**VERSION 1:**

Step 1: We’ll be using a source connector for reading Reddit posts and comments into Apache Kafka, via the Kafka connect framework

Step 2: We’ll us kafka Connect Reddit Source Connector to pipe new Reddit comments into our Kafka Cluster

Step 3: Next we’ll integrate our Kafka data source to Rocket which is a real time database to load the data from Kafka

Step 4: We have to download the Rockset JDBC driver from Maven.

Step 5: Create an API key in Rockset that Tableau will use for authenticating requests

Step 6: In Tableau, we connect to Rockset by choosing “Other Databases (JDBC)” and filling the fields, with our API key as the password:

Step 7: we’ll write the queries we need first in Rockset, and then use them to show our live Tableau dashboards using the ‘Custom SQL’ feature.

Step 8: After connecting Tableau to your data, there will be a option available

Called as New Custom SQL option on the Data Source page, Double click on it and create your custom sql queries and show the appropriate graphs.

**Version 2:**

**Implementing a live dashboard on event data using Tableau.**

|  |
| --- |
| Now we have our Data coming in, we’ll be using a Tableau Desktop to visualize our data. |

**Block Diagram:**

|  |
| --- |
| 1. Creating a Rockset account and creating a collection named wiki-events.  2. Then creating a Api key which we’ll be using in our python script to store the data coming in. |

|  |
| --- |
| Writing a Python script that subscribes to events from the Wikimedia stream and writes them to the wiki-events collections in Rockset. |

**STEP 1: STEP 2: STEP 3:**

**Data Source:**

1. **Ingesting Data from Wikimedia Event Stream**: I will first create a collection in Rockset, to which we will write a Python Script fetching the events originating from the [Wikimedia stream](https://wikitech.wikimedia.org/wiki/Event_Platform/EventStreams).
2. We are using Rockset's **Write API** to ingest the Wikimedia event stream.

**Python Script:**

import json

from sseclient import SSEClient as EventSource

from rockset import Client

rs=Client(api\_key='o9FvffQfu8yObGJlxAYzrY1Q0taQQmkB9Ahf71YXGjgXjJ8JsnDhccJfgM7zPaHX')

events = rs.Collection.retrieve("wiki-events")

streams = 'recentchange,page-links-change,page-create,page-move,page-properties-change,page-delete,test,recentchange,revision-create,page-undelete'

url = 'https://stream.wikimedia.org/v2/stream/{}'.format(streams)

for event in EventSource(url):

    try:

        if event.event == 'message':

            change = json.loads(event.data)

            events.add\_docs([change])

    except:

        continue

**Number of records and columns:**

|  |
| --- |
| **| Field Name |** |
| **|---------------------|** |
| **| $Schema |** |
| **| \_Event Time |** |
| **| \_Id |** |
| **| Added Links |** |
| **| Added Properties |** |
| **| Bot |** |
| **| Comment |** |
| **| Database |** |
| **| Id |** |
| **| Length |** |
| **| Log Action |** |
| **| Log Action Comment |** |
| **| Log Id |** |
| **| Log Params |** |
| **| Log Type |** |
| **| Meta |** |
| **| Minor |** |
| **| Namespace |** |
| **| Page Id |** |
| **| Page Is Redirect |** |
| **| Page Namespace |** |
| **| Page Title |** |
| **| Parsedcomment |** |
| **| Patrolled |** |
| **| Performer |** |
| **| Prior State |** |
| **| Removed Links |** |
| **| Removed Properties |** |
|  |
| **| Rev Content Changed |** |
| **| Rev Content Format |** |
| **| Rev Content Model |** |
| **| Rev Count |** |
| **| Rev Id |** |
| **| Rev Is Revert |** |
| **| Rev Len |** |
| **| Rev Minor Edit |** |
| **| Rev Parent Id |** |
| **| Rev Revert Details |** |
| **| Rev Sha1 |** |
| **| Rev Timestamp |** |
| **| Revision |** |
| **| Server Name |** |
| **| Server Script Path |** |
| **| Server Url |** |
| **| Test |** |
| **| Test Map |** |
| **| Timestamp |** |
| **| Title |** |
| **| Type (Wiki-Events) |** |
| **| User |** |
| **| Wiki |** |

We’ll be using custom SQL to deal with only limited columns from the above data.

**VISUALIZATION:**

1. We are using a **Tableau Dashboard** to visualize Real-Time Event Data.

Steps to connect to Tableau Dashboard:

* Place the Rockset Java SDK jar in the following folder depending on the operating system:

Windows: C:\Program Files\Tableau\Drivers

Mac: ~/Library/Tableau/Drivers

* In Tableau, navigate to Connect To a Server and select Other Databases (JDBC)
* Configure the connection as follows:

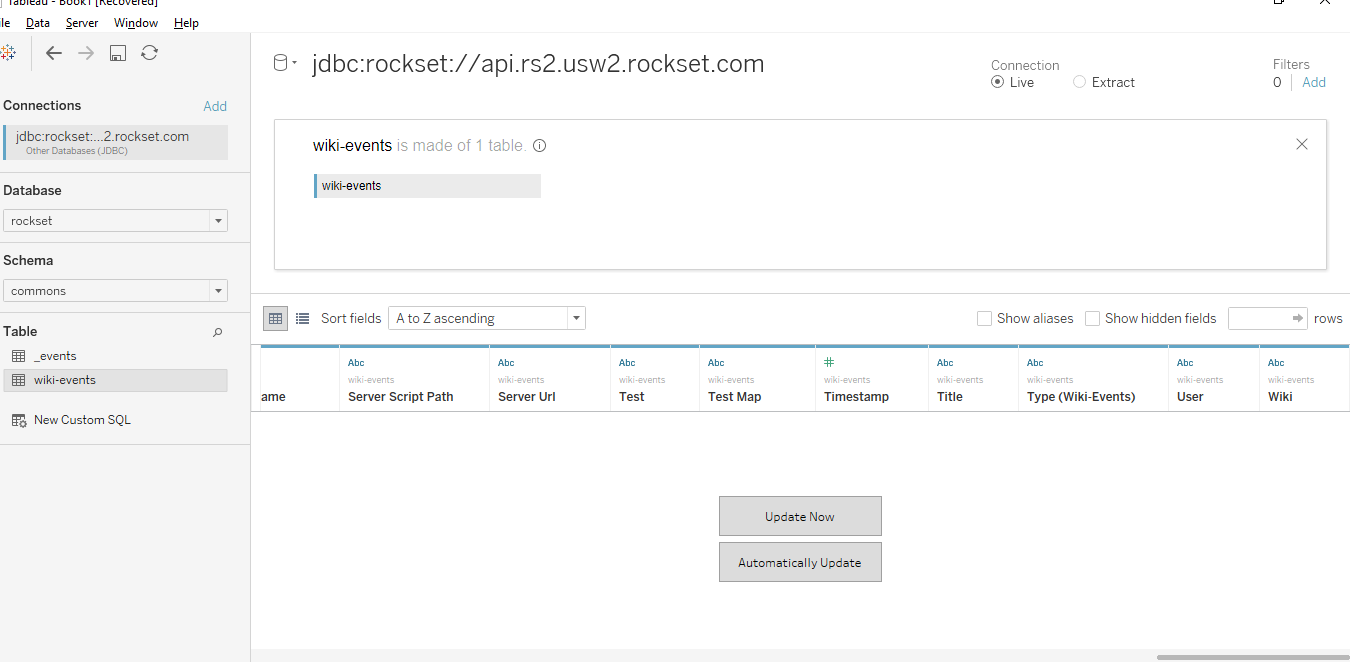
Use jdbc:rockset://api.rs2.usw2.rockset.com for URL

Select PostgreSQL as dialect.

Enter apikey as Username and Rockset API Key as the password.

Click on Sign In

* Below is the schema of all Rockset collections present in my workspace:



**GOAL:**

We can use a [custom SQL query](https://onlinehelp.tableau.com/current/pro/desktop/en-us/customsql.htm) within Tableau to specify the query for this, which gives us the resulting chart.

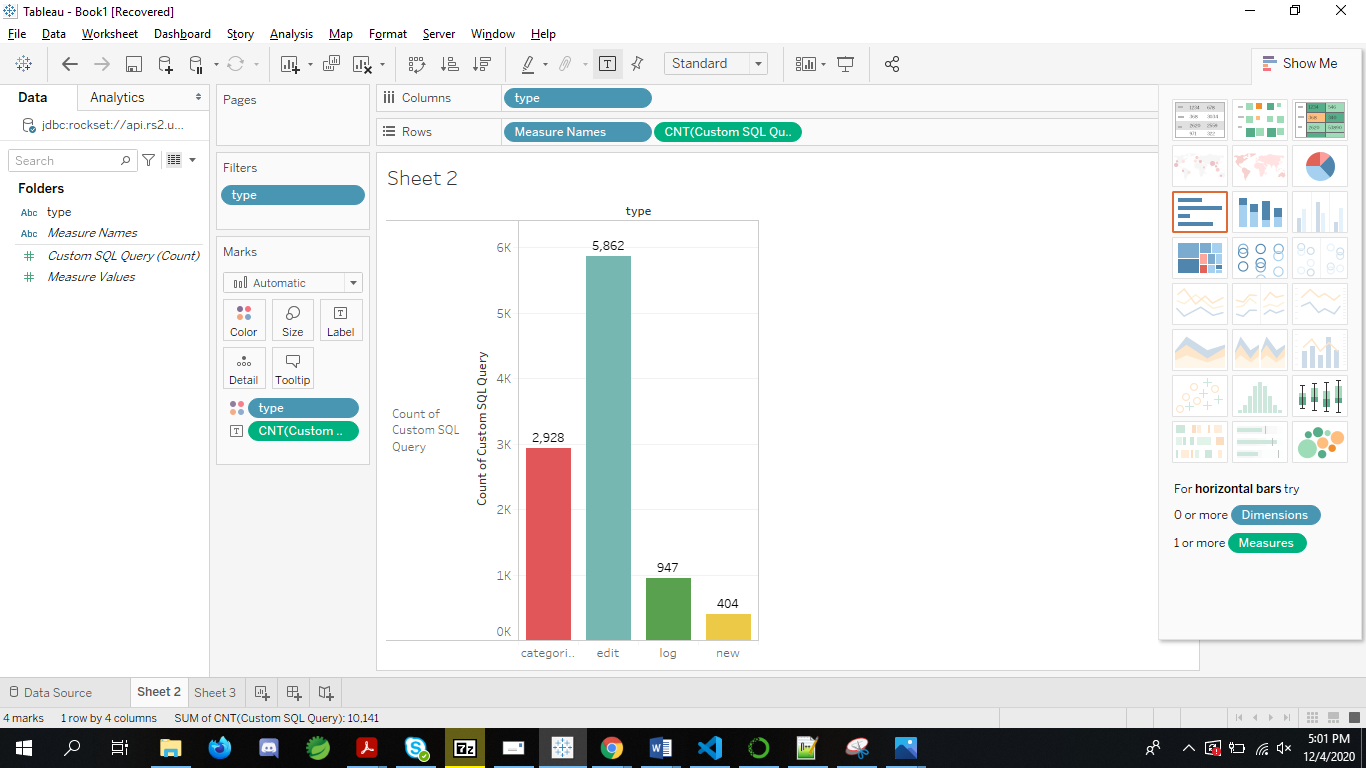
1. Wikimedia tracks several types of [change events](https://www.mediawiki.org/wiki/Manual:Recentchanges_table#rc_type): edit, new, log, and categorize. Our goal is to get an up-to-date count of the various types of changes made every minute.
2. Using the Type attribute we’ll be able to track the count of the type of modification made to a page.

### “Type” attribute: This field stores the type of modification that was made to a page:

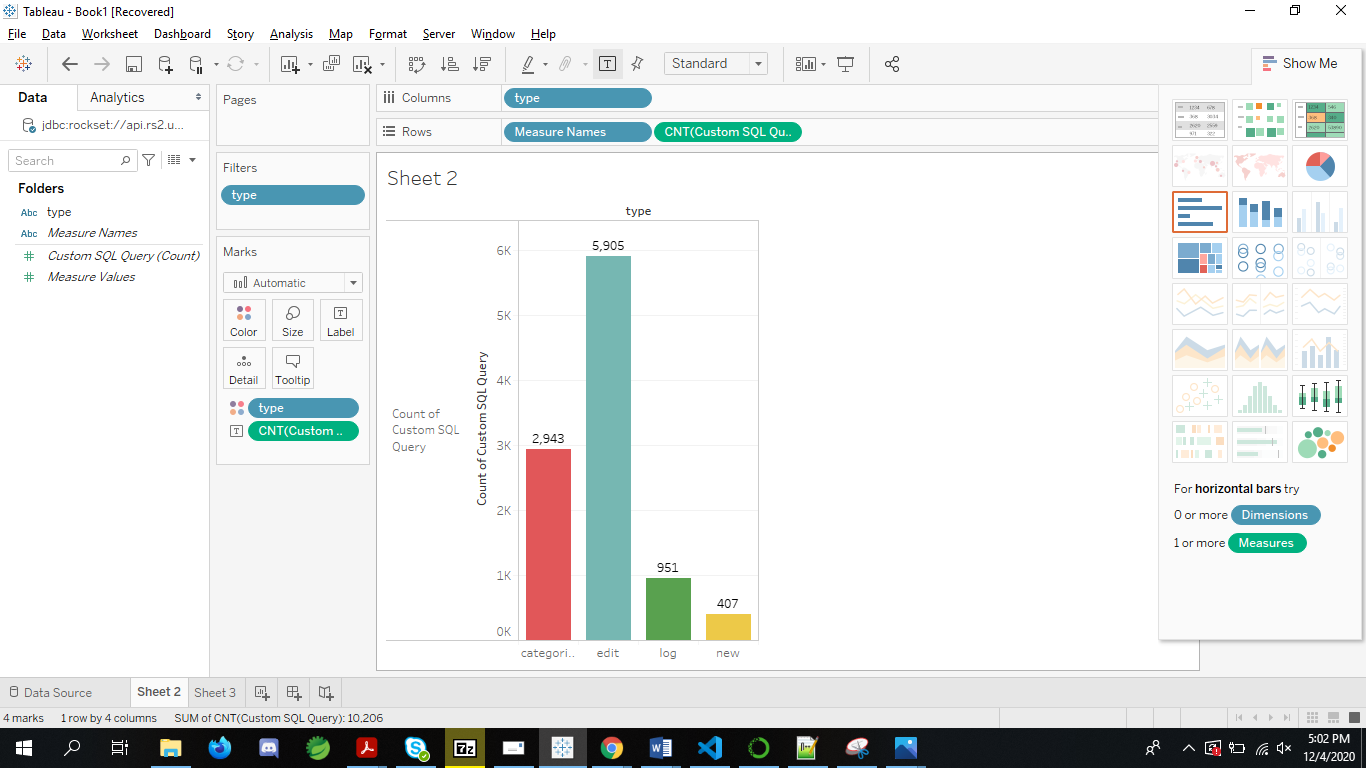
* (EDIT) – edit of existing page
* (NEW) – new page
* (LOG) – log action (added in MediaWiki 1.2)
* (CATEGORIZE) – category membership change, see [Manual:CategoryMembershipChanges](https://www.mediawiki.org/wiki/Special:MyLanguage/Manual:CategoryMembershipChanges).

**Bar Graph for the above goal:**

**In the below graph at 5:01 you can see the count of different categories of change events.**



**At 5.02 you can see the count of different categories.**



**STORY:**

* From the above two graphs, We can see that editing an existing page count is increasing quite a bit compared to the other categories.
* We can see that every minute new page creation count is the least among the four categories.
* Every minute after refreshing data we can clearly see that no of edits made across Wikimedia is increasing hugely.

**CONCLUSION:**

In a few steps, we ingested a stream of complex JSON event data, connected Tableau to the data in Rockset, and added some charts to our live dashboard. While it may normally take tens of minutes, if not longer, to process raw event data for use with a dashboarding tool, using Tableau on real-time data in Rockset allows users to perform live analysis on their data within seconds of the events occurring.