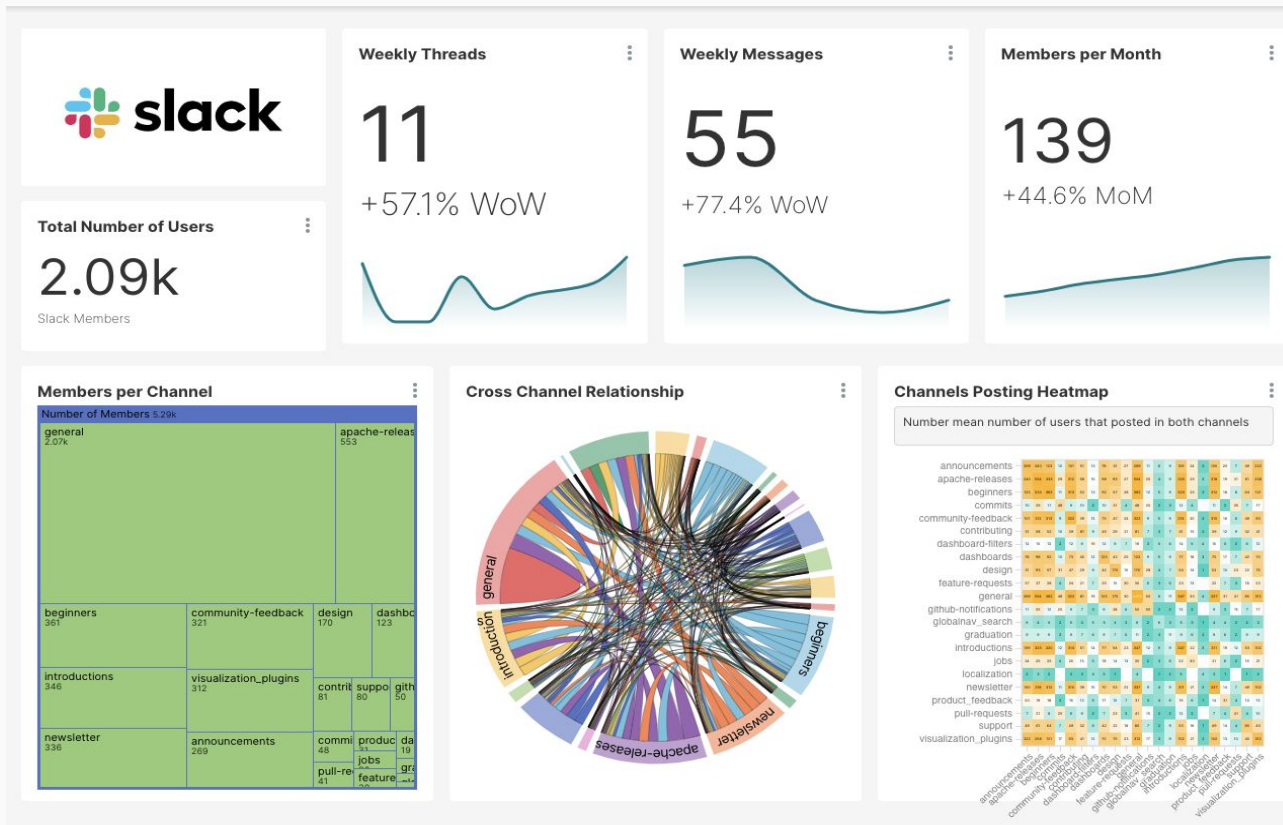
No Code Chart Builder (Explore)

The screenshot displays a 'No Code Chart Builder' interface. On the left, a 'Dataset' panel shows 'demo.Original Trilogy Scripts' with 'DATA' and 'CUSTOMIZE' tabs. The 'CUSTOMIZE' tab is active, showing configuration options for 'Chart type' (Table), 'Time' (Day), 'Query' (AGGREGATE), 'METRICS' (COUNT(*)), and 'SORT BY' (DESCENDING). The main area shows a bar chart titled 'Episode 4: Chattiest Characters' with 60 rows. The chart displays the count of characters for each character in Episode 4. The data is as follows:

Character	COUNT(*)
LUKE	254
HAN	153
THREEPIO	119
BEN	82
LEIA	57
VADER	41
RED LEADER	37
BIGGS	34
TARKIN	28
OWEN	25
TROOPER	19
WEDGE	14
GOLD LEADER	14
OFFICER	11
RED TEN	8
GOLD FIVE	7
JABBA	6
AUNT BERU	6
FIRST TROOPER	6
DEATH STAR INTERCOM VOICE	6
Totals	1.01k

Below the chart, a 'Data' table view shows the same data in a tabular format. The interface includes a search bar and pagination controls.

Slack Community Dashboard



[Build your own Slack dashboard](#)

Introducing ClickHouse and Superset

ClickHouse is a new option for analytic services

An industrial strength, Apache 2.0 SQL data warehouse



Installs on a
laptop in 60
seconds

Runs in cloud,
containers, and
bare metal

Answers
queries in
milliseconds

ClickHouse performance meets or exceeds proprietary SQL data warehouses

Portable C++ binary

Advanced SQL implementation

Column storage with high compression

Distributed, vectorized queries

Built-in sharding and replication

Scales from laptop to 100s of nodes

Popular Apache 2.0 licensing

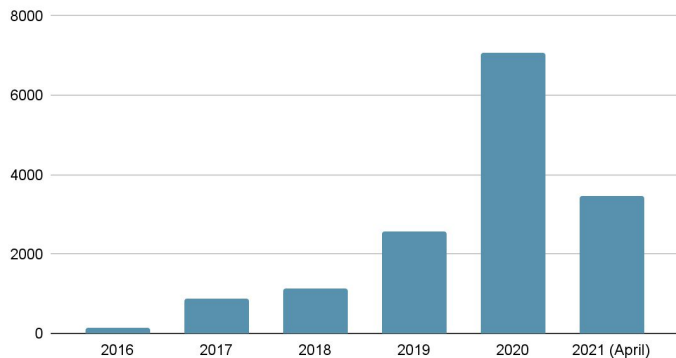
We use ClickHouse extensively and it's been great.

John Graham-Cumming, CloudFlare CTO
Hacker News, 2020

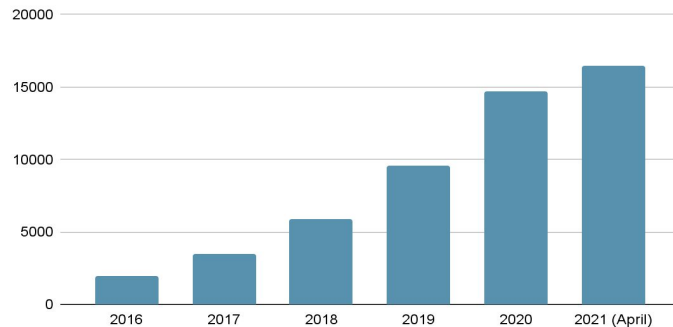


ClickHouse grew from an in-house Yandex project to an international community that rivals ElasticSearch

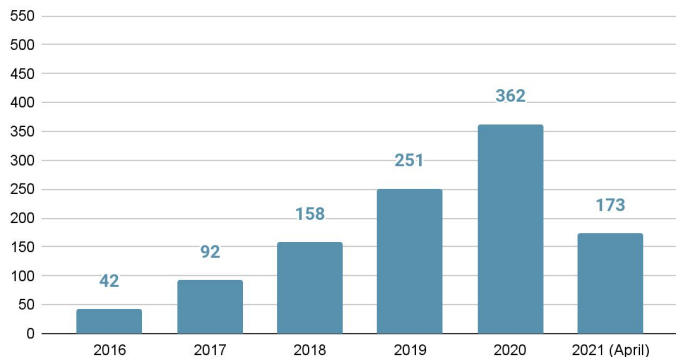
ClickHouse Pull Requests



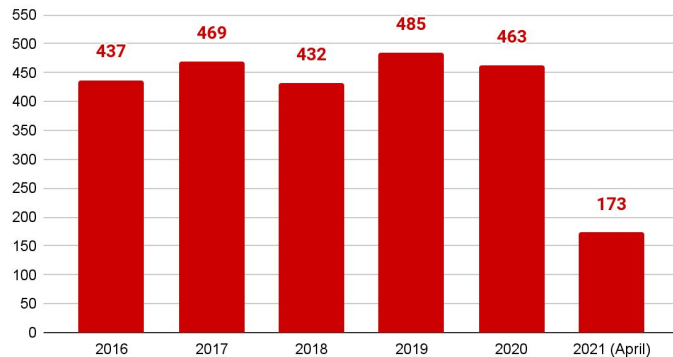
ClickHouse Github Watchers (aka Stars)



ClickHouse Unique Github Contributors

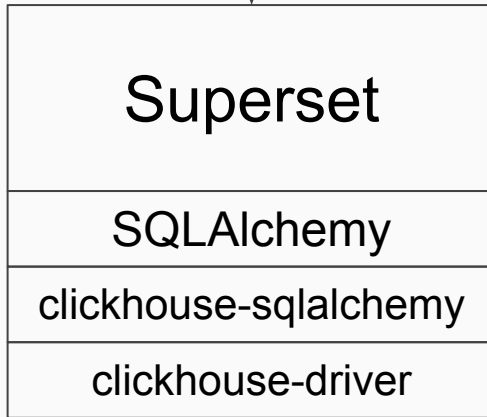


ElasticSearch Unique Github Contributors



Superset connection to ClickHouse

Web Browser



ClickHouse

(Clear)
:9000
:9440
(TLS)



ClickHouse SQLAlchemy Drivers (An aside)



sqlalchemy-clickhouse
Apache 2.0
Developed by Marek Vavrusa

- Currently documented in Superset
- Uses ClickHouse HTTP Interface
- No TLS support
- Current pypi.org release: 0.1.5.post0, Aug 9 2018



clickhouse-sqlalchemy
Apache 2.0
Developed by Konstantin Lebedev


- Supported by Altinity
- Uses ClickHouse Native TCP
- TLS support
- Bug fixes for Superset
- Current pypi.org release: 0.1.6, Mar 15 2021

Setting up Superset (Ubuntu)

```
# Install Superset
export FLASK_APP=superset
pip install apache-superset
superset db upgrade
superset fab create-admin
superset load_examples
superset init

# Add ClickHouse driver
pip install clickhouse-sqlalchemy

# Start Superset
superset run -p 8088 --with-threads --reload --debugger
```



ClickHouse connection strings

SQLAlchemy URL format:

```
clickhouse+native://[user:pw]@host[:port]/database[?options...]
```

ClickHouse on localhost (e.g., your laptop)

```
clickhouse+native://localhost/default
```

ClickHouse public endpoint:

```
clickhouse+native://demo:demo@github.demo.trial.altinity.cloud/default?secure=true
```

Database connection page

☰ Add Database ✕

CONNECTION * PERFORMANCE SQL LAB SETTINGS SECURITY EXTRA

DATABASE NAME*

SQLALCHEMY URI*

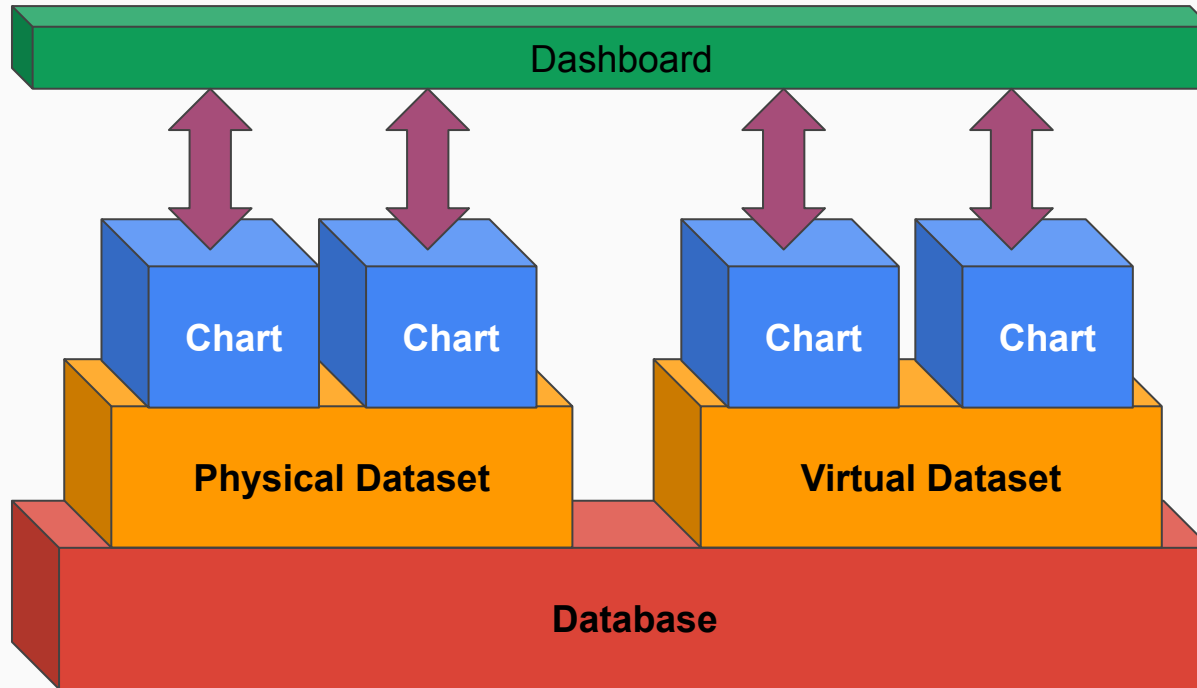
[TEST CONNECTION](#)

Refer to the [SQLAlchemy docs](#) for more information on how to structure your URI.

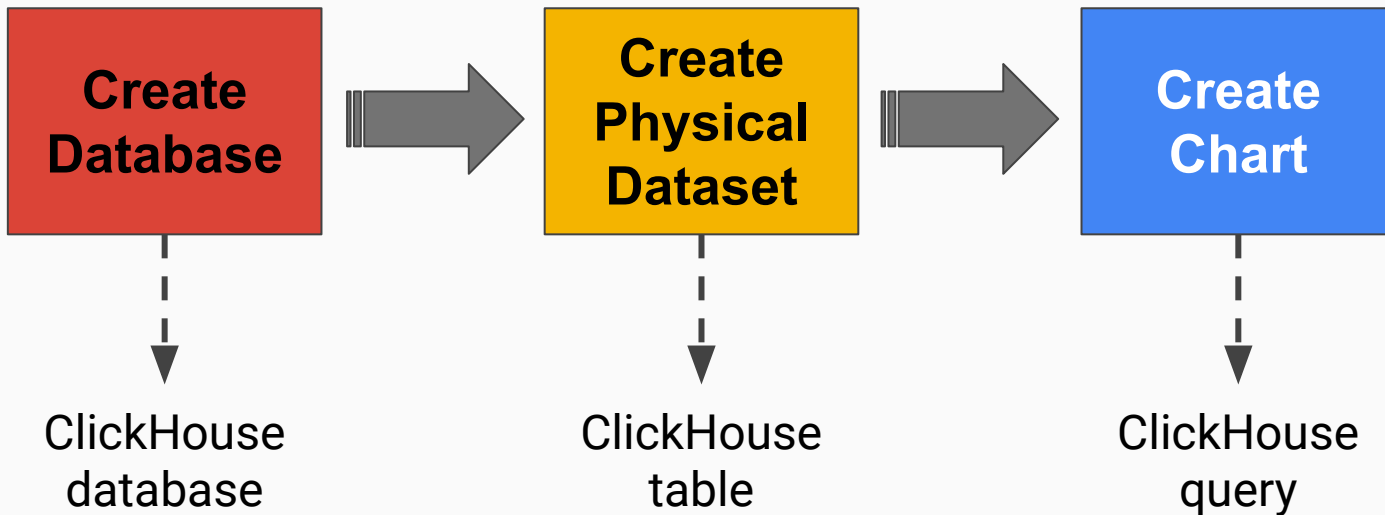
[CANCEL](#) [ADD](#)

Fun with ClickHouse Data

Superset dashboard organization



Creating a time series chart



Physical dataset creation page

Database

Add dataset

DATASOURCE

Database: clickhouse clickhouse-public

SCHEMA

Schema: default

SEE TABLE SCHEMA 5 IN DEFAULT

TABLE

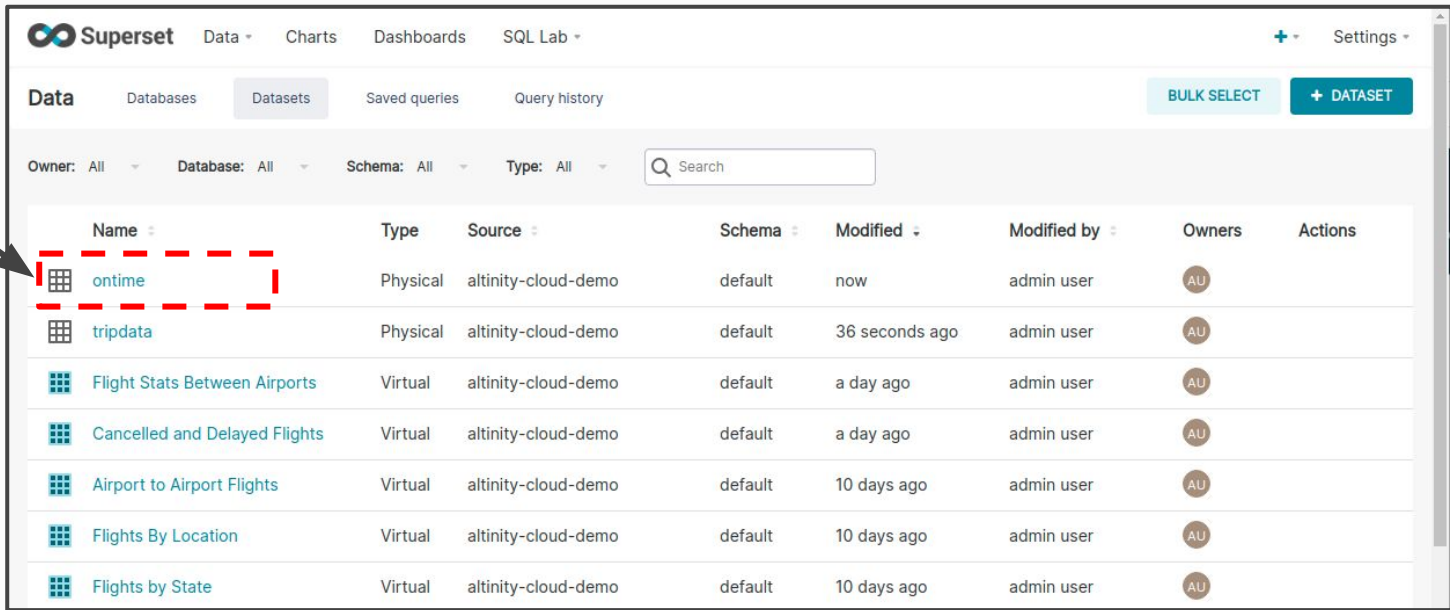
ontime

CANCEL ADD

ClickHouse Connection

Table

Starting a chart from a dataset



The screenshot shows the Superset web interface. At the top, there are navigation tabs for 'Data', 'Charts', 'Dashboards', and 'SQL Lab'. Below this, there are sub-tabs for 'Databases', 'Datasets', 'Saved queries', and 'Query history'. A search bar is present with the text 'Search'. The main content area displays a table of datasets with the following columns: Name, Type, Source, Schema, Modified, Modified by, Owners, and Actions. The 'ontime' dataset is highlighted with a red dashed box, and a yellow box with the word 'Dataset' and an arrow points to it.

Name	Type	Source	Schema	Modified	Modified by	Owners	Actions
ontime	Physical	altinity-cloud-demo	default	now	admin user	AU	
tripdata	Physical	altinity-cloud-demo	default	36 seconds ago	admin user	AU	
Flight Stats Between Airports	Virtual	altinity-cloud-demo	default	a day ago	admin user	AU	
Cancelled and Delayed Flights	Virtual	altinity-cloud-demo	default	a day ago	admin user	AU	
Airport to Airport Flights	Virtual	altinity-cloud-demo	default	10 days ago	admin user	AU	
Flights By Location	Virtual	altinity-cloud-demo	default	10 days ago	admin user	AU	
Flights by State	Virtual	altinity-cloud-demo	default	10 days ago	admin user	AU	

Time series chart from physical dataset

The screenshot shows the Superset interface with the following configuration for a 'Time-series Chart':

- Visualization Type:** Time-series Chart
- Time Column:** FlightDate
- Time Grain:** month
- Time Range:** previous calendar year
- Metrics:** COUNT(*)
- Group By:** Carrier

The chart displays a stacked bar chart titled 'Flights Per Month' from Dec 2019 to Oct 2020. The y-axis represents the number of flights, ranging from 0 to 700k. The x-axis shows months. The bars are stacked by carrier, with the top-most carrier in each bar being the most frequent.

timestamp	9E	AA	AS	B6	DL	EV	F
2020-01-01 00:00:00	23068	76276	21110	24709	80067	10564	1
2020-02-01 00:00:00	21623	72402	19515	23527	75446	11404	1
2020-03-01 00:00:00	25996	80330	21380	25657	86291	12412	1
2020-04-01 00:00:00	8282	42262	5064	6996	26002	4656	2

Chart Type

Time Dimension

Metric

Group

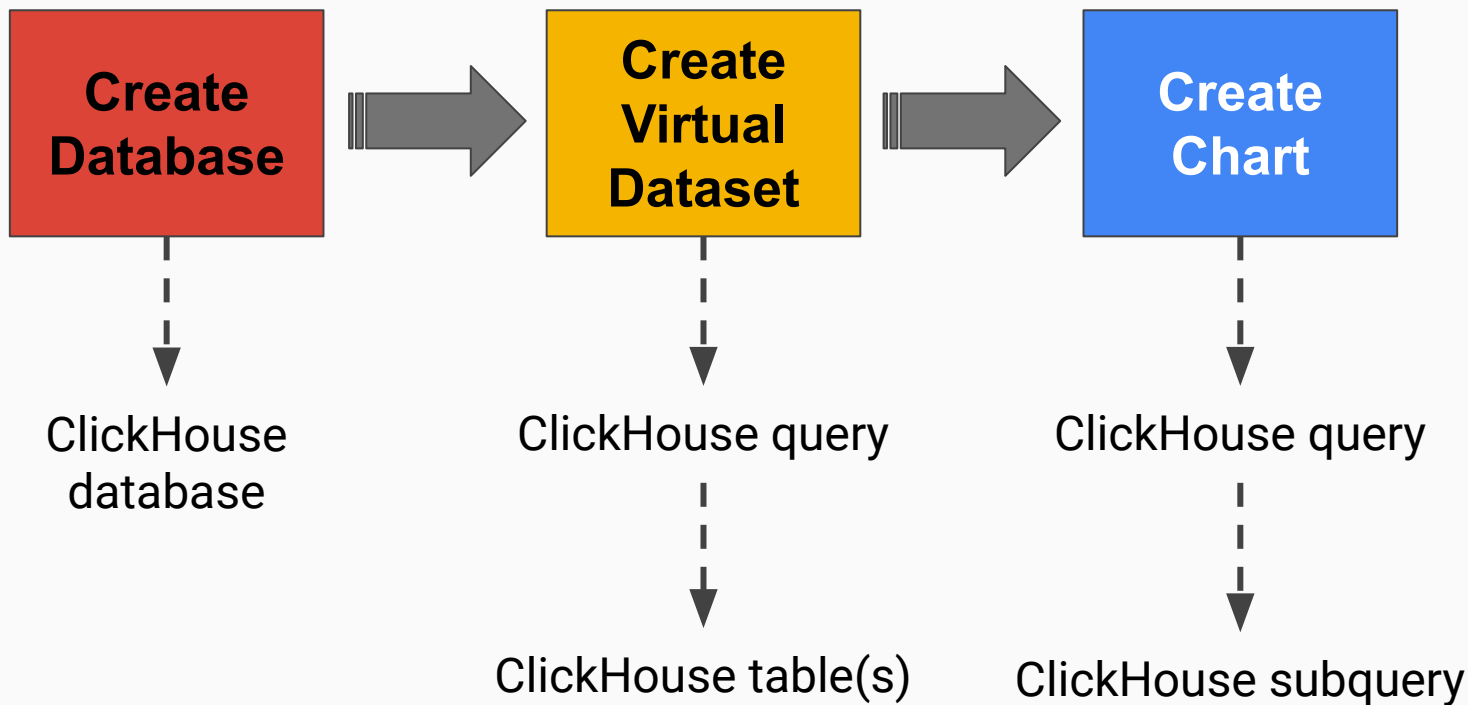
Time series query is generated

The screenshot displays the Superset interface with a 'View query' dialog box open. The dialog box contains the following SQL query:

```
SELECT toStartOfMonth(toDateTime("FlightDate")) AS __timestamp,  
       "Carrier" AS "Carrier",  
       COUNT(*) AS count  
FROM "default".ontime  
WHERE "FlightDate" ≥ toDate('2020-01-01')  
      AND "FlightDate" < toDate('2021-01-01')  
GROUP BY "Carrier",  
         toStartOfMonth(toDateTime("FlightDate"))  
LIMIT 10000;
```

The background shows a chart titled 'Flights Per Month' with a time grain of 'month'. The chart displays data for the months of Dec 2019, Feb 2020, Mar 2020, Apr 2020, May 2020, Jun 2020, Jul 2020, Aug 2020, Sep 2020, and Oct 2020. The chart is a stacked bar chart showing the number of flights per month, grouped by carrier.

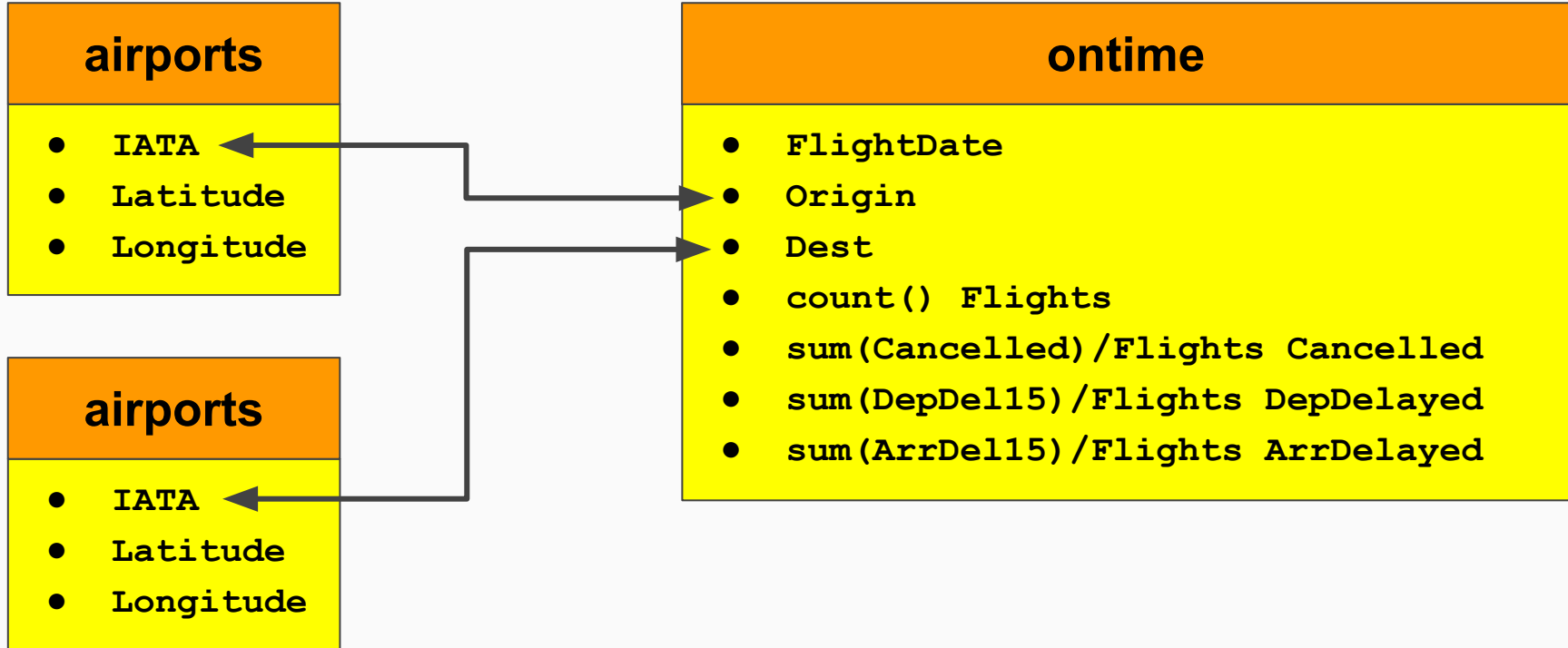
Creating a chart on a virtual dataset



A query that answers multiple questions

```
SELECT FlightDate, Origin, Dest,  
       oa.Latitude as Origin_Latitude, oa.Longitude AS Origin_Longitude,  
       od.Latitude as Dest_Latitude, od.Longitude AS Dest_Longitude,  
       Flights, Cancelled, DepDelayed, ArrDelayed  
FROM (  
  SELECT FlightDate, Origin, Dest, count() Flights,  
         sum(Cancelled)/Flights Cancelled, sum(DepDel15)/Flights DepDelayed,  
         sum(ArrDel15)/Flights ArrDelayed  
  FROM ontime  
  GROUP BY FlightDate, Origin, Dest ORDER BY FlightDate, Origin, Dest  
) AS o  
INNER JOIN airports AS oa ON toString(o.Origin) = oa.IATA  
INNER JOIN airports AS od ON toString(o.Dest) = od.IATA
```

Using JOIN to add airport LAT/LONG



Build, run, and save query in SQL Lab

Database

Tables

Query

Save Query

Results

The screenshot shows the Superset SQL Lab interface. On the left, the 'Database' dropdown is set to 'clickhouse' and 'clickhouse-public'. Below it, the 'Tables' section shows the 'airports' table schema with columns like AirportID, Name, City, Country, IATA, ICAO, Latitude, Longitude, Altitude, Timezone, DST, Tz, Type, and Source. The 'ontime' table schema is also visible with columns like Year, Quarter, Month, DayofMonth, DayOfWeek, FlightDate, and UniqueCarrier. The central query editor contains a SQL query for flight statistics. Below the query editor, the 'RUN' button is highlighted with a red dashed box, and the 'SAVE' button is also highlighted. The 'RESULTS' section shows a table with columns Year, Quarter, Month, DayofMonth, DayOfWeek, FlightDate, and UniqueCarrier, containing data for the year 1987.

```
1 SELECT FlightDate, Origin,
2       oa.Latitude as Origin_Latitude, oa.Longitude AS Origin_Longitude,
3       od.Latitude as Dest_Latitude, od.Longitude AS Dest_Longitude,
4       Cancelled, DepDelayed, ArrDelayed
5 FROM (
6   SELECT FlightDate, Origin, Dest,
7         count() Flights,
8         sum(Cancelled)/Flights Cancelled,
9         sum(DepDel15)/Flights DepDelayed,
10        sum(ArrDel15)/Flights ArrDelayed
11 FROM ontime
12 GROUP BY FlightDate, Origin, Dest ORDER BY FlightDate, Origin, Dest
13 ) AS o
14 INNER JOIN airports AS oa ON toString(o.Origin) = oa.IATA
15 INNER JOIN airports AS od ON toString(o.Dest) = od.IATA
```

Year	Quarter	Month	DayofMonth	DayOfWeek	FlightDate	UniqueCarrier
1987	4	10	1	4	1987-10-01	AA
1987	4	10	1	4	1987-10-01	AA
1987	4	10	1	4	1987-10-01	AA
1987	4	10	1	4	1987-10-01	AA
1987	4	10	1	4	1987-10-01	AA
1987	4	10	1	4	1987-10-01	AA

Use EXPLORE to save as dataset

Superset Database: clickhouse clickhouse

Select a schema (1)

SEE TABLE SCHEMA

Select table

airports

AirportID String
Name String
City String
Country String
IATA String
ICAO String
Latitude FLOAT
Longitude FLOAT
Altitude Int32
Timezone FLOAT
DST String

Save or Overwrite Dataset

Save this query as a virtual dataset to continue exploring

Save as new Flight Stats between Airports 05/02/2021 09:32:10

Overwrite existing Select or type dataset name

SAVE & EXPLORE

RUN LIMIT: 1 000 00:00:10.09 SAVE COPY LINK

RESULTS QUERY HISTORY PREVIEW: 'AIRPORTS' PREVIEW: 'ONTIME'

EXPLORE .CSV CLIPBOARD Filter results

FlightDate	Origin	Origin_Latitude	Origin_Longitude	Dest_Latitude
1987-10-01	ABE	40.652099609375	-75.44080352783203	33.6366996765136

Creating a deck.gl Arc chart

Chart Type*

* Requires a Mapbox token -- See docs

Time Dimension

Lat/Long

Filter

The screenshot shows the Superset interface for configuring a chart. The chart type is set to 'deck.gl Arc'. The time dimension is set to 'FlightDate' with a time range of 'previous calendar year'. The chart is configured to use 'Origin_Longitude' and 'Origin_Latitude' for the start longitude and latitude, and 'Dest_Longitude' and 'Dest_Latitude' for the end longitude and latitude. The row limit is set to 5000, and the 'IGNORE NULL LOCATIONS' checkbox is checked. The filter is set to 'Origin = 'SFO''. The chart displays flight paths from SFO to various destinations, including Mexico, Cuba, and the United States.

Dataset: default.Flight Stats Between ...

Chart type: VISUALIZATION TYPE: deck.gl Arc

Time: TIME COLUMN: FlightDate, TIME RANGE: previous calendar year

Query: START LONGITUDE & LATITUDE: Origin_Longitude | Origin_Latitude, END LONGITUDE & LATITUDE: Dest_Longitude | Dest_Latitude

ROW LIMIT: 5000

IGNORE NULL LOCATIONS

FILTERS: Origin = 'SFO'

Flights from SFO

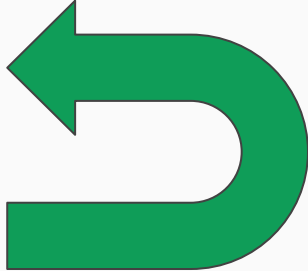
5k rows 00:00:01.93

mapbox

Origin_Longitude	Dest_Latitude	Origin_Latitude	Dest_
-122.375	35.040199279785156	37.61899948120117	-106.1
-122.375	40.97809982299805	37.61899948120117	-124.1

How Superset queries virtual datasets

```
SELECT "Origin_Longitude" AS "Origin_Longitude",  
       "Dest_Latitude" AS "Dest_Latitude",  
       "Origin_Latitude" AS "Origin_Latitude",  
       "Dest_Longitude" AS "Dest_Longitude"  
FROM  
  (  
    Dataset subquery  
  ) AS expr_qry  
WHERE "FlightDate" >= toDate('2020-01-01')  
      AND "FlightDate" < toDate('2021-01-01')  
      AND "Origin" = 'SFO'  
      AND "Dest_Latitude" IS NOT NULL AND "Dest_Longitude" IS NOT NULL  
      AND "Origin_Latitude" IS NOT NULL AND "Origin_Longitude" IS NOT NULL  
LIMIT 5000;
```



Filters pushed down to base table

Creating Word Cloud chart

Chart Type

Time Dimension

Series

Metric

Limit & Sort Order

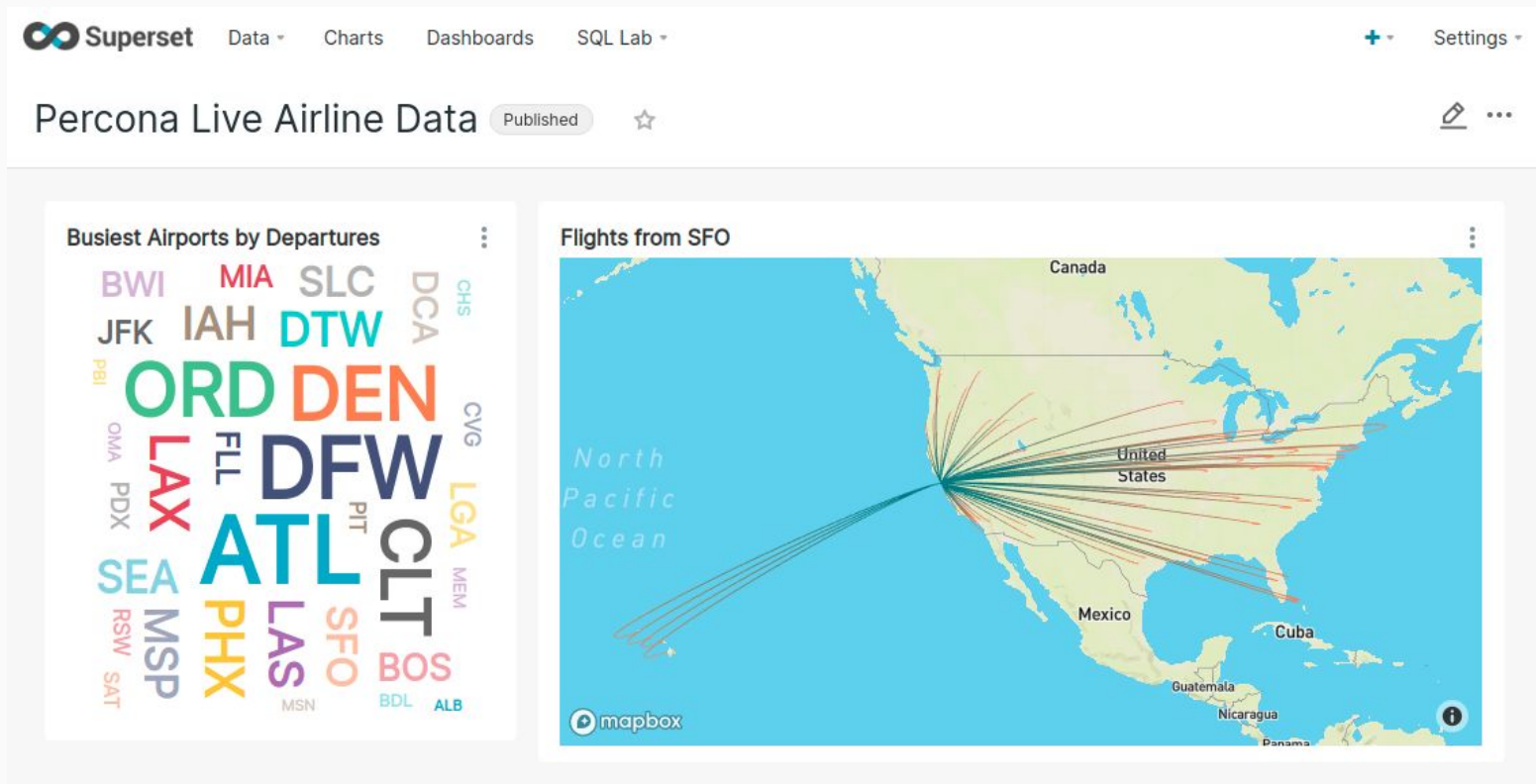
The screenshot shows the Superset interface for creating a Word Cloud chart. The configuration panel on the left is annotated with red dashed boxes and arrows pointing to the labels on the left:

- Chart Type:** Points to the 'Word Cloud' visualization type selection.
- Time Dimension:** Points to the 'FlightDate' time column selection.
- Series:** Points to the 'Origin' series selection.
- Metric:** Points to the 'SUM(Flights)' metric selection.
- Limit & Sort Order:** Points to the '100' row limit and 'SORT BY METRIC' checkbox.

The visualization on the right, titled 'Busiest Airports by Departures', shows a word cloud of airport codes. The largest words are DEN, ORD, ATL, and DFW. Other visible words include BOS, IAH, SEA, LAS, PHX, BWI, LAX, SFO, and many others.

Origin	SUM(Flights)
ATL	209870
DFW	187234
ORD	164288

Putting charts in a dashboard



Building from here

- There are *many* more charts available in Superset
- You can build complex dashboards
- Superset caching and clustering can amortize query overhead

Plans for the future

Roadmap for Superset and ClickHouse

- Improve clickhouse-sqlalchemy driver
 - There are still a few bugs :/
- Build out and document usage examples for users
 - [Blog Post on Clickhouse <> Superset](#)
- Full integration between Preset Cloud and Altinity.Cloud
 - Ensure both services available in common set of AWS regions
 - Automatic setup of AWS PrivateConnect between services

Questions?

Thank you!!

Srini Kadamati

Preset

<https://preset.io>

Robert Hodges

Altinity

<https://altinity.com>