ROCKSET

# Understanding Rockset

Your one-stop guide

April 2020

## Understanding Rockset

We created this guide to help you understand how Rockset works and think about the data latency, query performance and scale required for your application.

### WHAT IS ROCKSET:

Rockset is the real-time database that automatically builds indexes on the latest data from user interactions and other sources. Move faster using serverless data APIs for millisecond-latency search, aggregations and joins. Rockset is ideal for building intelligent applications like personalization engines, gaming leaderboards, IoT apps and vision-AI based automation.

- Watch a Rockset demo video Watch Now>>
- Explore use cases for Rockset Connect with a solution architect>>

### START A FREE TRIAL:

Rockset is a serverless database so there is nothing to download, configure or install. Simply start by creating a 14 day trial account at rockset.com and get $300 in credits.

Ingest data from your own transactional database or event stream, run different types of queries, and use an API to query Rockset directly from your own application. The credits in your free trial will cover unlimited queries on 100 GB of data.

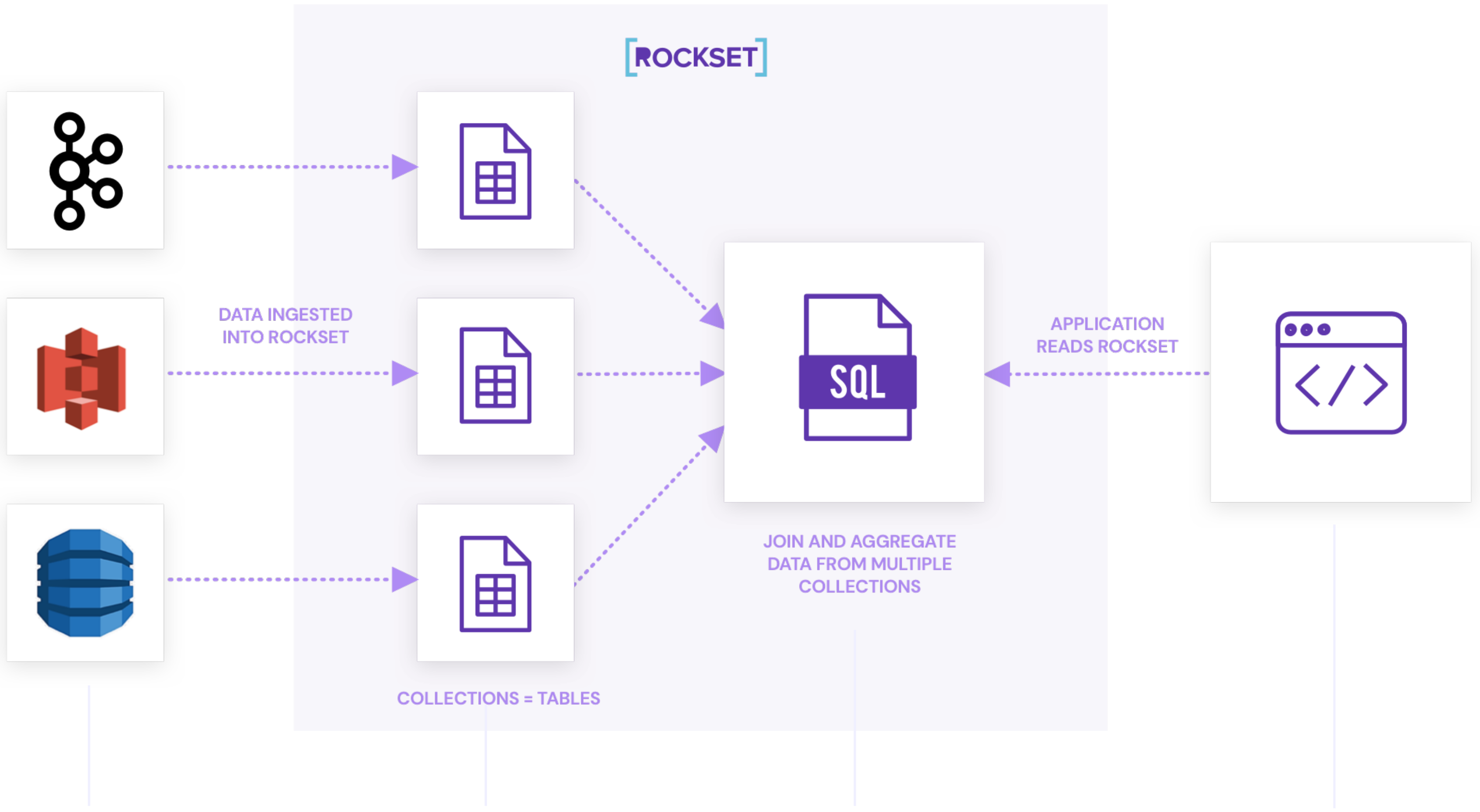
Here are sample projects and datasets to get started:

- We simulated an event stream of orders from an e-commerce site and ran SQL queries to determine the highest selling products Learn more>>
- We brought HackerNews data into DynamoDB to explore blockchain Learn more>>
- We used a Citi Bike in New York City dataset from S3 to explore the distribution of rides Learn more>>

It is helpful to understand the following concepts when starting a trial:

- Pricing is based on amount of data ingested and amount of data retained. Every GB of data in Rockset is allocated a certain amount of storage, compute and memory for high-performance queries. Learn more about pricing>>
- Query performance is based on the instance type, number of replicas and how the SQL query is constructed. Use the live chat in the console for help with SQL query construction and performance tuning.

# Understanding Rockset


ROCKSET
DATA INGESTED INTO ROCKSET
COLLECTIONS = TABLES
SQL
JOIN AND AGGREGATE DATA FROM MULTIPLE COLLECTIONS
APPLICATION READS ROCKSET

## 1. INTEGRATE DATA SOURCES

Set up a secure integration with your transactional database, event stream or data lake. An integration gives Rockset read-access to continuously ingest your data in real-time.

## 2. CREATE COLLECTIONS

You can think of collections as tables in a relational model. We recommend a 1:1 mapping between your database tables and collections or event stream topics and collections.

## 3. RUN SQL QUERIES

You can run millisecond-latency SQL queries across any of your collections including joins, filters, and aggregations, without doing any upfront schema definition.

## 4. BUILD DATA APPLICATIONS

Your application can query Rockset directly. Save parameterized SQL queries using Rockset's Query Lambdas which allow you to trigger query execution using dedicated REST endpoints.

## Step #1: Integrate Data Sources

### CREATE A DATA SOURCE INTEGRATION:

With Rockset’s native integrations with Apache Kafka, Amazon DynamoDB, Amazon S3, Amazon Kinesis, Google Cloud Storage and more, you do not need to write any code to get data into Rockset. Just select the data format and point Rockset at the corresponding topic, table, or bucket.

Data is continuously synced from the source without requiring upfront schema definition, removing the need for data cleaning or data transformations. Once ingested, Rockset automatically indexes every field, including nested fields, and generates a schema based on the exact fields and types present. Learn more about our approach to schemas on the Rockset blog.

You have unlimited access to data sources and APIs with Rockset. Here’s how data is ingested from different sources:

- Streaming platforms (Kafka, Kinesis): Rockset is a data sink for streaming events, it continuously receives data from Kafka or Kinesis. Rockset uses Kafka Connect for Kafka which is the best practice for connecting to any type of Kafka whether on-premises or in the cloud. View docs>>
- Transactional databases (DynamoDB, MongoDB): Rockset initially bulk loads data from transactional databases and then switches to tailing the database change stream to capture updates, inserts and deletes. A database change stream is an efficient way to build applications as it continuously captures changes to the transactional data within 1-2 seconds. Contrast that with a batch load which takes considerable time and effort. View docs>>
- Data lakes (S3, GCS): Rockset has native integrations with S3 and GCS and picks up new data as it lands in the data lake. That means you do not need to run constant jobs to make sure you are incorporating the latest data; Rockset does that for you. View docs>>
- Write API: Rockset has a generic Write API that can be used to continuously ingest data from other existing systems. You can upload an individual file or stream data through Rockset’s Write API or client libraries (Python, Java, CLI, Node.js, or Go). View docs>>
- Other data sources: Work with one of our solution architects for help ingesting data from other data sources. Connect with a solution architect >>

For loading data, Rocket automatically bulk loads data at a rate of 2 TB/hr and the default streaming ingest capacity is 5 MBps for trial accounts.

# Step #2 Create Collections

## CREATE A COLLECTION

Collections can be compared to tables in the relational world. All documents within a collection and all fields within a collection are fully mutable to keep in sync with any data source. We recommend a 1:1 mapping between your database tables and collections or event stream topics and collections. Join data from multiple collections at query time. Collections roll up to fully isolated workspaces- so you can have one workspace for each data application.

You can specify transformations using field mappings for type coercion, field masking, search tokenization, time series or geo indexing and also set time-based retention policies when creating a collection.

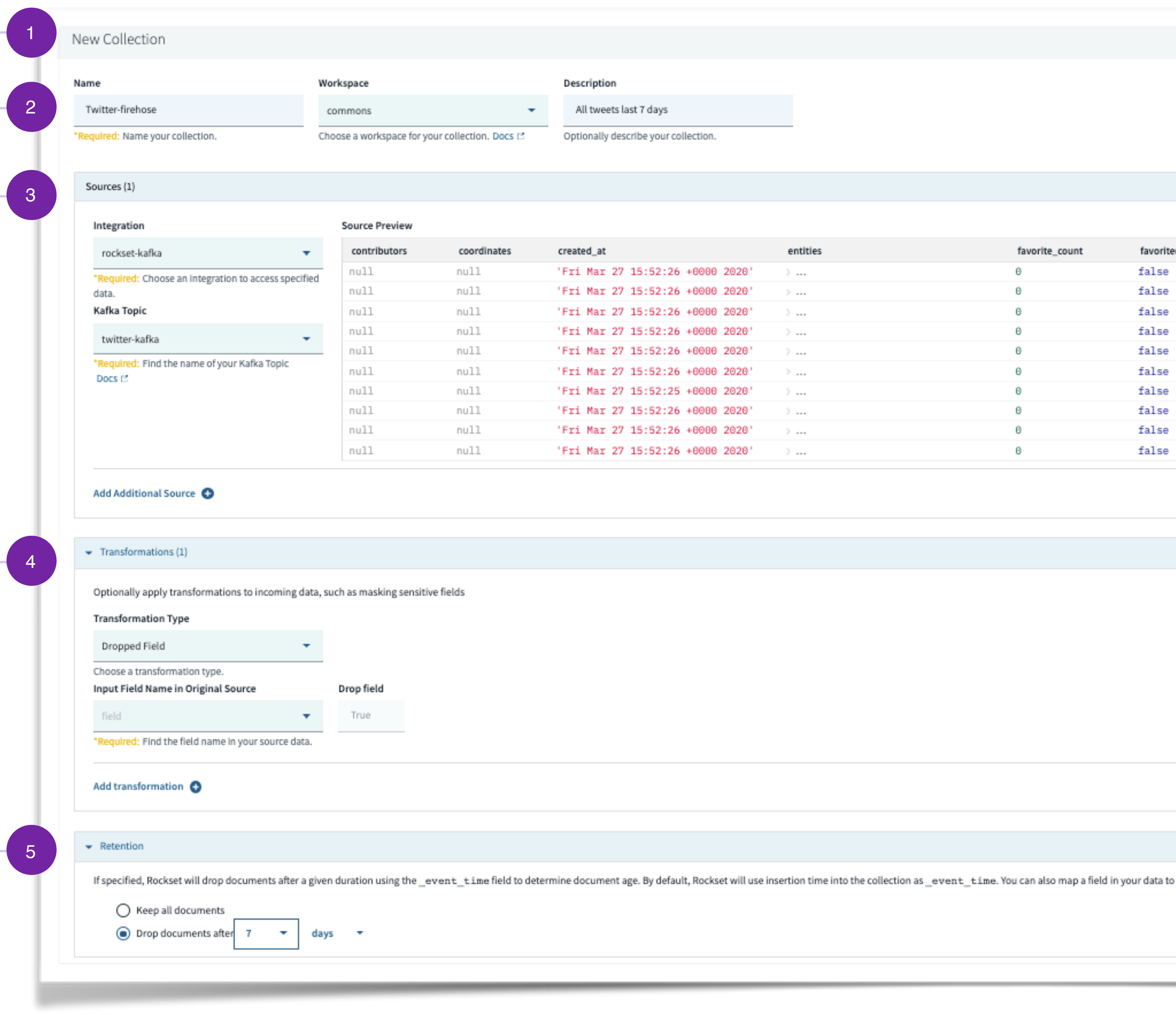
This is the interface where you can create a new collection or table in Rockset.

We collect metadata or the name you want to provide for the table, the description of the table, and the workspace. In Rockset, we can create a hierarchy of collections known as workspaces. Generally, you want one workspace per application.

You select the integration that is already set up and the corresponding topic for Kafka. As soon as you select the topic, Rockset will provide a preview of all your data in a SQL table. You can explore the messy data as it is coming into Rockset.

Transform the data before it enters Rockset to drop or mask personally identifiable information.

Select your retention period to automatically drop data after a period of time.



New Collection

| Name | Workspace | Description |
|---|---|---|
| Twitter-firehose | commons | All tweets last 7 days |
| *Required: Name your collection. | Choose a workspace for your collection. Docs | Optionally describe your collection. |

Sources (1)

Integration: rockset-kafka

*Required: Choose an integration to access specified data.

Kafka Topic: twitter-kafka

*Required: Find the name of your Kafka Topic Docs

Source Preview

| contributors | coordinates | created_at | entities | favorite_count | favorite |
|---|---|---|---|---|---|
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:25 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |
| null | null | 'Fri Mar 27 15:52:26 +0000 2020' | › ... | 0 | false |

Add Additional Source

Transformations (1)

Optionally apply transformations to incoming data, such as masking sensitive fields

Transformation Type: Dropped Field

Choose a transformation type.

Input Field Name in Original Source: field

Drop field: True

*Required: Find the field name in your source data.

Add transformation

Retention

If specified, Rockset will drop documents after a given duration using the `_event_time` field to determine document age. By default, Rockset will use insertion time into the collection as `_event_time`. You can also map a field in your data to

Keep all documents [✓ Drop documents after] 7 days

## Step #3 Run SQL Queries

### RUN QUERIES

With Rockset, you can run millisecond-latency SQL queries, including joins, filters and aggregations on semi-structured data.

Fast query performance is achieved by automatically indexing all fields, including nested fields.

We index all your data in three indexes:  an inverted index (used by search engines), a columnar index (used by many data warehouses for analytic queries), and a row-based index (used for updates and seeks). This is called Converged Indexing™ and you can learn more on the Rockset blog.

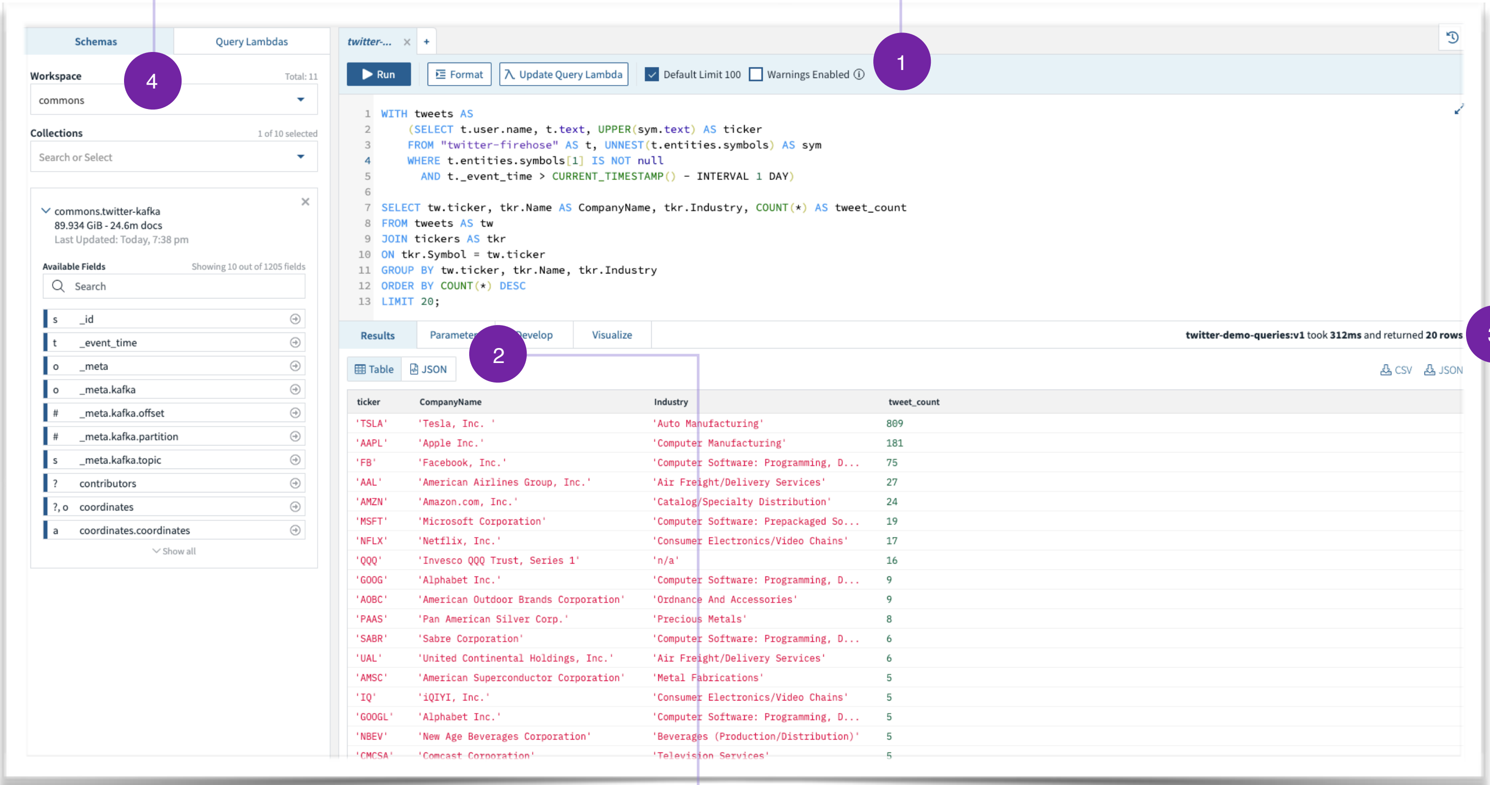
Rockset supports ANSI SQL, a common standard for databases, with added custom extensions for semi-structured data such as nested objects or arrays. One such extension, the UNNEST function, is used to expand arrays of values or documents to be queried. See the full guide to SQL in our documentation.

### BEST PRACTICES FOR RUNNING QUERIES

- Queries do not consume credits so go crazy running queries during your trial.
- Rockset supports different data types including geographic functions, date time functions, and windowing functions. Check out the docs for the full list of data types and SQL commands. Read now>>
- You should expect most queries to return in milliseconds. Use the live chat in the console for help troubleshooting and optimizing your queries.

You can explore the schema of your data and the available fields that are being brought in. You can inspect the field type and distribution.

The query editor interface of Rockset where you can construct your SQL queries.



Workspace Total: 11

commons

Collections 1 of 10 selected

Search or Select

commons.twitter-kafka
89.934 GiB - 24.6m docs
Last Updated: Today, 7:38 pm

Available Fields Showing 10 out of 1205 fields

| | |
|---|---|
| s | _id |
| t | _event_time |
| o | _meta |
| o | _meta.kafka |
| # | _meta.kafka.offset |
| # | _meta.kafka.partition |
| s | _meta.kafka.topic |
| ? | contributors |
| ?, o | coordinates |
| a | coordinates.coordinates |

✓ Default Limit 100 ☐ Warnings Enabled

```sql
WITH tweets AS
    (SELECT t.user.name, t.text, UPPER(sym.text) AS ticker
    FROM "twitter-firehose" AS t, UNNEST(t.entities.symbols) AS sym
    WHERE t.entities.symbols[1] IS NOT null
      AND t._event_time > CURRENT_TIMESTAMP() - INTERVAL 1 DAY)

SELECT tw.ticker, tkr.Name AS CompanyName, tkr.Industry, COUNT(*) AS tweet_count
FROM tweets AS tw
JOIN tickers AS tkr
ON tkr.Symbol = tw.ticker
GROUP BY tw.ticker, tkr.Name, tkr.Industry
ORDER BY COUNT(*) DESC
LIMIT 20;
```

**twitter-demo-queries:v1** took **312ms** and returned **20 rows**

| ticker | CompanyName | Industry | tweet_count |
|---|---|---|---|
| 'TSLA' | 'Tesla, Inc. ' | 'Auto Manufacturing' | 809 |
| 'AAPL' | 'Apple Inc.' | 'Computer Manufacturing' | 181 |
| 'FB' | 'Facebook, Inc.' | 'Computer Software: Programming, D... | 75 |
| 'AAL' | 'American Airlines Group, Inc.' | 'Air Freight/Delivery Services' | 27 |
| 'AMZN' | 'Amazon.com, Inc.' | 'Catalog/Specialty Distribution' | 24 |
| 'MSFT' | 'Microsoft Corporation' | 'Computer Software: Prepackaged So... | 19 |
| 'NFLX' | 'Netflix, Inc.' | 'Consumer Electronics/Video Chains' | 17 |
| 'QQQ' | 'Invesco QQQ Trust, Series 1' | 'n/a' | 16 |
| 'GOOG' | 'Alphabet Inc.' | 'Computer Software: Programming, D... | 9 |
| 'AOBC' | 'American Outdoor Brands Corporation' | 'Ordnance And Accessories' | 9 |
| 'PAAS' | 'Pan American Silver Corp.' | 'Precious Metals' | 8 |
| 'SABR' | 'Sabre Corporation' | 'Computer Software: Programming, D... | 6 |
| 'UAL' | 'United Continental Holdings, Inc.' | 'Air Freight/Delivery Services' | 6 |
| 'AMSC' | 'American Superconductor Corporation' | 'Metal Fabrications' | 5 |
| 'IQ' | 'iQIYI, Inc.' | 'Consumer Electronics/Video Chains' | 5 |
| 'GOOGL' | 'Alphabet Inc.' | 'Computer Software: Programming, D... | 5 |
| 'NBEV' | 'New Age Beverages Corporation' | 'Beverages (Production/Distribution)' | 5 |
| 'CMCSA' | 'Comcast Corporation' | 'Television Services' | 5 |

You can see the amount of time that it takes to run the query. This complex query took 541 milliseconds to return.

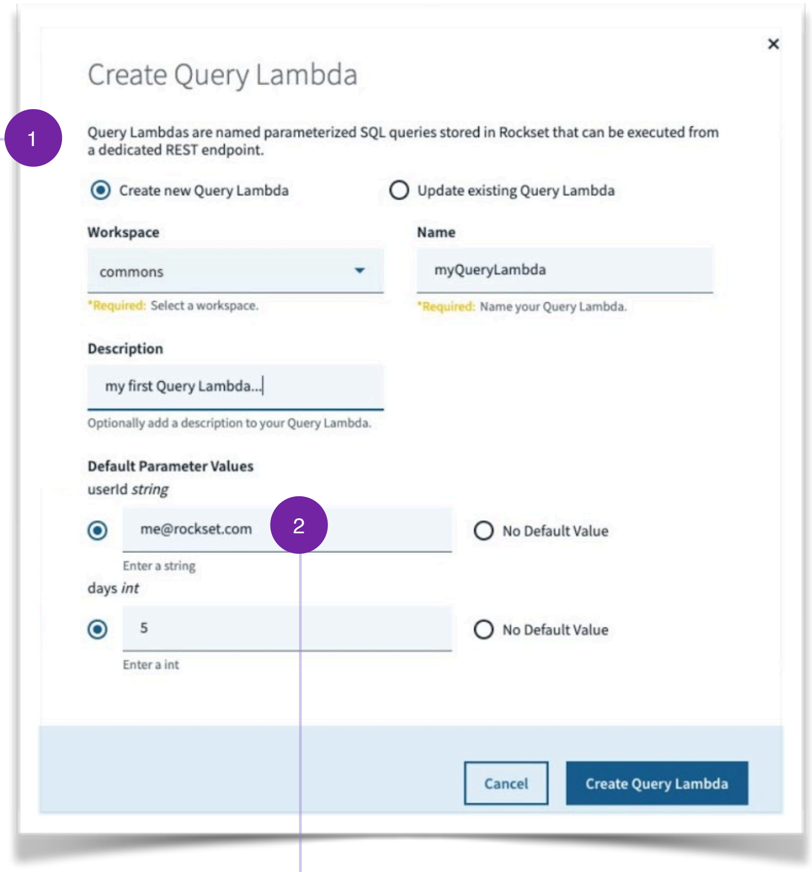
The results of the query are shown below and you can toggle between a table view and JSON view of the results.

## Step #4 Build Data Applications

### BUILD APPLICATION

You can query Rockset directly from an application or microservice using Query Lambdas. Query Lambdas are named parameterized SQL queries stored in Rockset that can be executed from a dedicated REST endpoint. Use Rockset's SDKs- Python, Node.js, Java, etc.- or the REST API to trigger the execution of the Query Lambda. Query Lambdas ensure that raw SQL no longer needs to live in application code and that query iteration and application iteration are separated. Learn more>>

Query Lambdas also work with dashboarding tools like Redash, Grafana and Tableau for operational monitoring and analytics.



Create Query Lambda

Query Lambdas are named parameterized SQL queries stored in Rockset that can be executed from a dedicated REST endpoint.

[✓ Create new Query Lambda] Update existing Query Lambda

Workspace: commons
*Required: Select a workspace.

Name: myQueryLambda
*Required: Name your Query Lambda.

Description: my first Query Lambda...
Optionally add a description to your Query Lambda.

Default Parameter Values

userId *string*
[✓ me@rockset.com] No Default Value
Enter a string

days *int*
[✓ 5] No Default Value
Enter a int

The interface for creating Query Lambdas, named parameterized SQL queries, that can be executed from a dedicated REST endpoint.

You can define default parameter values or you can make them mandatory for each execution.

```
# CURL Command
~ juliemills$ curl --request POST \
> --url https:'YourLambdaURL'
> -H 'Authorization: ApiKey [Your API Key]'\
> -H 'Content-Type: application/json' \
> | python -m json.tool

# Sample Response
{
 "collections": [
 "commons.tickers",
 "commons.twitter-firehose"
 ],
 "column_fields": [
 {
 "name": "ticker",
 "type": ""
 },
 {
 "name": "CompanyName",
 "type": ""
 },
 {
 "name": "Industry",
 "type": ""
 },
 {
 "name": "tweet_count",
 "type": ""
 }
 ],
```

```
"query_id": "[your query ID]",
"results": [
{
"CompanyName": "Tesla, Inc. ",
"Industry": "Auto Manufacturing",
"ticker": "TSLA",
"tweet_count": 742
},
{
"CompanyName": "Apple Inc.",
"Industry": "Computer Manufacturing",
"ticker": "AAPL",
"tweet_count": 166
}
```

1 A Query Lambda executed from a curl REST endpoint.

ROCKSET

# About Rockset

Rockset is the real-time database that automatically builds indexes on the latest data from user interactions and other sources. Move faster using serverless data APIs for millisecond-latency search, aggregations and joins. Rockset is ideal for building intelligent applications like personalization engines, gaming leaderboards, IoT apps and vision-AI based automation.

Find out more at rockset.com
Connect with us at support@rockset.com