

18 ДЗ - Интеграции с BI-инструментами

Собираем Dockerfile для clickhouse.

Dockerfile:

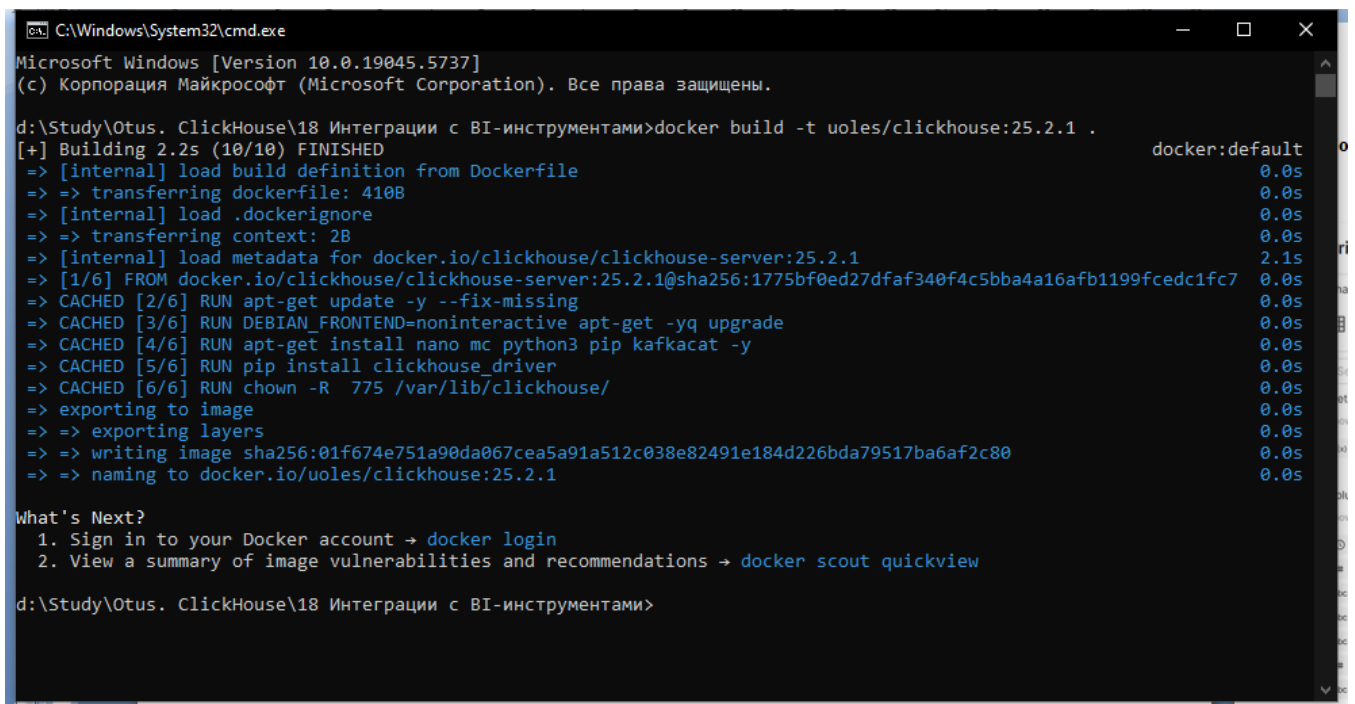
```
FROM clickhouse/clickhouse-server:25.2.1
MAINTAINER Maksim Kulikov max.uoles@rambler.ru
```

```
RUN apt-get update -y --fix-missing
RUN DEBIAN_FRONTEND=noninteractive apt-get -yq upgrade
RUN apt-get install nano mc python3 pip kafkacat -y
RUN pip install clickhouse_driver
```

```
EXPOSE 8123 9000
ENTRYPOINT ["/entrypoint.sh"]
```

Собираем образ командой:

```
docker build -t uoles/clickhouse:25.2.1 .
```



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.5737]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

d:\Study\Otus. ClickHouse\18 Интеграции с BI-инструментами>docker build -t uoles/clickhouse:25.2.1 .
[+] Building 2.2s (10/10) FINISHED
=> [internal] load build definition from Dockerfile                                docker:default 0.0s
=> => transferring dockerfile: 410B                                              0.0s
=> [internal] load .dockerignore                                                 0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/clickhouse/clickhouse-server:25.2.1 2.1s
=> [1/6] FROM docker.io/clickhouse/clickhouse-server:25.2.1@sha256:1775bf0ed27dfaf340f4c5bba4a16afb1199fcdcd1fc7 0.0s
=> CACHED [2/6] RUN apt-get update -y --fix-missing                             0.0s
=> CACHED [3/6] RUN DEBIAN_FRONTEND=noninteractive apt-get -yq upgrade           0.0s
=> CACHED [4/6] RUN apt-get install nano mc python3 pip kafkacat -y             0.0s
=> CACHED [5/6] RUN pip install clickhouse_driver                             0.0s
=> CACHED [6/6] RUN chown -R 775 /var/lib/clickhouse/                          0.0s
=> exporting to image                                                           0.0s
=> => exporting layers                                                         0.0s
=> => writing image sha256:01f674e751a90da067cea5a91a512c038e82491e184d226bda79517ba6af2c80 0.0s
=> => naming to docker.io/uoles/clickhouse:25.2.1                             0.0s

What's Next?
 1. Sign in to your Docker account → docker login
 2. View a summary of image vulnerabilities and recommendations → docker scout quickview

d:\Study\Otus. ClickHouse\18 Интеграции с BI-инструментами>
```

Поднимаем superset с clickhouse.

Скачиваем репозиторий superset:

```
git clone https://github.com/apache/superset.git
cd superset
git checkout 2.1.0
```

Редактируем файл docker-compose-non-dev.yml и сохраняем как docker-compose-clickhouse.yml:

- добавил установку clickhouse-connect в superset
- добавил разворачивание clickhouse-server
- добавил зависимость от clickhouse-server для superset и линк

docker-compose-clickhouse.yml:

```
x-superset-image: &superset-image apache/superset:${TAG:-latest-dev}
x-superset-depends-on: &superset-depends-on
- db
- redis
x-superset-only-depends-on: &superset-only-depends-on
- db
- redis
- clickhouse-server
x-superset-volumes: &superset-volumes
# /app/pythonpath_docker will be appended to the PYTHONPATH in the final container
- ./docker:/app/docker
- superset_home:/app/superset_home
```

version: "3.7"

services:

redis:

```
image: redis:7
container_name: superset_cache
restart: unless-stopped
volumes:
- redis:/data
```

db:

```
env_file: docker/.env-non-dev
image: postgres:14
container_name: superset_db
restart: unless-stopped
volumes:
- db_home:/var/lib/postgresql/data
```

superset:

```
env_file: docker/.env-non-dev
image: *superset-image
container_name: superset_app
command: [sh, -c, "pip install clickhouse-connect && /app/docker/docker-bootstrap.sh app-gunicorn"]
user: "root"
restart: unless-stopped
ports:
- 8088:8088
depends_on: *superset-only-depends-on
volumes: *superset-volumes
links:
- clickhouse-server
```

superset-init:

```
image: *superset-image
container_name: superset_init
command: ["/app/docker/docker-init.sh"]
env_file: docker/.env-non-dev
depends_on: *superset-depends-on
user: "root"
volumes: *superset-volumes
healthcheck:
disable: true
```

superset-worker:

```
image: *superset-image
container_name: superset_worker
command: ["/app/docker/docker-bootstrap.sh", "worker"]
env_file: docker/.env-non-dev
```

```
restart: unless-stopped
depends_on: *superset-depends-on
user: "root"
volumes: *superset-volumes
healthcheck:
  test: ["CMD-SHELL", "celery inspect ping -A superset.tasks.celery_app:app -d celery@$${HOSTNAME}"]
```

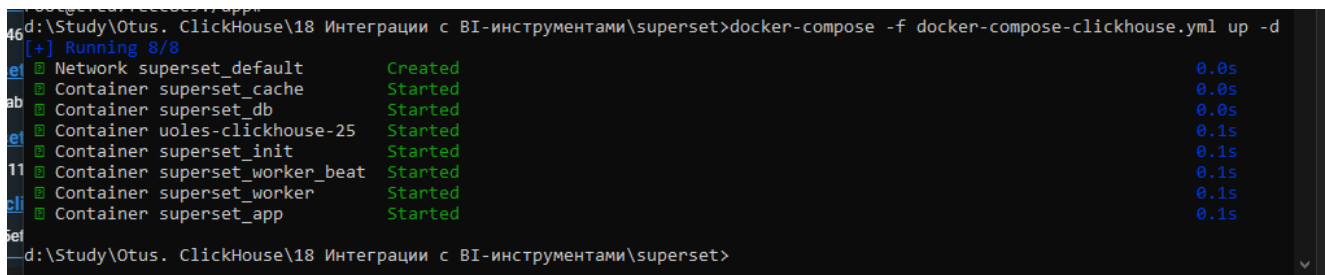
```
superset-worker-beat:
  image: *superset-image
  container_name: superset_worker_beat
  command: ["/app/docker/docker-bootstrap.sh", "beat"]
  env_file: docker/.env-non-dev
  restart: unless-stopped
  depends_on: *superset-depends-on
  user: "root"
  volumes: *superset-volumes
  healthcheck:
    disable: true
```

```
clickhouse-server:
  container_name: uoles-clickhouse-25
  image: uoles/clickhouse:25.2.1
  environment:
    CLICKHOUSE_DB: my_database
    CLICKHOUSE_USER: username
    CLICKHOUSE_DEFAULT_ACCESS_MANAGEMENT: 1
    CLICKHOUSE_PASSWORD: password
  ports:
    - "18123:8123"
    - "19000:9000"
  ulimits:
    nofile:
      soft: 262144
      hard: 262144
```

```
volumes:
  superset_home:
    external: false
  db_home:
    external: false
  redis:
    external: false
```

Поднимаем приложения командой:

```
docker-compose -f docker-compose-clickhouse.yml up -d
```



```
d:\Study\Otus. ClickHouse\18 Интеграции с BI-инструментами\superset>docker-compose -f docker-compose-clickhouse.yml up -d
[+] Running 8/8
et Network superset_default      Created                                0.0s
ab Container superset_cache      Started                               0.0s
et Container superset_db        Started                               0.0s
et Container uoles-clickhouse-25 Started                               0.1s
et Container superset_init       Started                               0.1s
11 Container superset_worker_beat Started                               0.1s
cli Container superset_worker    Started                               0.1s
et Container superset_app        Started                               0.1s
d:\Study\Otus. ClickHouse\18 Интеграции с BI-инструментами\superset>
```

Проверяем в логах установку clickhouse-connect:

```
superset_app
apache/superset:latest-dev
5a45080d5546
8088:8088

Logs Inspect Bind mounts Exec Files Stats
2025-04-20 02:24:34 Collecting clickhouse-connect
2025-04-20 02:24:34 Downloading clickhouse_connect-0.8.17-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.4 kB)
2025-04-20 02:24:34 Requirement already satisfied: certifi in /usr/local/lib/python3.10/site-packages (from clickhouse-connect) (2024.2.2)
2025-04-20 02:24:34 Requirement already satisfied: urllib3>=1.26 in /usr/local/lib/python3.10/site-packages (from clickhouse-connect) (1.26.18)
2025-04-20 02:24:34 Requirement already satisfied: pytz in /usr/local/lib/python3.10/site-packages (from clickhouse-connect) (2024.1)
2025-04-20 02:24:34 Requirement already satisfied: zstandard in /usr/local/lib/python3.10/site-packages (from clickhouse-connect) (0.22.0)
2025-04-20 02:24:34 Collecting lz4 (from clickhouse-connect)
2025-04-20 02:24:34 Downloading lz4-4.4.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.8 kB)
2025-04-20 02:24:34 Downloading clickhouse_connect-0.8.17-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (979 kB)
2025-04-20 02:24:34 979.5/979.5 kB 7.6 MB/s eta 0:00:00
2025-04-20 02:24:34 Downloading lz4-4.4.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.3 MB)
2025-04-20 02:24:34 1.3/1.3 MB 19.6 MB/s eta 0:00:00
2025-04-20 02:24:35 Installing collected packages: lz4, clickhouse-connect
2025-04-20 02:24:35 Successfully installed clickhouse-connect-0.8.17 lz4-4.4.4
2025-04-20 02:24:36 Skipping local overrides
```

Создаем dataset в superset.

Connect a database

STEP 2 OF 3

Enter the required ClickHouse Connect (Superset) credentials

Need help? Learn more about connecting to ClickHouse Connect (Superset).

HOST * ⓘ

PORT

clickhouse-server

8123

DATABASE NAME

my_database

Copy the name of the database you are trying to connect to.

USERNAME

username

PASSWORD

password

DISPLAY NAME *

ClickHouse Connect (Superset)

Pick a nickname for how the database will display in Superset.

ADDITIONAL PARAMETERS

e.g. param1=value1¶m2=value2

Add additional custom parameters

☐ SSL ⓘ

BACKCONNECT

uk_price_paid

DATABASE

clickhousedb

ClickHouse C... ▾

SCHEMA

uk ▾

TABLE

uk_price_paid 🔍

uk_price_paid ✓



This table already has a dataset

[View Dataset](#)

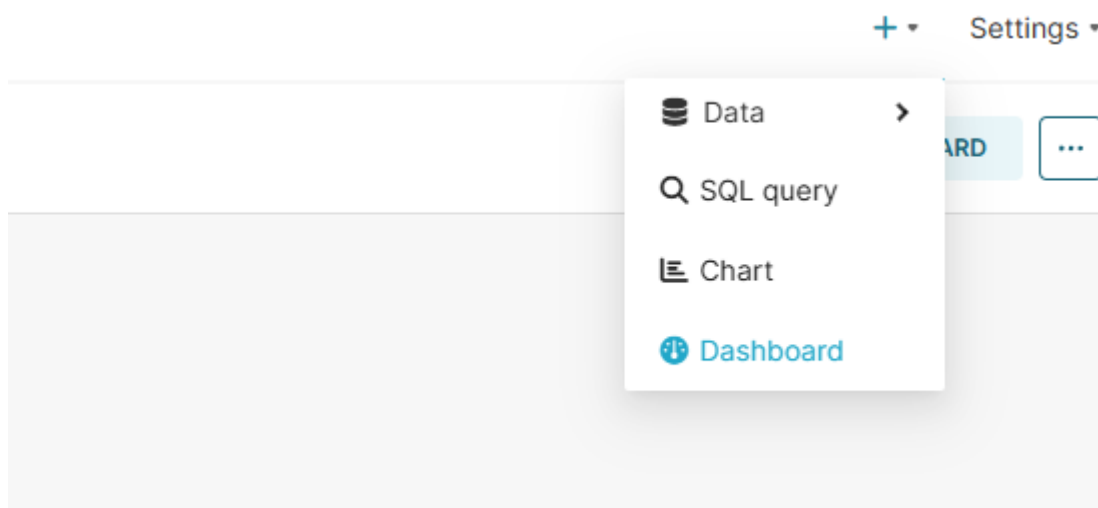
This table already has a dataset associated with it. You can only associate one dataset with a table.

uk_price_paid

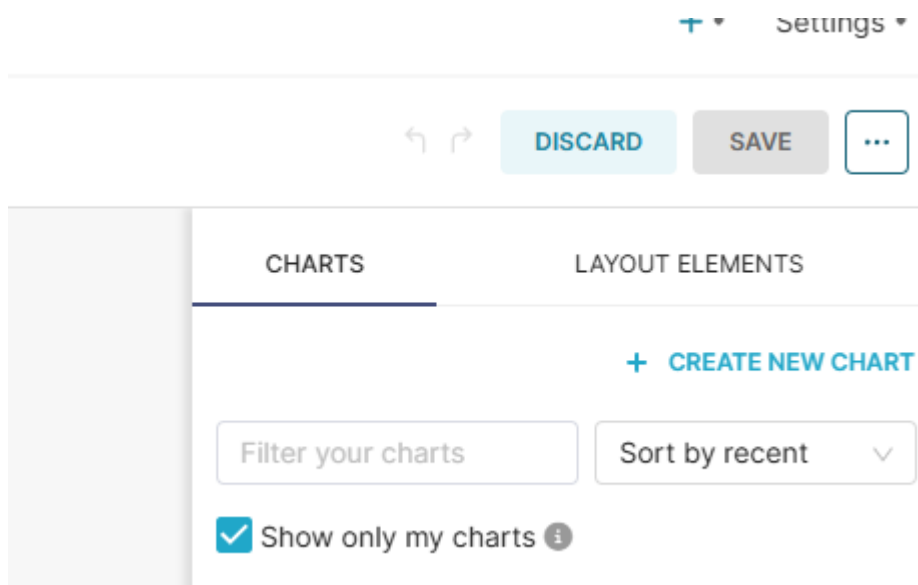
Table columns

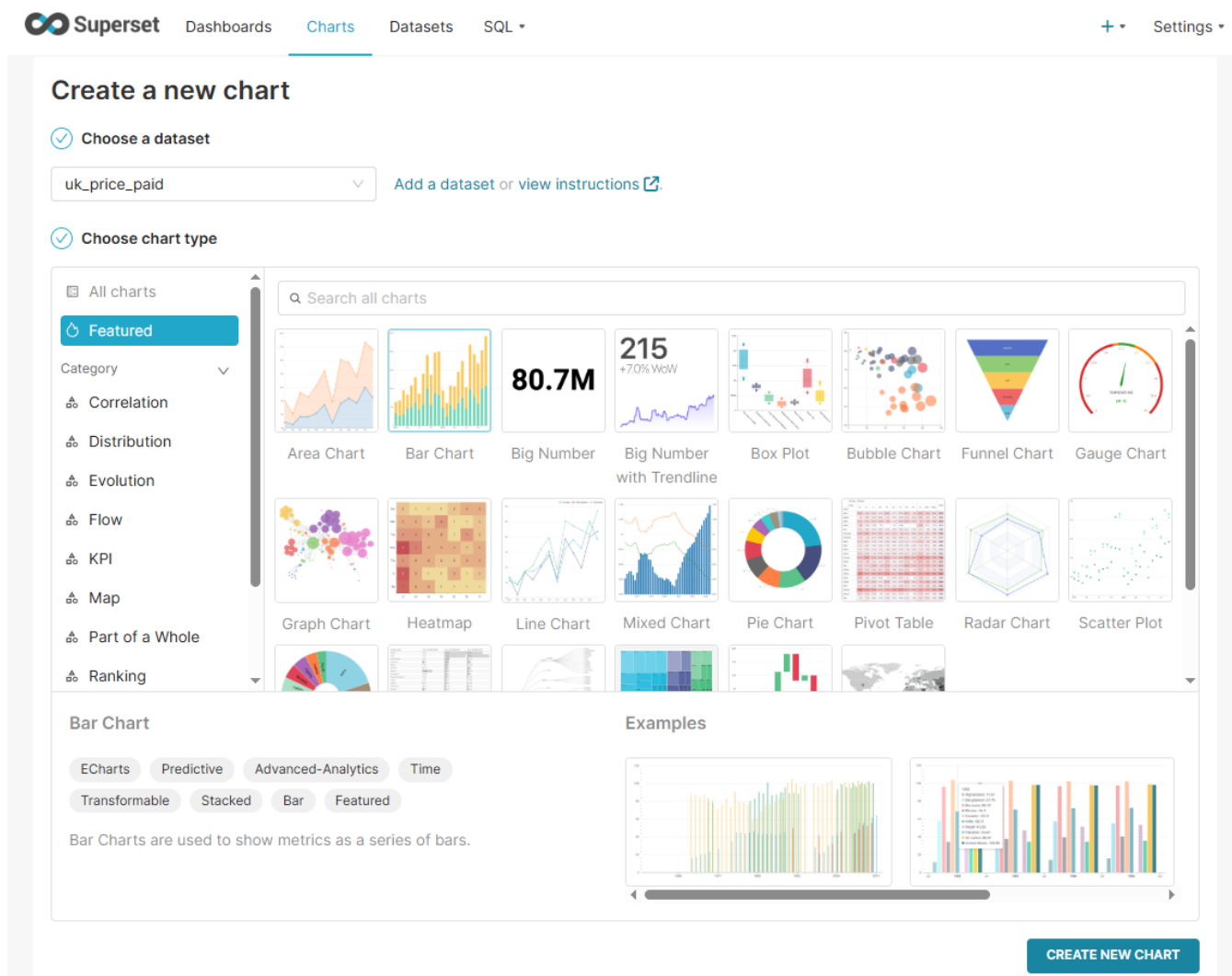
Column Name ▾	Datatype ▾
price	UInt32
date	Date
postcode1	LowCardinali...
postcode2	LowCardinali...
type	Enum8

Далее создаем дашборд для графиков:



Жмем CREATE NEW CHART для создания нового графика





Выбираем график и созданный ранее датасет.

Добавление данных в clickhouse.

Я использовал стандартный набор данных – стоимость жилья в Англии:

<https://clickhouse.com/docs/getting-started/example-datasets/uk-price-paid>

Загрузил ежемесячные изменения:

<https://www.gov.uk/government/statistical-data-sets/price-paid-data-downloads>

Создание схемы и таблицы:

```
CREATE DATABASE uk;
```

```
CREATE TABLE uk.uk_price_paid
```

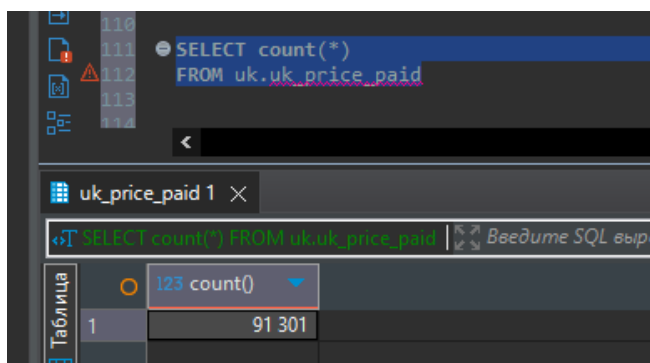
```
(
  price UInt32,
  date Date,
  postcode1 LowCardinality(String),
  postcode2 LowCardinality(String),
  type Enum8('terraced' = 1, 'semi-detached' = 2, 'detached' = 3, 'flat' = 4, 'other' = 0),
  is_new UInt8,
  duration Enum8('freehold' = 1, 'leasehold' = 2, 'unknown' = 0),
  addr1 String,
  addr2 String,
  street LowCardinality(String),
```

```

locality LowCardinality(String),
town LowCardinality(String),
district LowCardinality(String),
county LowCardinality(String)
)
ENGINE = MergeTree
ORDER BY (postcode1, postcode2, addr1, addr2);

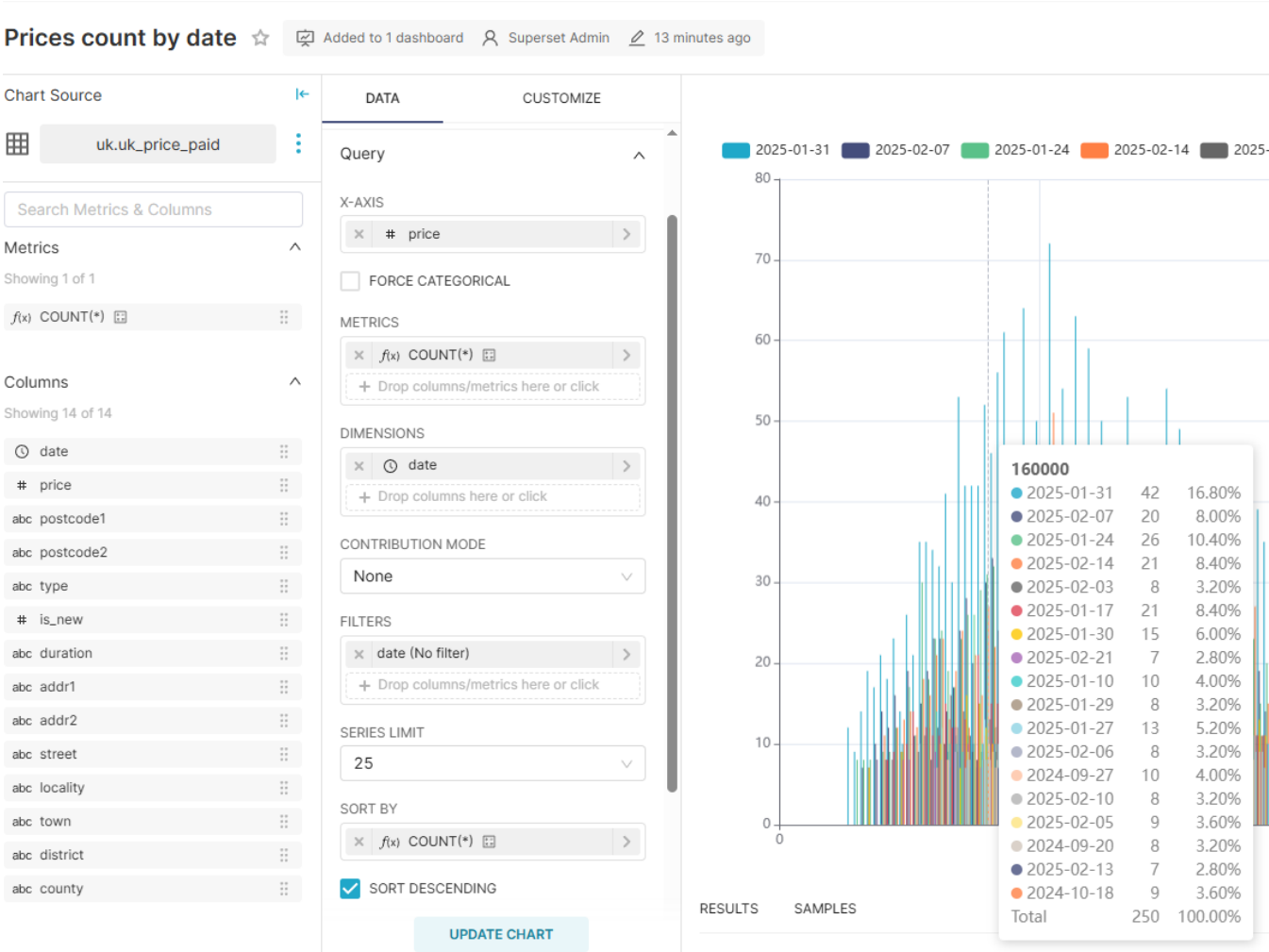
INSERT INTO uk.uk_price_paid
SELECT
    toUInt32(price_string) AS price,
    parseDateTimeBestEffortUS(time) AS date,
    splitByChar(' ', postcode)[1] AS postcode1,
    splitByChar(' ', postcode)[2] AS postcode2,
    transform(a, ['T', 'S', 'D', 'F', 'O'], ['terraced', 'semi-detached', 'detached', 'flat', 'other']) AS type,
    b = 'Y' AS is_new,
    transform(c, ['F', 'L', 'U'], ['freehold', 'leasehold', 'unknown']) AS duration,
    addr1,
    addr2,
    street,
    locality,
    town,
    district,
    county
FROM url(
    'http://prod.publicdata.landregistry.gov.uk.s3-website-eu-west-1.amazonaws.com/pp-monthly-update-new-version.csv',
    'CSV',
    'uuid_string String,
    price_string String,
    time String,
    postcode String,
    a String,
    b String,
    c String,
    addr1 String,
    addr2 String,
    street String,
    locality String,
    town String,
    district String,
    county String,
    d String,
    e String'
) SETTINGS max_http_get_redirects=10;

```



Создание графиков и их проверка запросом.

1. Bar – выборка кол-ва цен жилья относительно даты.



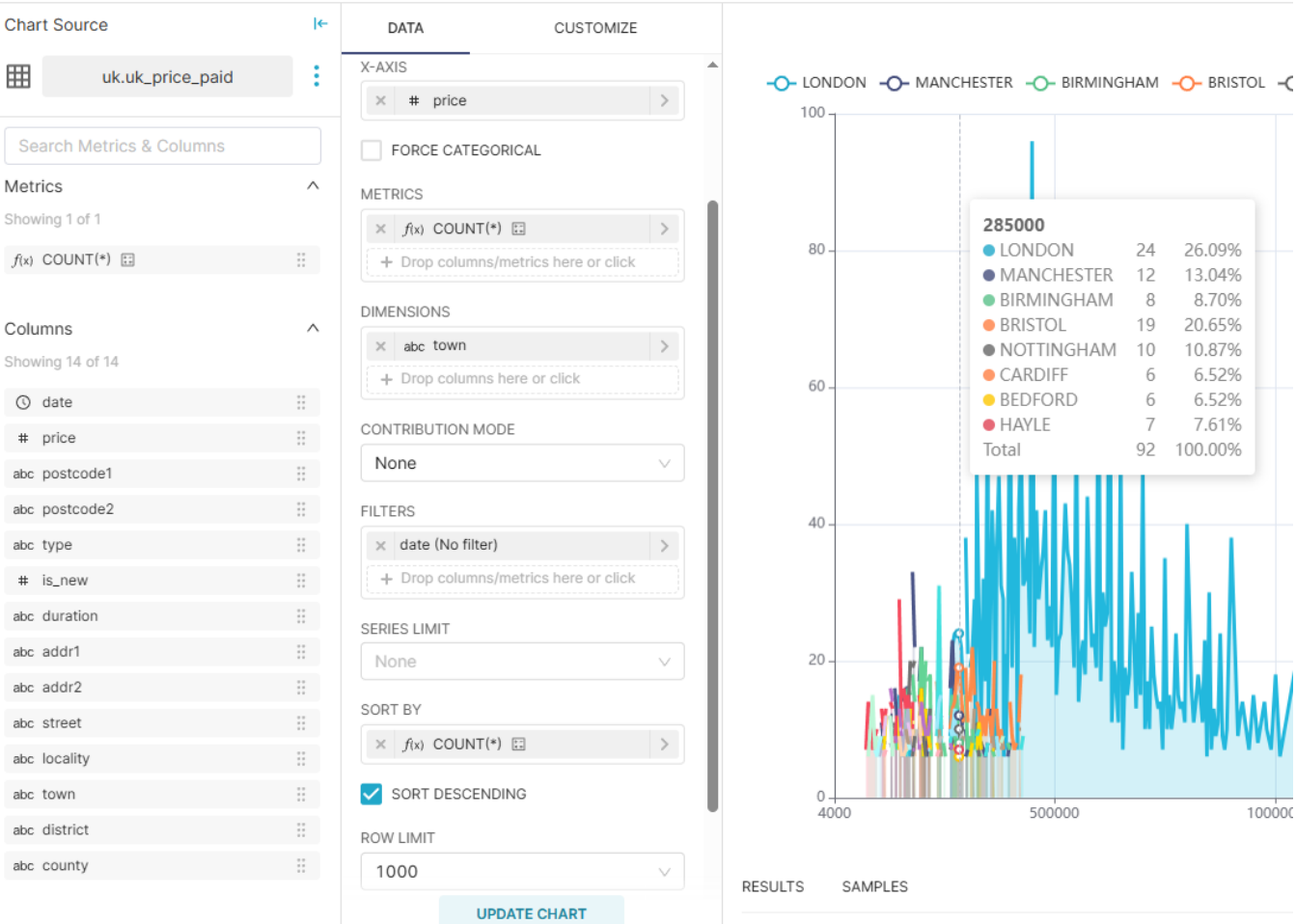
```
147
148
149
150
151
152
153
154
155
select count(price) cnt, date, price
from uk.uk_price_paid
where price = 160000
GROUP BY date, price
ORDER BY cnt desc
```

	123 cnt	date	123 price
1	42	2025-01-31	160 000
2	26	2025-01-24	160 000
3	21	2025-02-14	160 000
4	21	2025-01-17	160 000
5	20	2025-02-07	160 000
6	15	2025-01-30	160 000
7	13	2025-01-27	160 000
8	10	2025-01-10	160 000
9	10	2024-09-27	160 000
10	9	2024-10-18	160 000
11	9	2025-02-05	160 000
12	8	2025-01-29	160 000
13	8	2024-09-20	160 000
14	8	2025-02-10	160 000
15	8	2025-02-06	160 000

2. Line – выборка кол-ва цен жилья относительно города.

Prices count by town

Added to 1 dashboard Superset Admin 30 minutes ago



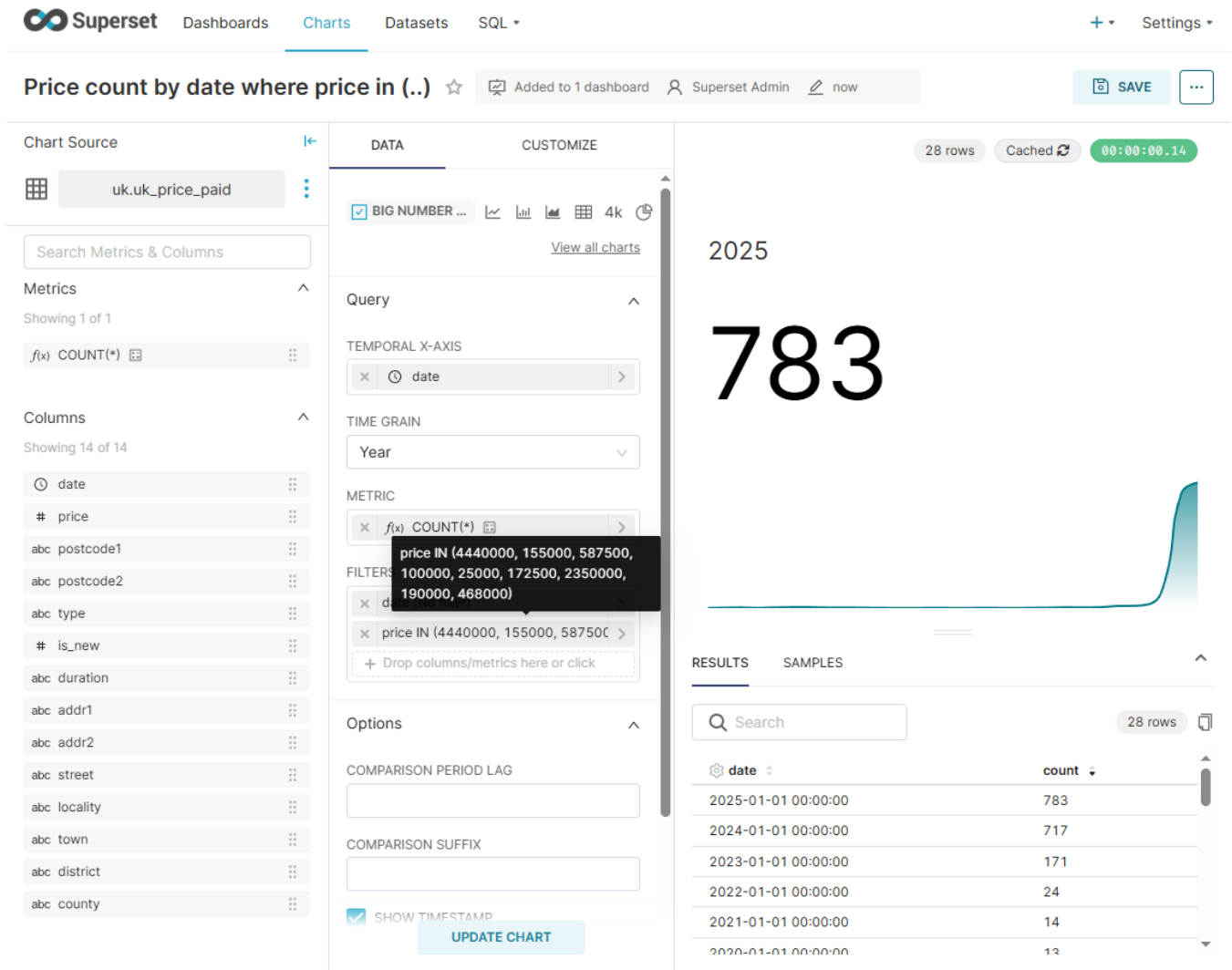
```
148
149 select count(price) cnt, town, price
150 from uk.uk_price_paid
151 where price = 285000
152 GROUP BY town, price
153 ORDER BY cnt desc
154
155
```

uk_price_paid 1

select count(price) cnt, town, price from uk.uk_price_paid where price = 285000

	cnt	town	price
1	24	LONDON	285 000
2	19	BRISTOL	285 000
3	12	MANCHESTER	285 000
4	10	NOTTINGHAM	285 000
5	8	BIRMINGHAM	285 000
6	7	HAYLE	285 000
7	6	PETERBOROUGH	285 000
8	6	SHREWSBURY	285 000
9	6	BEDFORD	285 000
10	6	LIVERPOOL	285 000
11	6	LUTON	285 000
12	6	CARDIFF	285 000
13	5	YORK	285 000
14	5	BASILDON	285 000
15	5	MILTON KEYNES	285 000

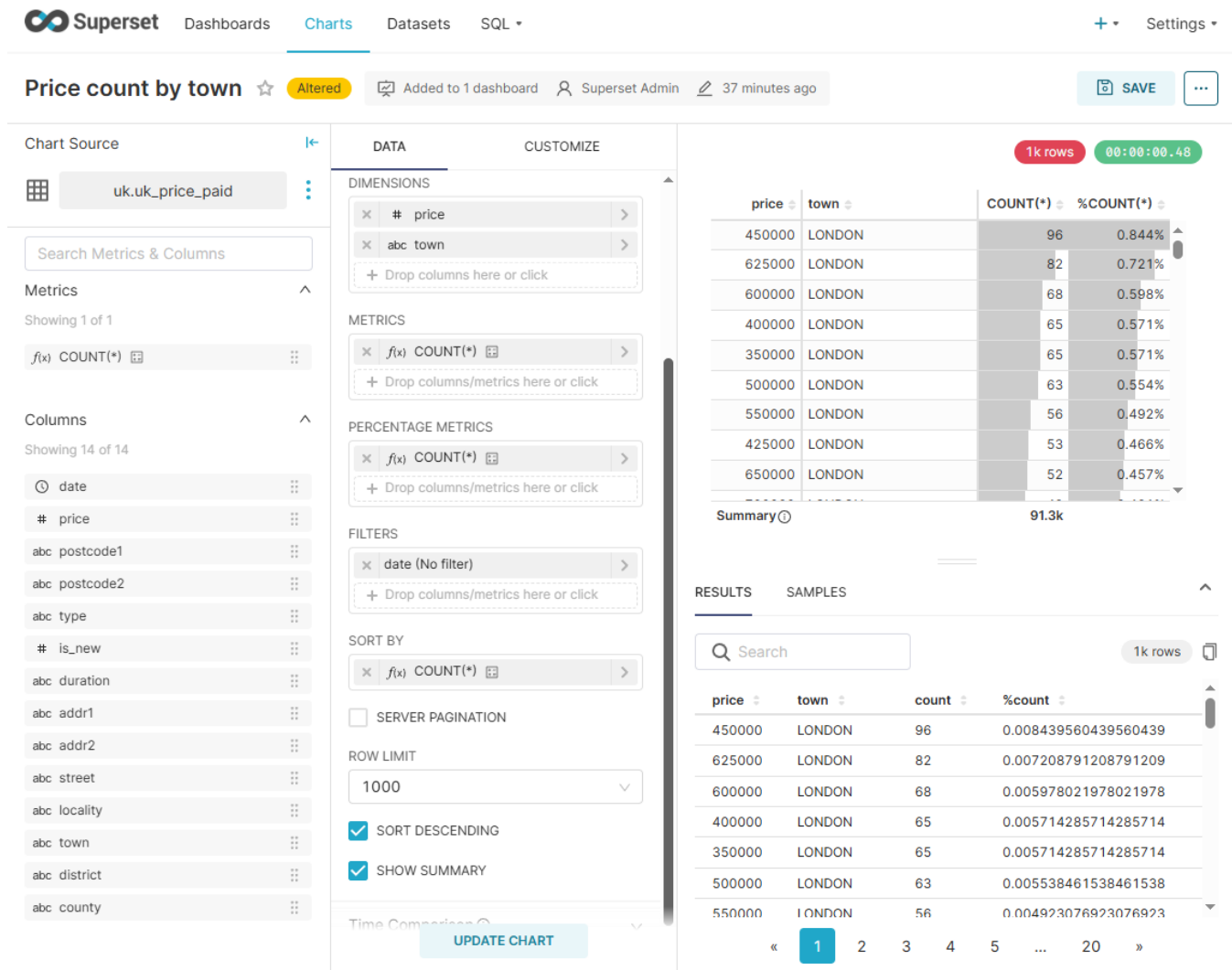
3. Bug number with trend line – выборка кол-ва жилья по годам, цена которого находится в массиве значений.



```
select count(price) cnt, toYear(date) year
from uk.uk_price_paid
where price in (4440000, 155000, 587500, 100000, 25000, 172500, 2350000, 190000, 468000)
GROUP BY year
ORDER BY cnt desc
```

cnt	year
783	2025
717	2024
171	2023
24	2022
14	2021
13	2020
6	2001
5	2019
5	2017
4	1997
4	1999
4	2002
4	2018
3	2016
3	2003
3	2004

4. Table – выборка кол-ва цен жилья относительно города.



```

175
176
177 select count(price) cnt,
178 price,
179 town,
180 (select count(price) from uk.uk_price_paid) cntTotal
181 from uk.uk_price_paid
182 GROUP BY price, town
183 ORDER BY cnt desc
184
185

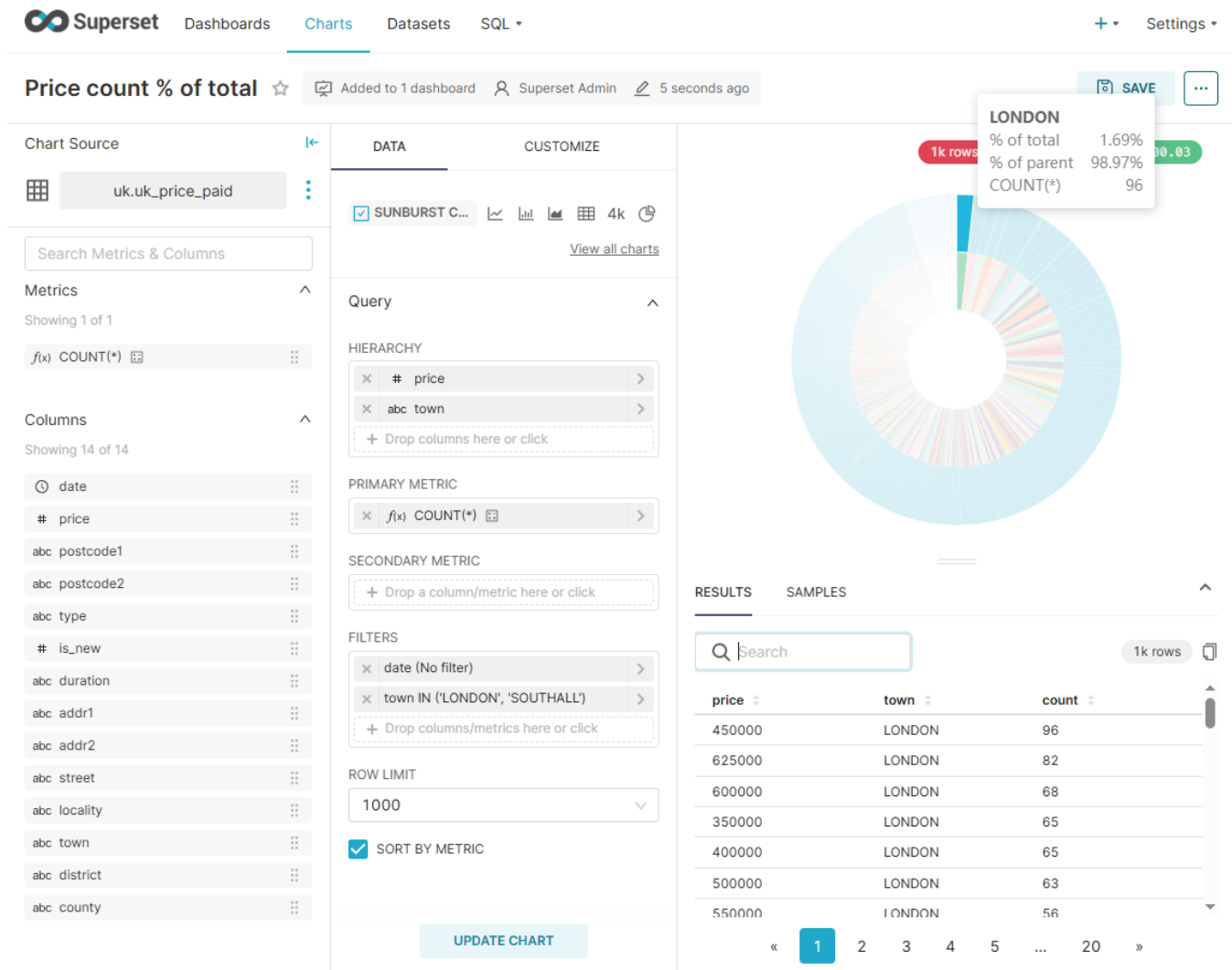
```

uk_price_paid 1

select count(price) cnt, price, town, (select count(price) from uk.uk_price_paid) cntTotal

	cnt	price	town	cntTotal
1	96	450 000	LONDON	91 301
2	82	625 000	LONDON	91 301
3	68	600 000	LONDON	91 301
4	65	400 000	LONDON	91 301
5	65	350 000	LONDON	91 301
6	63	500 000	LONDON	91 301
7	56	550 000	LONDON	91 301
8	53	425 000	LONDON	91 301
9	52	650 000	LONDON	91 301
10	49	700 000	LONDON	91 301
11	49	325 000	LONDON	91 301
12	47	375 000	LONDON	91 301
13	44	575 000	LONDON	91 301
14	43	525 000	LONDON	91 301
15	42	360 000	LONDON	91 301
16	42	480 000	LONDON	91 301

5. Sunburst chart – выборка кол-ва цен жилья относительно города.



```

176
177 select
178     count(*) cnt,
179     price,
180     town,
181 from uk.uk_price_paid
182 where town in ('LONDON', 'SOUTHALL')
183 GROUP BY price, town
184 ORDER BY cnt desc
185
186
187

```

unknown 1 X

select count(*) cnt, price, town from uk.uk_price_paid where town in ('LONDON', 'SOUTHALL')

	cnt	price	town
1	96	450 000	LONDON
2	82	625 000	LONDON
3	68	600 000	LONDON
4	65	350 000	LONDON
5	65	400 000	LONDON
6	63	500 000	LONDON
7	56	550 000	LONDON
8	53	425 000	LONDON
9	52	650 000	LONDON
10	49	325 000	LONDON
11	49	700 000	LONDON
12	47	375 000	LONDON
13	44	575 000	LONDON
14	43	525 000	LONDON
15	42	480 000	LONDON
16	42	360 000	LONDON