

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadinia

Introduction

This report documents the steps I followed to complete Assignment 6: Cloud Native Visual Analytics using ClickHouse and Grafana deployed on CSC Rahti OpenShift. The main objective of the assignment was to practice deploying cloud-native applications in a containerized environment and to build an end-to-end data analytics workflow. The work focused on deploying ClickHouse as a database service, importing and managing datasets using SQL, deploying Grafana for visualization, and creating interactive dashboards that demonstrate aggregation and analytical queries. Screenshots are included throughout the report to verify that each deployment and configuration step was completed successfully.

The assignment was divided into several phases. I began by preparing the Rahti OpenShift project and ensuring both the web console and oc CLI were correctly configured. Next, I deployed ClickHouse using the provided YAML configuration and verified that the pod, service, and route were working. After accessing the ClickHouse Web UI, I created tables and imported a movie dataset for analysis. I then deployed Grafana in the same OpenShift App to ensure internal networking, configured the ClickHouse data source, and created dashboards showing aggregations such as movies per year, average metascore by genre, and top directors. In the optional part, I added a second time-series dataset and implemented advanced analytics including monthly averages, yearly ranges, and percentile-based metrics.

During the assignment, I encountered several technical challenges. One key issue was ensuring that ClickHouse and Grafana were placed in the same OpenShift App so they could communicate over the internal service network. Another challenge involved correctly configuring database credentials and routes to avoid connection errors. Additionally, careful attention was required when writing SQL queries for aggregation and statistical metrics to ensure accurate results and efficient execution within Rahti's resource limits.

Phase A — Rahti project preparation (login and project selection)

In this phase, I prepared my Rahti OpenShift environment so I could deploy applications later without issues. First, I logged in to the CSC Rahti OpenShift web console using my university account. After logging in, I made sure I was in the Developer view instead of the Administrator view. Then I went to Home → Projects and selected my course project named my-assignment-6-cc. I confirmed that the project name appeared at the top of the page and that the Topology view opened correctly. This confirmed that I was working in the correct namespace. Next, I set up the OpenShift command line tool (oc CLI). In the web console, I opened Command Line Tools and downloaded the oc client for Windows (x86_64). After extracting the files, I opened Command Prompt in the folder where oc.exe

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadinia

was located and ran oc version to test it. The command worked successfully. Then I copied the login command from the OpenShift console by clicking Copy login command and selecting Display Token. I copied the full oc login command and pasted it into the terminal. The login was successful. After that, I ran oc whoami to confirm my username and oc project to confirm that I was using the correct project, my-assignment-6-cc. At the end of this phase, both the web console and the oc CLI were working correctly. The correct project was selected, and the environment was ready for deployment in the next phase.

The screenshot shows the My CSC web interface. The top navigation bar includes links for Online Course, Data Science, Systematic review a..., Courses, Podcast, Article Editor, Uusi kansio, Assignments, Mail - hamed.ahma..., Mail - Hamed.Ahma..., and All Bookmarks. The user is logged in as Hamed Ahmadinia. The main content area displays the details of Project 2017120, which is titled "Cloud Computing and Data Engineering". The project information includes:

- Project number: 2017120
- Title: Cloud Computing and Data Engineering
- Description: This is my project for course of Cloud Computing and Data Engineering (2025-2026) at Arcada
- Project type: Academic
- Project manager: Hamed Ahmadinia (hahmadin) ahmadin@arcada.fi
- Home organization: Arcada University of Applied Sciences
- Unix group: project_2017120
- Field of science: Other engineering and technologies

The Members section shows "No members found" and has a button to "+ Add members". The Services section lists four services: cPouta IaaS cloud, Puhti Supercomputer, Rahti Container Cloud, and Pukki Database as a Service, each with a "Login" button. The Funding Decisions section has a "+ Add funding decision" button.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

The screenshot shows the Rahti console's Overview dashboard. On the left, a sidebar lists navigation items such as Home, Favorites, Operators, Helm, Workloads, Networking, Storage, Builds, User Management, and Administration. The main content area is titled "Overview" and contains three panels: "Cluster Details" (Cluster API address: Not available, Cluster ID: Not available, Infrastructure provider: Not available, OpenShift version: Not available, Update channel: Not available), "Status" (Cluster status: Cluster, Control Plane, Insights, Dynamic Plugins), and "Activity" (Ongoing: No ongoing activities, Recent events: None, Error loading events: An error occurred during event retrieval, Attempting to reconnect...).

The screenshot shows the Rahti console's Topology view for the project "my-assignment-6-cc". The left sidebar lists workloads: Helm, Repositories, Releases, and a detailed section for Workloads including Topology, Pods, Deployments, DeploymentConfigs, StatefulSets, Secrets, ConfigMaps, CronJobs, Jobs, DaemonSets, ReplicaSets, ReplicationControllers, and HorizontalPodAutoscalers. The main content area displays a graph icon and the message "No resources found". It includes a note: "Start building your application or visit the [Add page](#) for more details."

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

The screenshot shows the Rahti interface with the 'Import YAML' page open. The left sidebar lists various Kubernetes resources: Helm, Repositories, Releases, Workloads, Topology, Pods, Deployments, DeploymentConfigs, StatefulSets, Secrets, ConfigMaps, CronJobs, Jobs, DaemonSets, ReplicaSets, ReplicationControllers, and HorizontalPodAutoscalers. The main area is titled 'Import YAML' with the sub-instruction 'Drag and drop YAML or JSON files into the editor, or manually enter files and use --- to separate each definition.' A code editor window is open, showing a single line of code: '1 |'. Below the code editor are 'Create' and 'Cancel' buttons.

The screenshot shows a GitHub repository page for 'rahti-openshift-yaml'. The repository name is 'rahti-openshift-yaml'. The 'clickhouse-rahti.yaml' file is displayed. The file content is as follows:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: clickhouse
spec:
  replicas: 1
  selector:
    matchLabels:
      app: clickhouse
  template:
    metadata:
      labels:
        app: Clickhouse
    spec:
      containers:
        - name: clickhouse
          image: clickhouse/clickhouse-server:latest
          ports:
            - containerPort: 8123 # HTTP interface
            - containerPort: 9000 # native client
          env:
            - name: CLICKHOUSE_USER
              value: clickhouse
            - name: CLICKHOUSE_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: clickhouse
                  key: Clickhouse-password
          volumes:
            - name: clickhouse-data
              mountPath: /var/lib/clickhouse
          volumes:
            - name: clickhouse-data
              emptyDir: {} # for demo, ephemeral storage
```

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

The screenshot shows the Rahti UI interface. On the left, there's a sidebar with categories like Helm, Workloads, and Jobs. The main area is titled "Import YAML" and contains a code editor with the following YAML configuration:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: clickhouse
spec:
  replicas: 1
  selector:
    matchLabels:
      app: clickhouse
  template:
    metadata:
      labels:
        app: clickhouse
    spec:
      containers:
        - name: clickhouse
          image: clickhouse/clickhouse-server:latest
```

Below the code editor are two buttons: "Create" and "Cancel".

The screenshot shows the Rahti UI interface after a deployment. The main area displays a success message: "Resources successfully created". Below this, a table lists the created resources:

Name	Namespace	Creation status
D clickhouse	NS my-assignment-6-cc	Created
S clickhouse	NS my-assignment-6-cc	Created
RT clickhouse	NS my-assignment-6-cc	Created

At the bottom of the main area, there's a link: "Import more YAML".

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

The screenshot shows the Rahti UI interface for creating a key/value secret. The left sidebar lists various Kubernetes resources under 'Workloads'. The main panel is titled 'Create key/value secret' for project 'my-assignment-6-cc'. It has fields for 'Secret name' (set to 'clickhouse'), 'Key' (set to 'clickhouse-password'), and 'Value' (containing 'MyStrongPass123'). A 'Create' button is at the bottom.

The screenshot shows the Rahti UI interface for the 'clickhouse' deployment. The left sidebar lists various Kubernetes resources under 'Workloads'. The main panel shows the 'Deployment details' for the 'clickhouse' deployment. It includes sections for 'Deployment details' (with a summary of 1 pod), 'Details' (Name: clickhouse, Namespace: my-assignment-6-cc, Labels: No labels), 'Metrics', 'YAML', 'ReplicaSets', 'Pods', 'Environment', and 'Events'. On the right, there is an 'Actions' dropdown menu with options like 'Edit Pod count', 'Add HorizontalPodAutoscaler', 'Add PodDisruptionBudget', etc.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

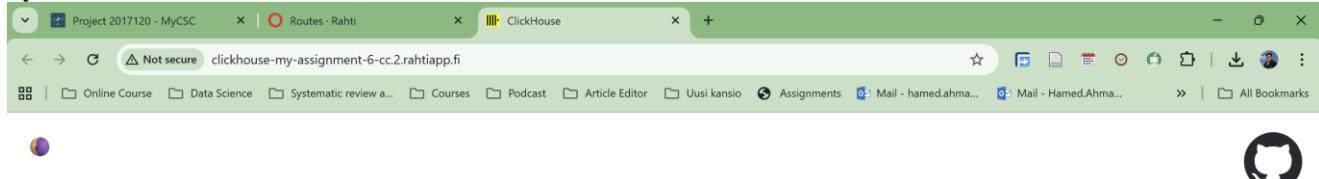
The screenshot shows the deployment details for a ClickHouse pod named 'clickhouse' in the 'my-assignment-6-cc' project. The deployment is currently at version 1, with 1 of 1 replicas ready. The pod has been running since Mon 18 Jul 2022, 04:45 UTC. The deployment configuration is set to 'On demand' and 'Always'. The deployment status is 'Not configured'. The deployment logs show no errors.

The screenshot shows the routes for the 'clickhouse' service in the 'my-assignment-6-cc' project. There is one route named 'clickhouse' with the URL <http://clickhouse-my-assignment-6-cc.rahtiapp.fi>. The route status is 'Accepted' and it is associated with the 'clickhouse' service.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadinia



ClickHouse

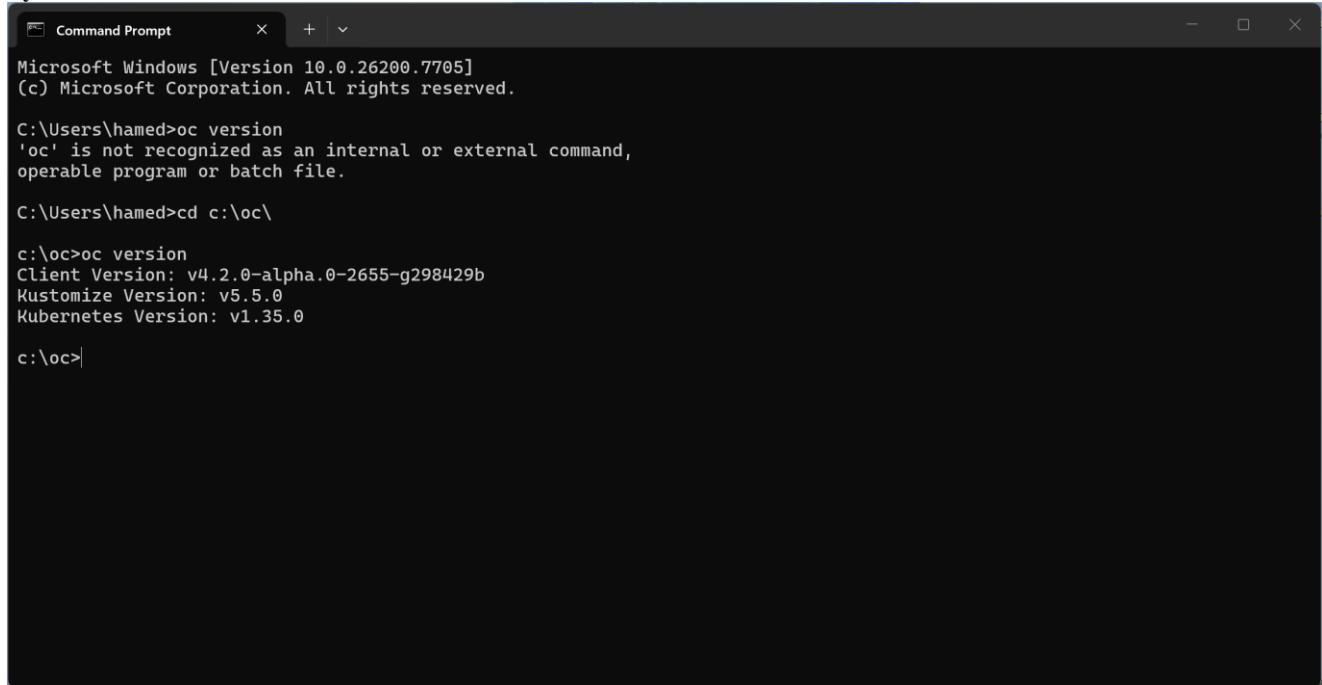
Web SQL UI Dashboard
Source code Documentation Website Cloud
Binary visualizer Merges visualizer

A screenshot of a web browser window. The address bar shows 'console.rahti.csc.fi/command-line-tools'. The page content is titled 'Command Line Tools'. It contains sections for 'oc - OpenShift Command Line Interface (CLI)' and 'helm - Helm 3 CLI'. The 'oc' section includes a 'Copy login command' button and a list of download links for various architectures. The 'helm' section includes a link to download Helm. A sidebar on the left lists various OpenShift resources like Home, Favorites, Operators, Helm, Workloads, Topology, Pods, Deployments, DeploymentConfigs, StatefulSets, Secrets, ConfigMaps, CronJobs, and Jobs. At the bottom, there is a green button with the URL 'https://downloads.openshift-console.apps.2.rahti.csc.fi/amd64/windows/oc.zip'.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania



```
Command Prompt      + | - X

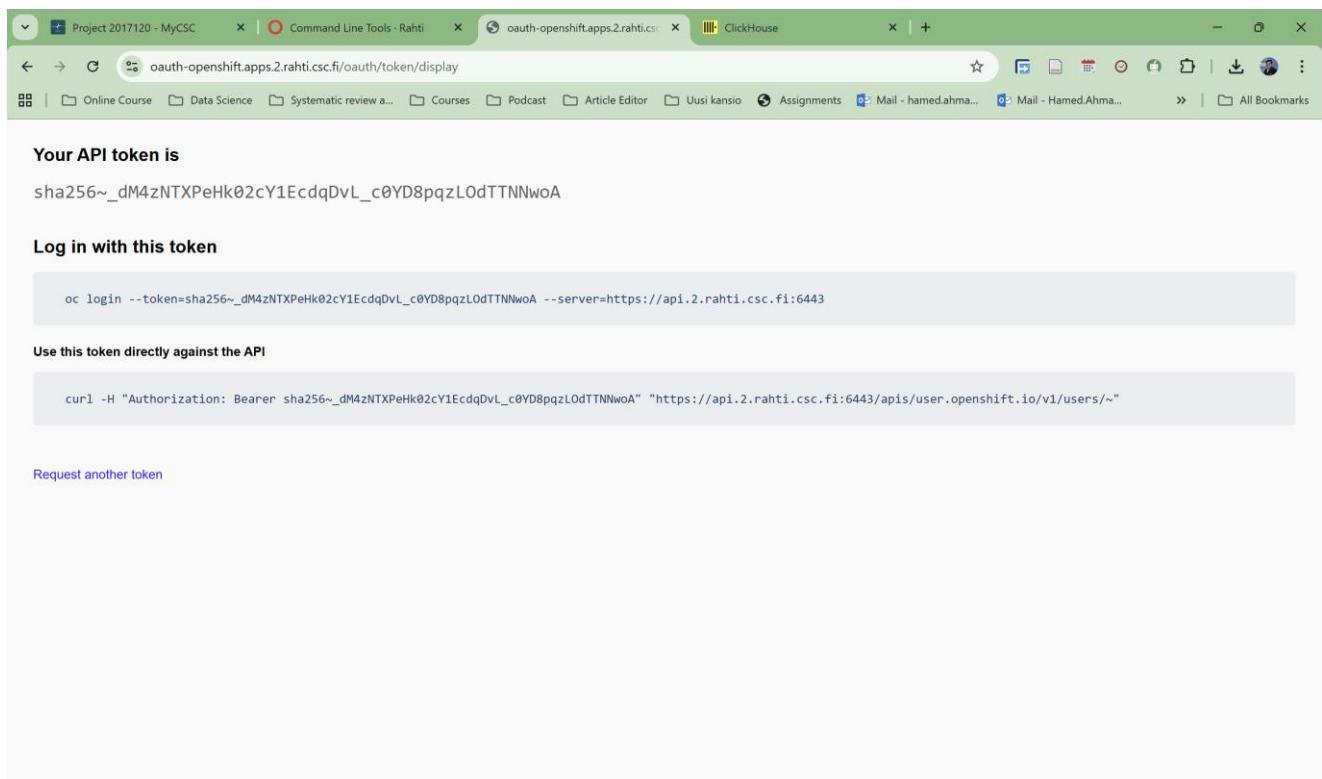
Microsoft Windows [Version 10.0.26200.7705]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hamed>oc version
'oc' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\hamed>cd c:\oc\

c:\oc>oc version
Client Version: v4.2.0-alpha.0-2655-g298429b
Kustomize Version: v5.5.0
Kubernetes Version: v1.35.0

c:\oc>
```



Your API token is
sha256~_dM4zNTXPeHk02cY1EcdqDvL_c0YD8pqzLoTTNNwoA

Log in with this token

```
oc login --token=sha256~_dM4zNTXPeHk02cY1EcdqDvL_c0YD8pqzLoTTNNwoA --server=https://api.2.rahti.csc.fi:6443
```

Use this token directly against the API

```
curl -H "Authorization: Bearer sha256~_dM4zNTXPeHk02cY1EcdqDvL_c0YD8pqzLoTTNNwoA" "https://api.2.rahti.csc.fi:6443/apis/user.openshift.io/v1/users/~"
```

[Request another token](#)

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadinia

```
Command Prompt Microsoft Windows [Version 10.0.26200.7705]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hamed>oc version
'oc' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\hamed>cd c:\oc\  
c:\oc>oc version
Client Version: v4.2.0-alpha.0-2655-g298429b
Kustomize Version: v5.5.0
Kubernetes Version: v1.35.0

c:\oc>oc login --token=sha256~_dM4zNTXPeHk02cY1EcdqDvL_c0YD8pqzL0dTTNNwoA --server=https://api.2.rahti.csc.fi:6443
Logged into "https://api.2.rahti.csc.fi:6443" as "hahmadin" using the token provided.

You have one project on this server: "my-assignment-6-cc"

Using project "my-assignment-6-cc".  
c:\oc>
```

```
Command Prompt Microsoft Windows [Version 10.0.26200.7705]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hamed>oc version
'oc' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\hamed>cd c:\oc\  
c:\oc>oc version
Client Version: v4.2.0-alpha.0-2655-g298429b
Kustomize Version: v5.5.0
Kubernetes Version: v1.35.0

c:\oc>oc login --token=sha256~_dM4zNTXPeHk02cY1EcdqDvL_c0YD8pqzL0dTTNNwoA --server=https://api.2.rahti.csc.fi:6443
Logged into "https://api.2.rahti.csc.fi:6443" as "hahmadin" using the token provided.

You have one project on this server: "my-assignment-6-cc"

Using project "my-assignment-6-cc".  
c:\oc>oc whoami
hahmadin  
c:\oc>oc project
Using project "my-assignment-6-cc" on server "https://api.2.rahti.csc.fi:6443".  
c:\oc>
```

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadinia

Phase B — Deploy ClickHouse and Load Data

In this phase, I deployed ClickHouse in my Rahti OpenShift project and verified that it was working correctly. First, I used the provided YAML configuration from the course GitHub repository to deploy ClickHouse in my project namespace (my-assignment-6-cc). After applying the configuration, I opened the Topology view in the OpenShift web console and confirmed that the ClickHouse pod was created and running without errors. Next, I checked that the ClickHouse route was working. I opened the generated route URL in my browser, which loaded the ClickHouse Web SQL interface. This confirmed that the service was publicly accessible. I then logged in to the ClickHouse Web UI using the username clickhouse and the password that I created earlier as a secret in OpenShift. After logging in successfully, I tested the connection by running a simple SQL query:

```
SELECT 1;
```

The query returned a result without errors, confirming that ClickHouse was functioning correctly.

After confirming the connection, I created a new database called **movies** using:

```
CREATE DATABASE movies;
```

I verified that the database was created by running:

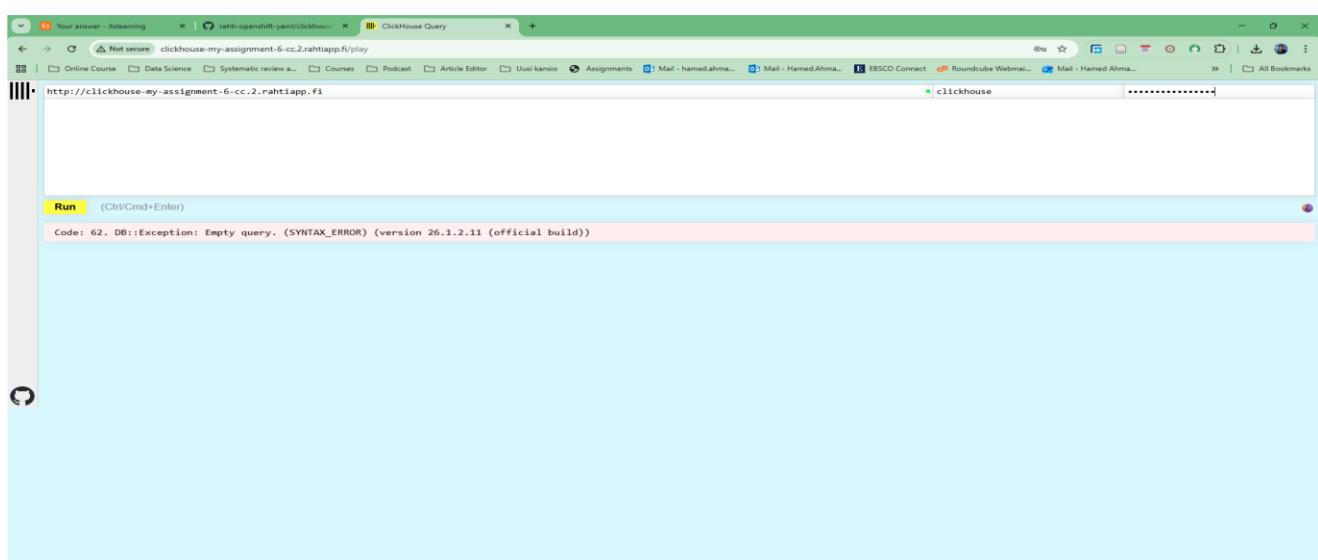
```
SHOW DATABASES;
```

The database “movies” appeared in the list.

Then, I created a table called movies_data inside the movies database. After creating the table, I verified its existence using:

```
SHOW TABLES FROM movies;
```

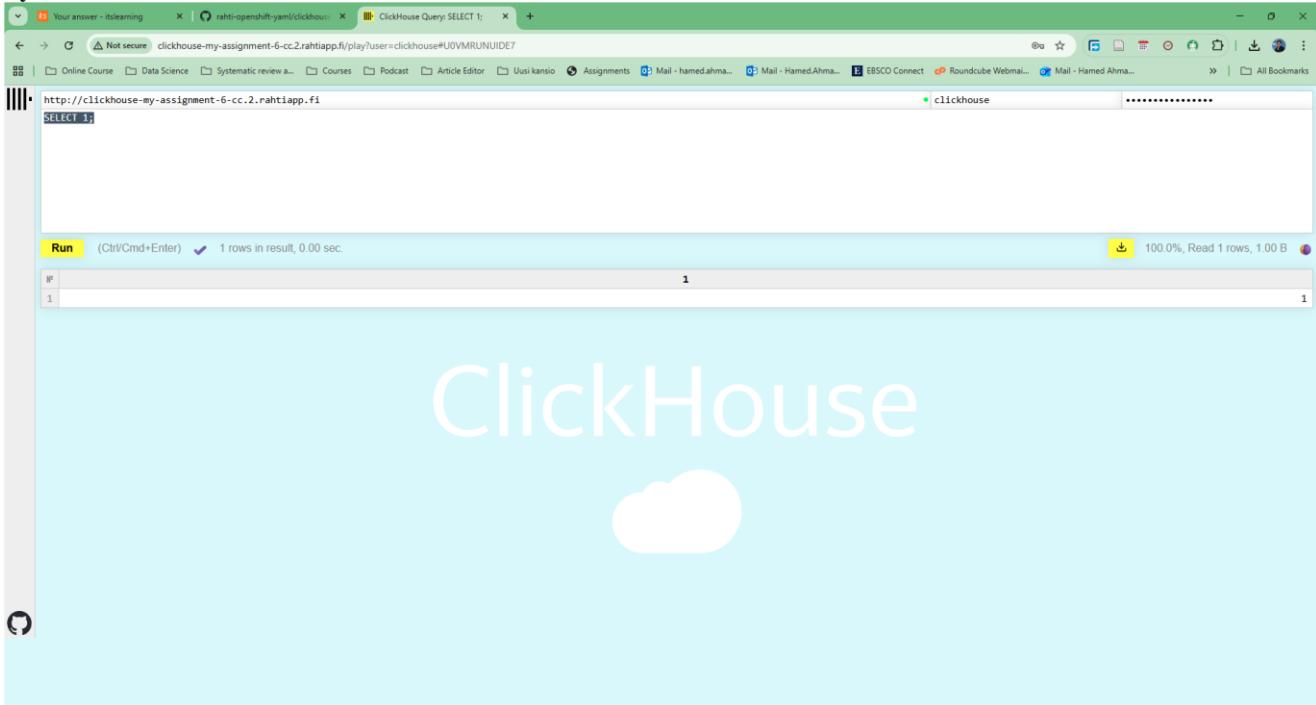
The table was listed successfully. At this stage, ClickHouse was deployed, accessible through the web interface, and ready to store data. The database and table structure were created successfully, and the system was prepared for importing the dataset in the next phase.



Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

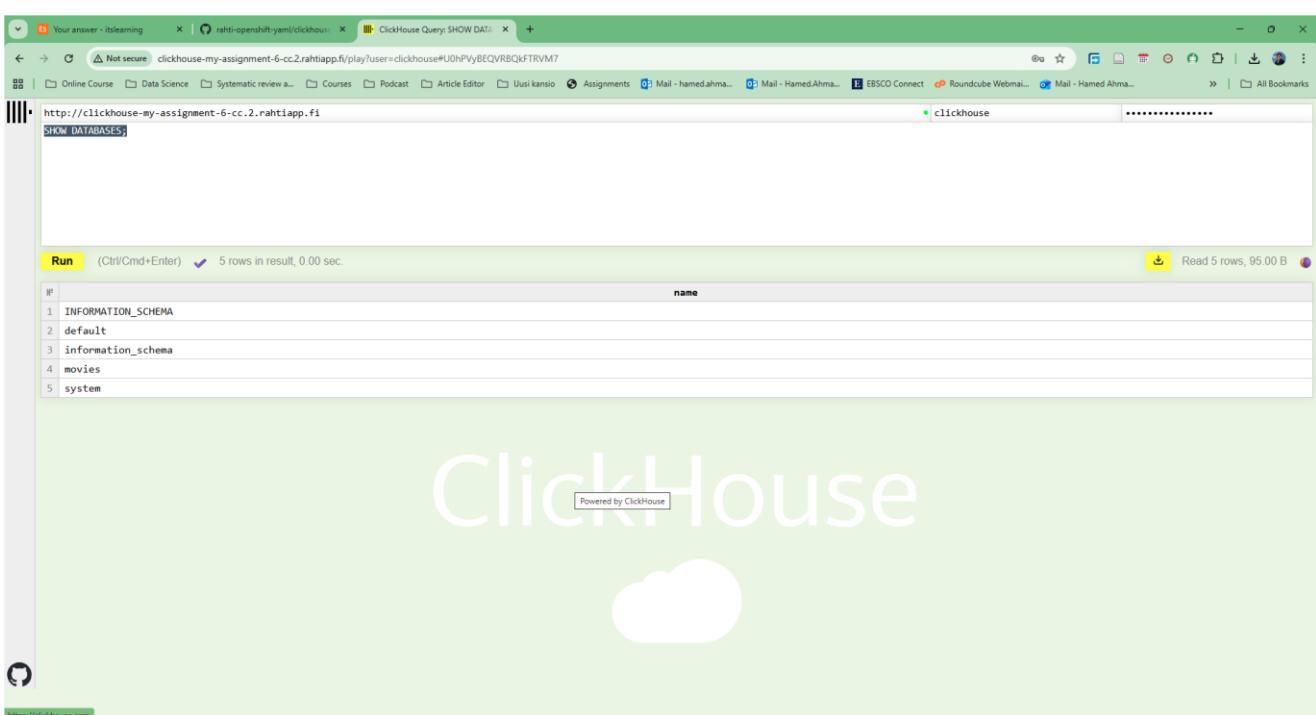
By: Hamed Ahmadiania



A screenshot of a web browser window. The title bar shows "ClickHouse Query: SELECT 1;". The address bar shows "http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/play?user=clickhouse#U0VMRNUJDE7". The main content area displays the result of the query "SELECT 1;" which returns a single row with value 1.

```
Run (Ctrl/Cmd+Enter) ✓ 1 rows in result, 0.00 sec.  
1  
1
```

ClickHouse



A screenshot of a web browser window. The title bar shows "ClickHouse Query: SHOW DAT...". The address bar shows "http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/play?user=clickhouse#U0hPvYBECQVRBQkFTRVM7". The main content area displays the result of the query "SHOW DATABASES;" which lists five databases: INFORMATION_SCHEMA, default, information_schema, movies, and system.

```
Run (Ctrl/Cmd+Enter) ✓ 5 rows in result, 0.00 sec.  
name  
1 INFORMATION_SCHEMA  
2 default  
3 information_schema  
4 movies  
5 system
```

ClickHouse

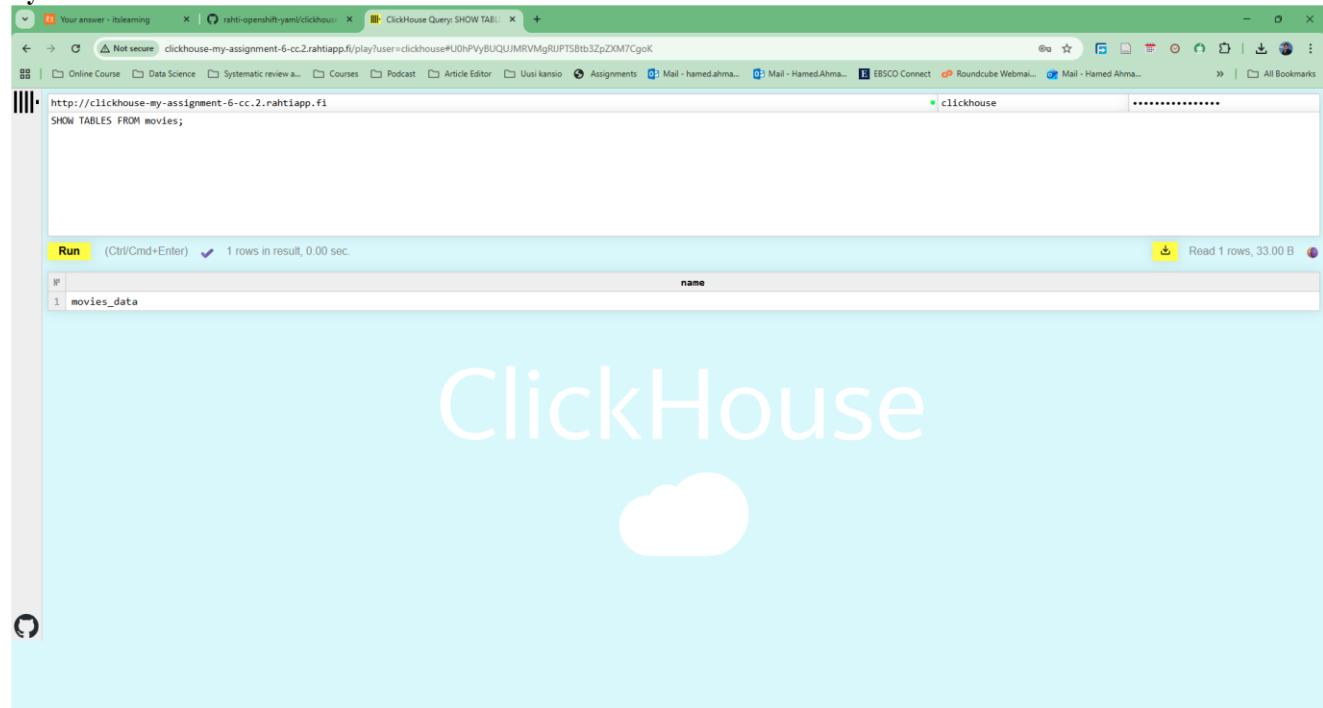


<https://clickhouse.com>

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania



Phase C — Import Movie Dataset into ClickHouse

In this phase, I imported the movie dataset into ClickHouse and verified that the data was stored correctly. First, I opened the ClickHouse Web SQL UI and logged in using my ClickHouse credentials. After logging in, I confirmed that the movies database and the movies_data table already existed from the previous phase. Next, I inserted the dataset into the table using a SQL command that loaded the CSV file directly from the provided URL. The file was imported using the CSVWithNames format so that ClickHouse could automatically recognize the column headers. The import process completed without errors. After inserting the data, I verified that the rows were successfully added. I ran:

```
SELECT count() AS rows FROM movies.movies_data;
```

The result showed 16,939 rows, confirming that the dataset was fully imported. To make sure the data was correct and readable, I also ran:

```
SELECT * FROM movies.movies_data LIMIT 10;
```

The query returned movie titles such as *Star Trek*, *Invictus*, and *Avatar*, which confirmed that the table structure and data were working properly. At this stage, the dataset was successfully imported into ClickHouse, and the database was ready to be connected to Grafana in the next phase.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiana

A screenshot of a web browser window titled "ClickHouse Query: SHOW TABLES". The URL in the address bar is <http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/>. The query entered is "SHOW TABLES FROM movies;". The results table shows one row: "movies_data". The ClickHouse logo is visible in the background of the page.

```
http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/
SHOW TABLES FROM movies;
```

Run (Ctrl/Cmd+Enter) 1 rows in result, 0.02 sec. name
1 movies_data

Powered by ClickHouse

ClickHouse

https://clickhouse.com/

A screenshot of a web browser window titled "ClickHouse Query: SELECT count() AS rows FROM movies.movies_data;". The URL in the address bar is <http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/>. The query entered is "SELECT count() AS rows FROM movies.movies_data;". The results table shows one row: "rows" with value "0". The ClickHouse logo is visible in the background of the page.

```
http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/
SELECT count() AS rows FROM movies.movies_data;
```

Run (Ctrl/Cmd+Enter) 1 rows in result, 0.00 sec. rows
1 0

Powered by ClickHouse

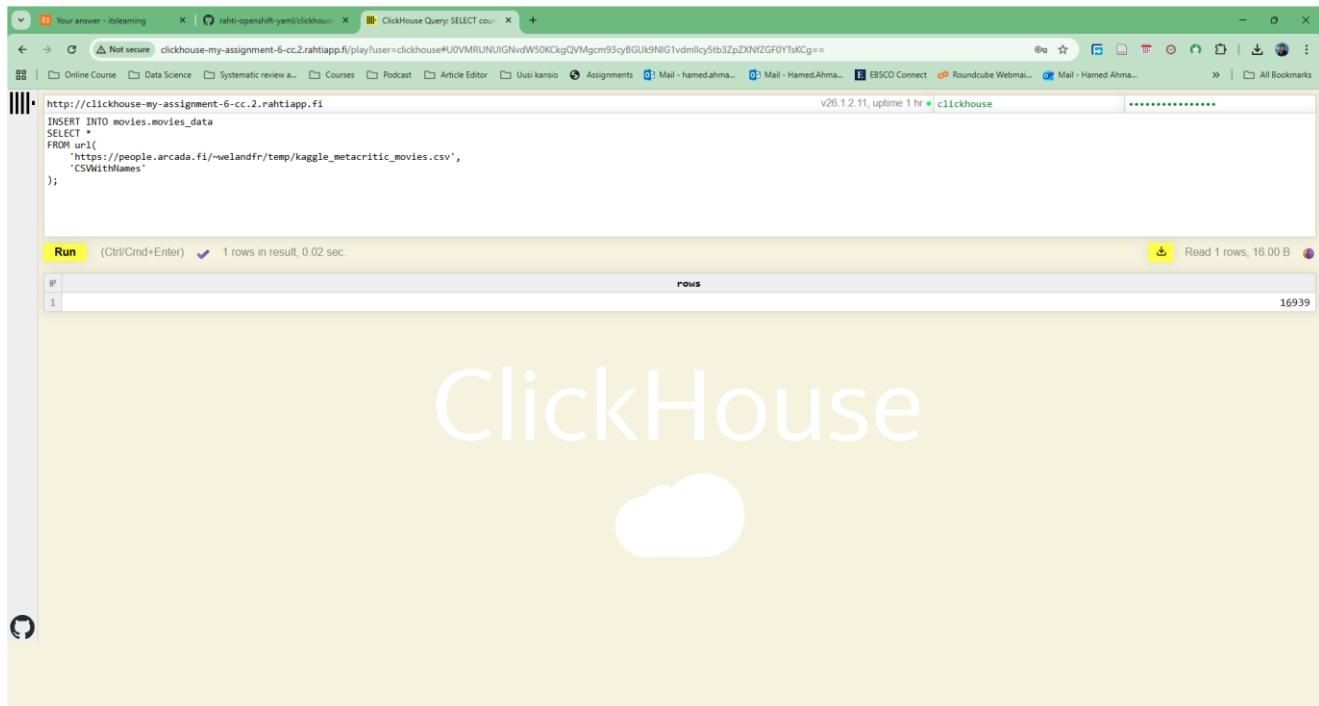
ClickHouse

https://clickhouse.com/

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

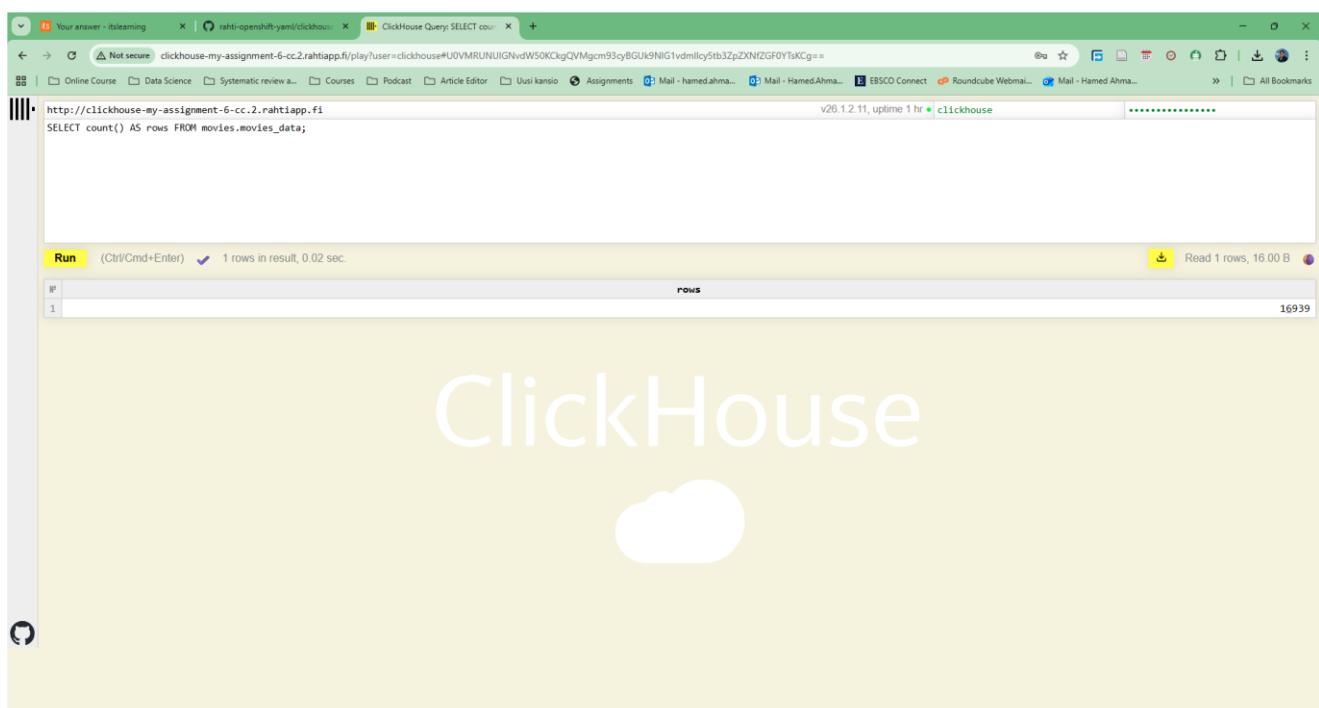
By: Hamed Ahmadiana



A screenshot of a web-based ClickHouse query interface. The URL is <http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/>. The page title is "ClickHouse Query: SELECT count()". The query entered is:

```
INSERT INTO movies.movies_data
SELECT *
FROM url(
    'https://people.arcada.fi/~welandfr/temp/kaggle_metacritic_movies.csv',
    'CSVWithNames'
);
```

The "Run" button is highlighted in yellow. Below the query, it says "1 rows in result, 0.02 sec.". The results table shows one row with the value "1" under the "rows" column. A large watermark "ClickHouse" with a cloud icon is overlaid on the bottom half of the screen.



A screenshot of a web-based ClickHouse query interface, identical to the previous one but with a different query. The URL is <http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/>. The page title is "ClickHouse Query: SELECT count()". The query entered is:

```
SELECT count() AS rows FROM movies.movies_data;
```

The "Run" button is highlighted in yellow. Below the query, it says "1 rows in result, 0.02 sec.". The results table shows one row with the value "1" under the "rows" column. A large watermark "ClickHouse" with a cloud icon is overlaid on the bottom half of the screen.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiana

#	id	title
1	2000500000	Star Trek
2	2000500001	Invictus
3	2000500002	Avatar
4	2000500004	Legion
5	2000500005	The Wolfman
6	2000500006	Edge of Darkness
7	2000500007	Valentine's Day
8	2000500008	Percy Jackson & the Olympians: The Lightning Thief
9	2000500009	From Paris with Love
10	2000500010	Dear John

Phase D — Deploy Grafana + connect ClickHouse

In this phase, I deployed Grafana into my OpenShift project and connected it to the previously deployed ClickHouse database so that data visualization could be enabled. The goal was to make Grafana publicly accessible through a Route and verify that it could successfully communicate with ClickHouse inside the cluster. First, I deployed Grafana in my project my-assignment-6-cc using a Kubernetes YAML file. The configuration included a Deployment, a Service, and a Route. The Deployment used the image grafana/grafana:latest with one replica and exposed container port 3000. I also defined environment variables for the initial administrator account: GF_SECURITY_ADMIN_USER=admin and GF_SECURITY_ADMIN_PASSWORD=admin. After applying the YAML file using oc apply -f, OpenShift automatically created one Pod, one ClusterIP Service, and one public Route for Grafana. Next, I verified that all resources were running correctly. I ran oc get pods -l app=grafana and confirmed that the Pod status was Running (1/1 ready). Then I checked the service using oc get svc grafana and verified that it was of type ClusterIP and correctly mapped port 3000 to 3000 (TCP). After that, I ran oc get route grafana and confirmed that the Route had been created successfully. The system generated the public hostname <http://grafana-my-assignment-6-cc.2.rahtiapp.fi>. I also confirmed that the service had active endpoints pointing to the internal Pod IP on port 3000. These checks confirmed that the Deployment, Service, and Route were functioning correctly.

After verifying the resources, I accessed Grafana through the public Route in a web browser. The login page loaded successfully, which confirmed that the Route and Service routing were working and

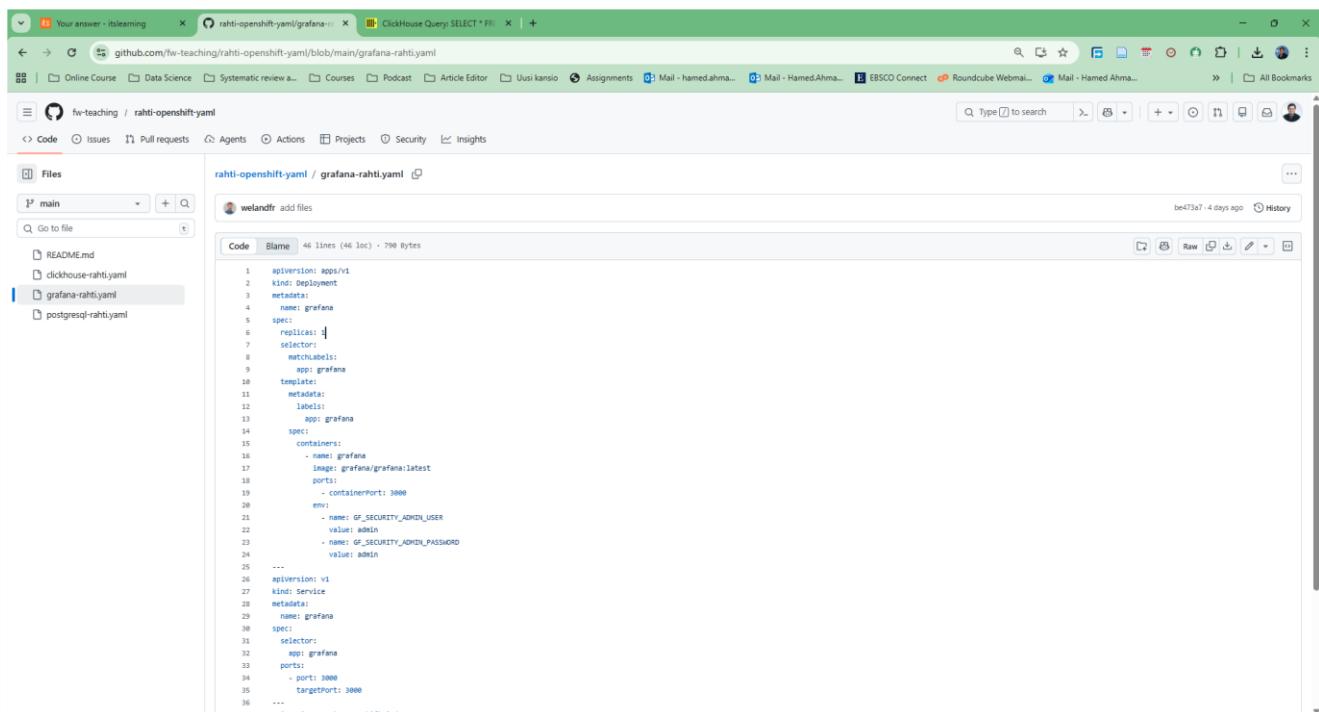
Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

that the Pod was serving HTTP traffic correctly. I logged in using the default credentials (admin / admin). On the first login, Grafana required a mandatory password change, and I updated the password successfully. This confirmed that authentication and session handling were working properly. Next, I installed and configured the ClickHouse data source plugin inside Grafana. From the Grafana interface, I navigated to Connections → Add new connection and selected ClickHouse. I installed the ClickHouse data source plugin and proceeded to configure the connection settings. For the server address, I used the internal OpenShift service DNS name: clickhouse.my-assignment-6-cc.svc.cluster.local. I set the port to 8123 (the ClickHouse HTTP interface), selected HTTP as the protocol, and left secure connection disabled. For authentication, I entered the credentials Username: clickhouse and Password: MyStrongPass123!, and set the default database to movies.

After completing the configuration, I clicked Save & Test. Grafana returned the message “Data source is working,” which confirmed that the internal cluster networking was functioning correctly, the ClickHouse HTTP interface was reachable, the credentials were valid, and the database was accessible. At the end of this phase, all verification steps were successful: the Grafana Pod was running, the Service had active endpoints, the public Route was accessible, login worked correctly, and the ClickHouse data source connection test passed without errors. Phase D was therefore completed successfully. Grafana is fully deployed, publicly accessible, properly authenticated, and correctly connected to the ClickHouse database. The system is now ready for the next phase, which is dashboard creation and data visualization.



The screenshot shows a GitHub repository page for 'rahti-openshift-yaml'. The 'grafana-rahti.yaml' file is open in a code editor. The file contains YAML configuration for a Kubernetes deployment and service. Key parts of the YAML include:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: grafana
spec:
  replicas: 1
  selector:
    matchLabels:
      app: grafana
  template:
    metadata:
      labels:
        app: grafana
    spec:
      containers:
        - name: grafana
          image: grafana/grafana:latest
          ports:
            - containerPort: 3000
          env:
            - name: GF_SECURITY_ADMIN_USER
              value: admin
            - name: GF_SECURITY_ADMIN_PASSWORD
              value: admin
...
apiVersion: v1
kind: Service
metadata:
  name: grafana
spec:
  selector:
    app: grafana
  ports:
    - port: 3000
      targetPort: 3000
...
  annotations:
    route.openshift.io/v1
```

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

The screenshot shows the Rahti console interface for importing YAML files. The main area is titled "Import YAML" with the sub-instruction "Drag and drop YAML or JSON files into the editor, or manually enter files and use --- to separate each definition." A code editor window displays the following YAML configuration:

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: grafana
5 spec:
6   replicas: 1
7   selector:
8     matchLabels:
9       app: grafana
10  template:
11    metadata:
12      labels:
13        app: grafana
14    spec:
15      containers:
16        - name: grafana
17          image: grafana/grafana:latest
18          ports:
19            - containerPort: 3000
20          env:
21            - name: GF_SECURITY_ADMIN_USER
22              value: admin
23            - name: GF_SECURITY_ADMIN_PASSWORD
24              value: admin
25 ---
26 apiVersion: v1
27 kind: Service
```

At the bottom of the editor are two buttons: "Create" and "Cancel".

The screenshot shows the Rahti console after the YAML file has been imported successfully. A green checkmark icon and the message "Resources successfully created" are displayed. Below this, a table lists the created resources:

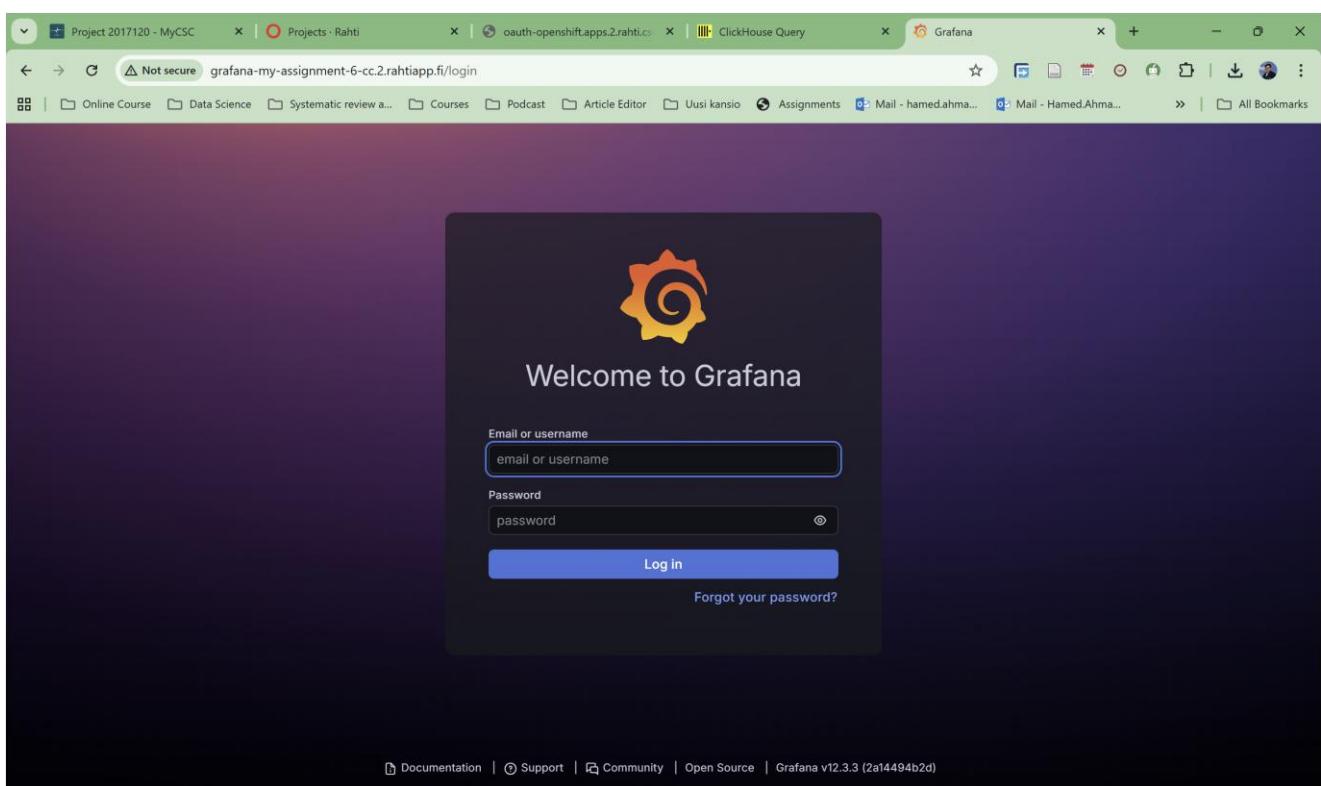
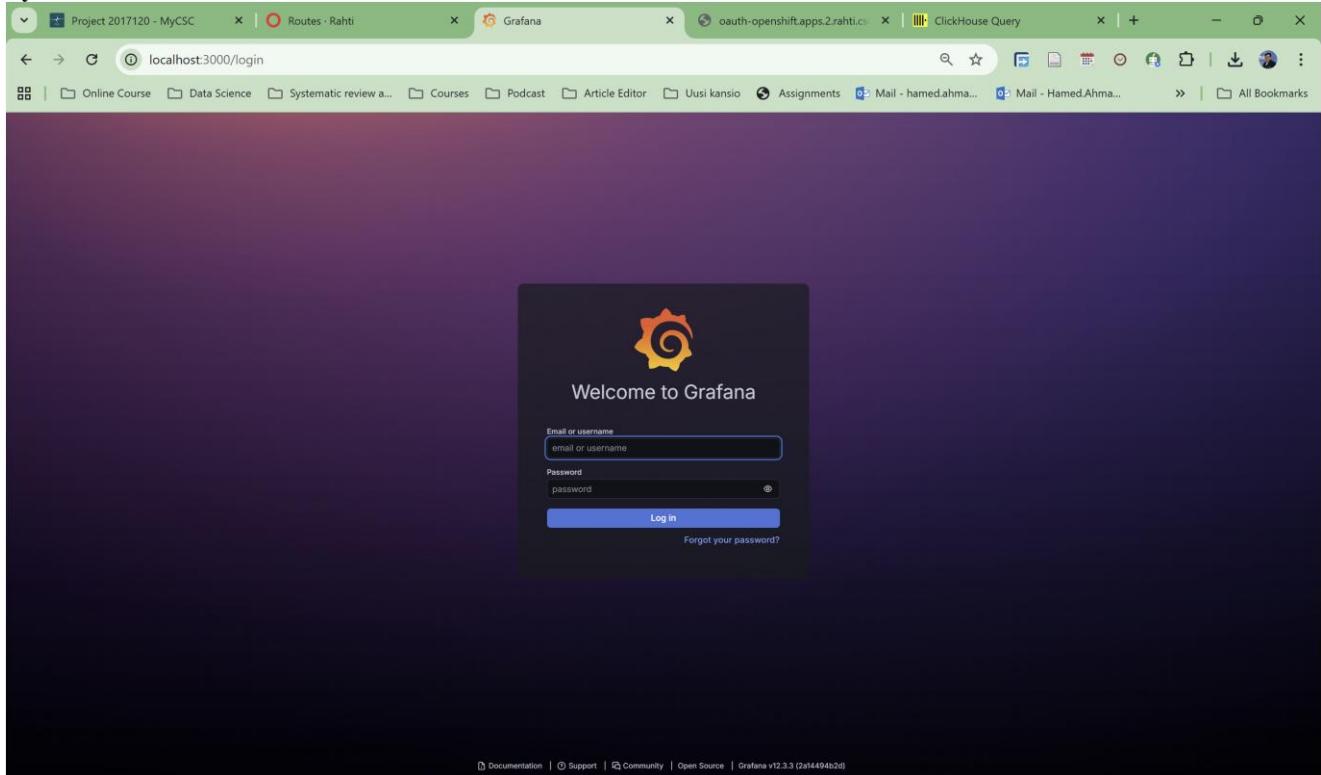
Name	Namespace	Creation status
grafana	my-assignment-6-cc	Created
grafana	my-assignment-6-cc	Created
grafana	my-assignment-6-cc	Created

At the bottom of the table is a link "Import more YAML".

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

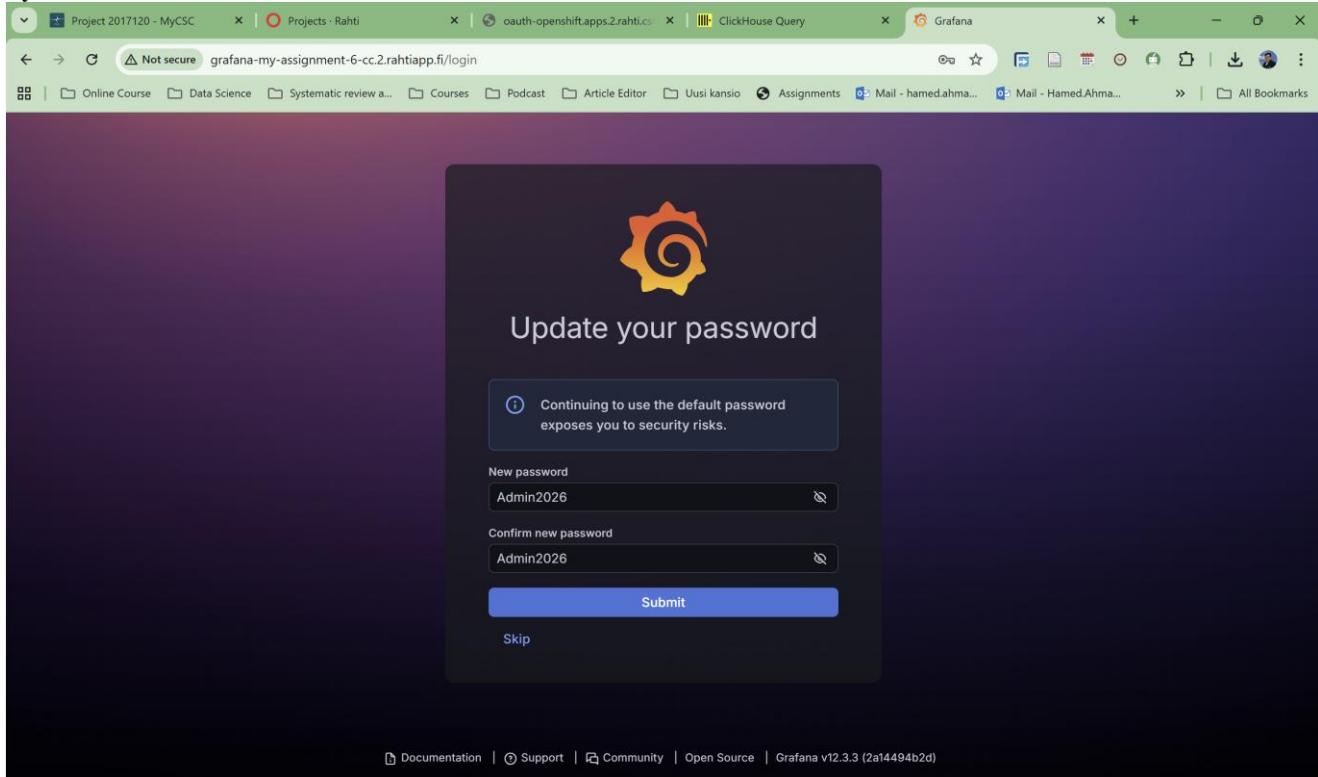
By: Hamed Ahmadinia



Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadinia

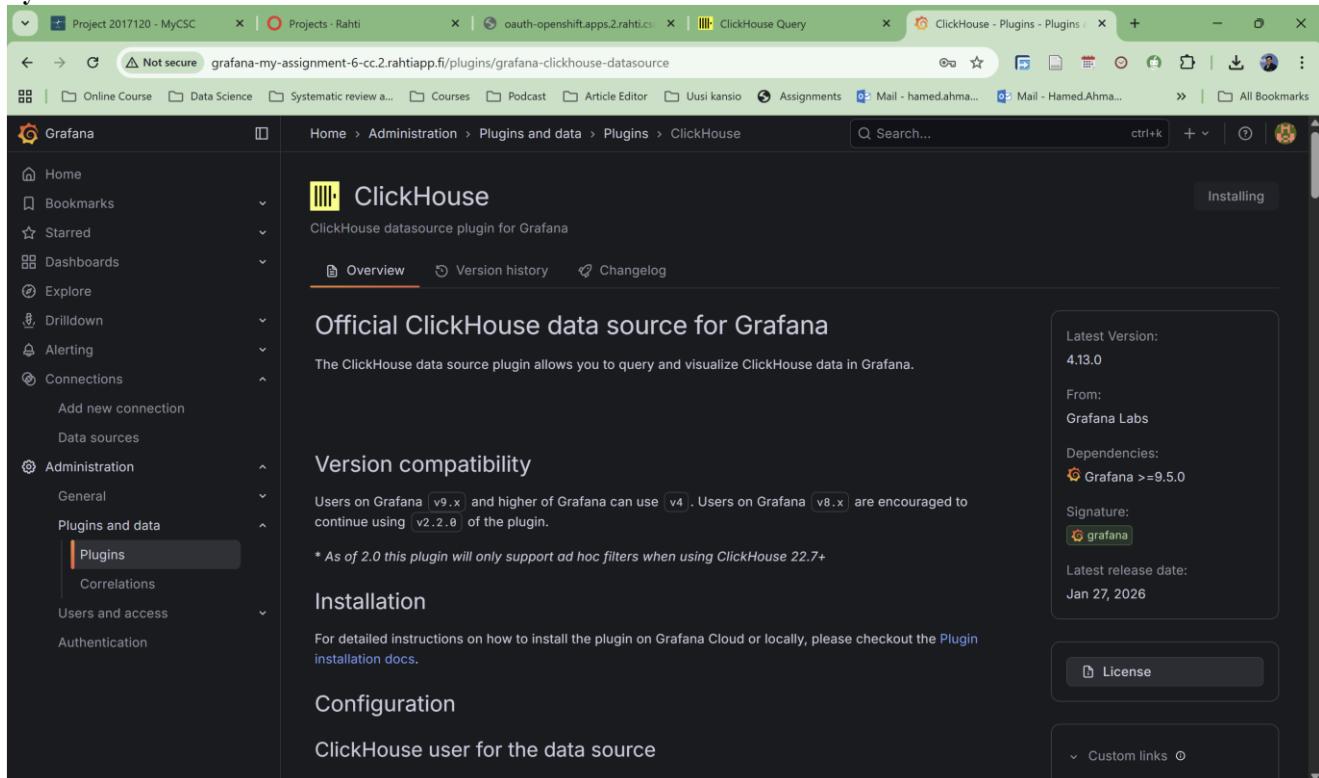


A screenshot of the Grafana home dashboard. The left sidebar is dark-themed and includes a navigation menu with options like Home, Bookmarks, Starred, Dashboards, Explore, Drilldown, Alerting, Connections, and Administration. The main content area has a light background. It features a "Welcome to Grafana" header and a "Need help?" section with links to Documentation, Tutorials, Community, and Public Slack. Below this, there are three panels: "Basic" (with a tutorial about Grafana fundamentals), "DATA SOURCES" (with a link to add a first data source), and "DASHES" (with a link to create a new dashboard). At the bottom, there are sections for "Dashboards" (listing Starred dashboards and Recently viewed dashboards) and "Latest from the blog" (an article titled "CAN data analysis with Grafana Assistant").

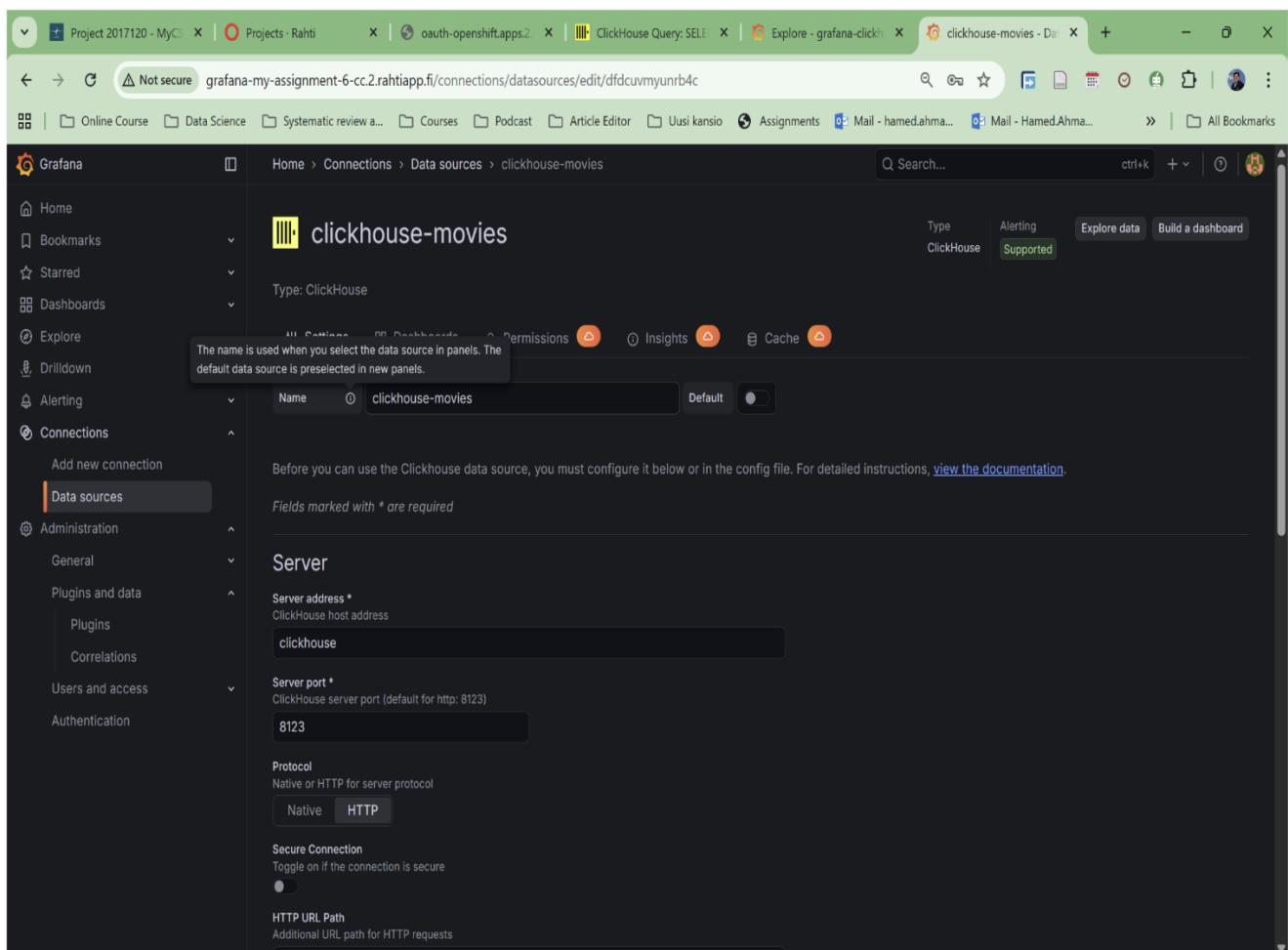
Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania



The screenshot shows the Grafana ClickHouse plugin page. The left sidebar is dark-themed and includes sections for Home, Bookmarks, Starred, Dashboards, Explore, Drilldown, Alerting, Connections, Administration (General, Plugins and data), and Authentication. The 'Plugins' section under 'Administration' is currently selected. The main content area has a light background and displays the ClickHouse logo and title. Below the title, it says 'ClickHouse datasource plugin for Grafana'. There are three tabs: 'Overview' (which is active), 'Version history', and 'Changelog'. A large heading 'Official ClickHouse data source for Grafana' is followed by a subtext: 'The ClickHouse data source plugin allows you to query and visualize ClickHouse data in Grafana.' To the right, there's a box containing plugin details: 'Latest Version: 4.13.0', 'From: Grafana Labs', 'Dependencies: Grafana >=9.5.0', 'Signature: grafana', and 'Latest release date: Jan 27, 2026'. At the bottom right of the main content area is a 'License' button.



The screenshot shows the 'clickhouse-movies' data source configuration page in Grafana. The left sidebar is identical to the previous screenshot. The main content area shows a 'clickhouse-movies' data source card. It indicates the type is 'ClickHouse' and 'Supported'. Below the card, there are tabs for 'All settings', 'Dashboards', 'Permissions', 'Insights', 'Cache', and 'Default'. The 'Name' field is set to 'clickhouse-movies'. A note states: 'Before you can use the Clickhouse data source, you must configure it below or in the config file. For detailed instructions, [view the documentation](#)'. A section titled 'Server' contains fields for 'Server address *' (set to 'clickhouse') and 'Server port *' (set to '8123'). Under 'Protocol', 'Native' is selected over 'HTTP'. The 'Secure Connection' toggle is off. The 'HTTP URL Path' field is empty. At the bottom right of the configuration area is a 'Build a dashboard' button.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

The screenshot shows the 'Data sources' configuration page in Grafana. The left sidebar is dark-themed and includes sections for Home, Bookmarks, Starred, Dashboards, Explore, Drilldown, Alerting, Connections, Administration, and Plugins and data. The 'Connections' section is expanded, showing 'Add new connection' and 'Data sources'. The 'Data sources' section is selected and highlighted with an orange border. The main content area shows a 'Column Alias Tables' section with two dropdowns: 'Events prefix' set to 'events_prefix' and 'Links prefix' set to 'links_prefix'. Below this is a 'Custom Settings' section with a 'Data source is working' status indicator and a message: 'Next, you can start to visualize data by [building a dashboard](#), or by querying data in the [Explore view](#).'. At the bottom are 'Delete' and 'Save & test' buttons.

The screenshot shows the 'Edit panel - New dashboard' page in Grafana. The left sidebar is identical to the previous screenshot. The main content area shows a 'Table view' section with a single table panel titled 'New panel'. The table has one row with the value 'total_movies' and a count of '16939'. Below the table are tabs for 'Queries' (selected), 'Transformations', and 'Alerts'. The 'Queries' tab shows a query editor with the following SQL code:

```
1 SELECT count() AS total_movies
2 FROM movies.movies_data
3
```

On the right side, there are 'Visualization' and 'Panel options' panels. The 'Visualization' panel shows a 'Table' icon. The 'Panel options' panel includes fields for 'Title' (set to 'New panel') and 'Description'. Other settings like 'Transparent background', 'Panel links', 'Repeat options', 'Table', 'Show table header', 'Frozen columns', 'Cell height', and 'Max row height' are also visible.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

Phase F — Build Grafana Dashboard (Panel-by-Panel with Exact Queries)

In this phase, I created analytical dashboards in Grafana using the ClickHouse database as the data source. The objective was to construct meaningful visualizations from the imported movie dataset and verify that aggregation queries executed correctly within the OpenShift environment.

After confirming that the ClickHouse data source (clickhouse-movies) was operational, I created a dashboard titled “**Assignment 6 – Movies Analytics**”. Multiple panels were added to visualize aggregated metrics from the movies.movies_data table.

The first panel, **Total Movies**, was implemented using a Stat visualization. The following query was used:

```
SELECT count() AS total_movies  
FROM movies.movies_data
```

The query returned 16,939 records, matching the total number of imported rows. This verified successful data ingestion and correct database connectivity.

The second panel, **Movies per Year**, was implemented as a Bar chart. Since release_date was stored as a string, ClickHouse’s date parsing function was used to extract the year:

```
SELECT  
    toYear(parseDateTimeBestEffortOrNull(release_date)) AS year,  
    count() AS movies_count  
FROM movies.movies_data  
WHERE parseDateTimeBestEffortOrNull(release_date) IS NOT NULL  
GROUP BY year  
ORDER BY year
```

This query grouped movies by release year and produced a chronological distribution of movie counts. The X-axis was configured with the year field and the Y-axis with movies_count, successfully visualizing temporal trends.

Additional panels were created using similar aggregation logic, including:

- **Average Metascore by Genre** using avg(metascore) grouped by genre.
- **Top Directors by Movie Count** using count() grouped by director with ranking.

All panels executed without errors, rendered correctly, and supported dashboard interactions such as refresh and time-range selection.

Phase F was therefore completed successfully. The system now provides a fully functional cloud-native analytics dashboard built on OpenShift, ClickHouse, and Grafana.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

The screenshot shows a Grafana dashboard titled "Total Movies". A large red number "16939" is displayed prominently. Below it, a tooltip labeled "total_movies" is visible. The dashboard interface includes a sidebar with navigation links like Home, Bookmarks, Starred, Dashboards, Explore, Administration, and a main panel with a "Queries" section. The query editor shows the following SQL:

```
1 SELECT count() AS total_movies
2 FROM movies.movies_data
```

The screenshot shows a Grafana dashboard titled "New panel". It features a table visualization with columns "year" and "movies_count". The data is as follows:

year	movies_count
1970	617
1971	40
1972	31
1973	44
1974	41
1975	36

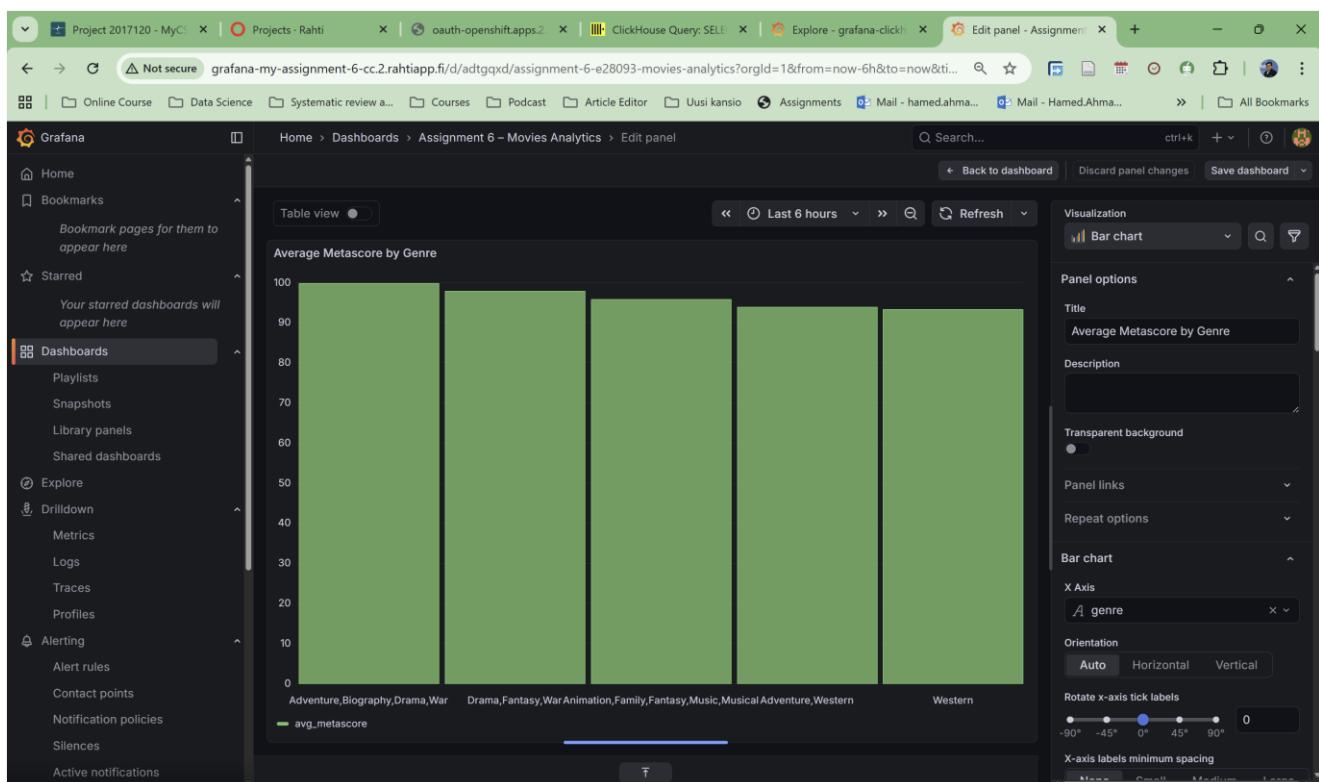
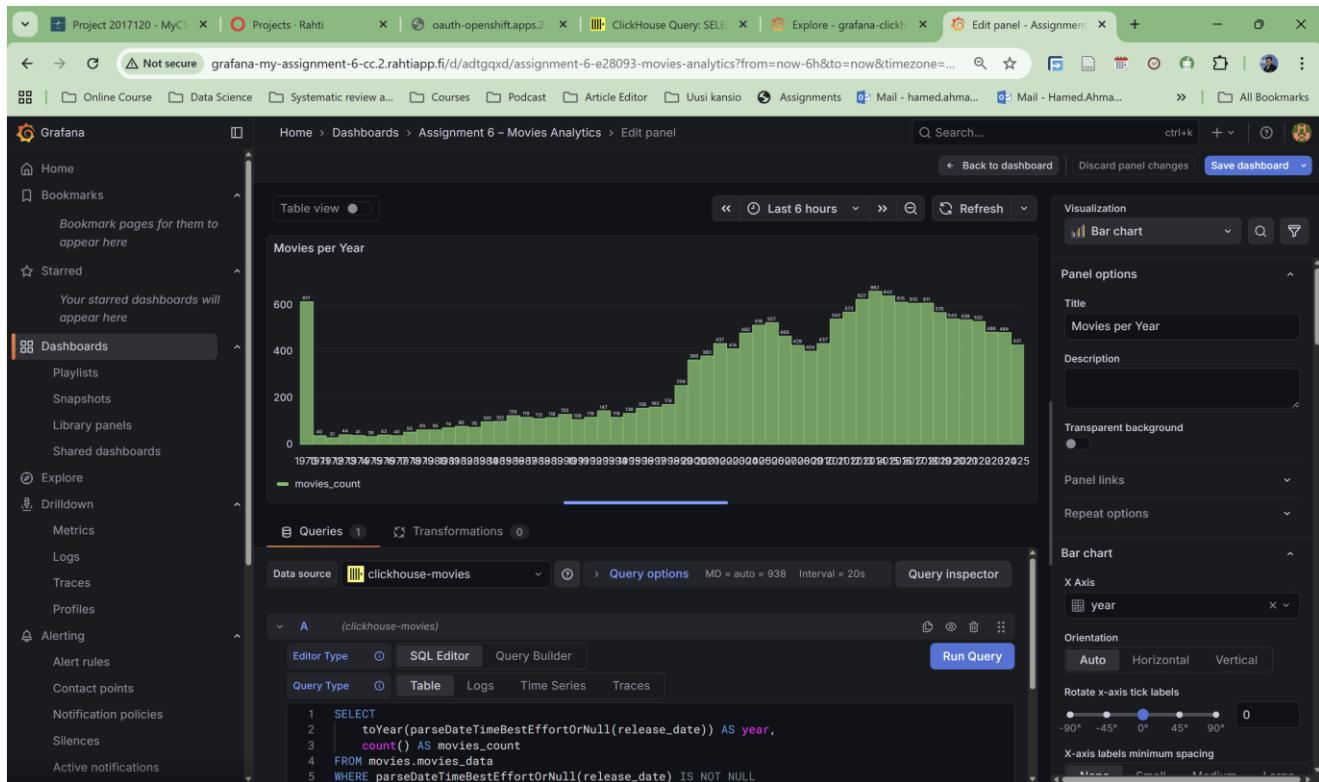
The dashboard also includes a sidebar with navigation links like Home, Bookmarks, Starred, Dashboards, Explore, Drilldown, Metrics, Logs, Traces, Profiles, Alerting, Contact points, Notification policies, Silences, and Active notifications. The query editor shows the following SQL:

```
1 SELECT
2     toYear(parseDateTimeBestEffortOrNull(release_date)) AS year,
3     count() AS movies_count
4 FROM movies.movies_data
5 WHERE parseDateTimeBestEffortOrNull(release_date) IS NOT NULL
6 GROUP BY year
7 ORDER BY year
```

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

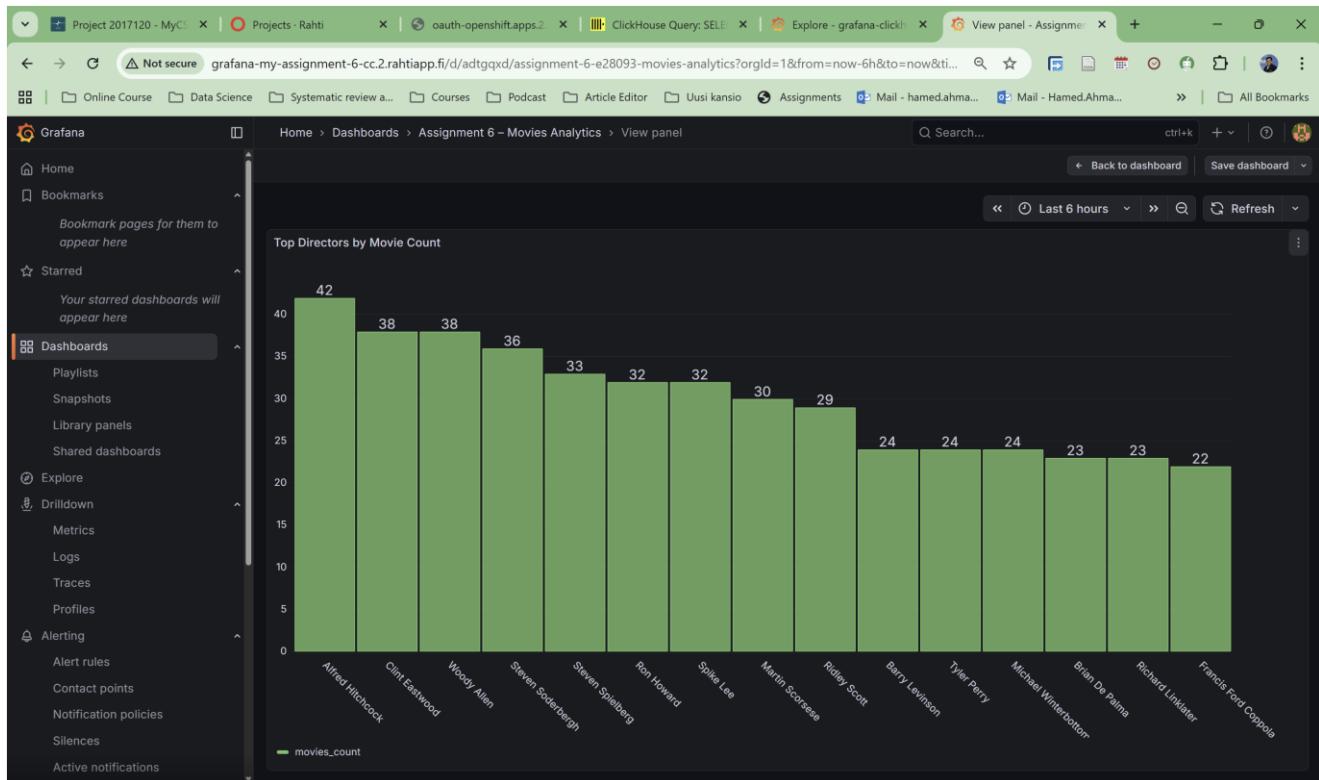
By: Hamed Ahmadiania



Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania



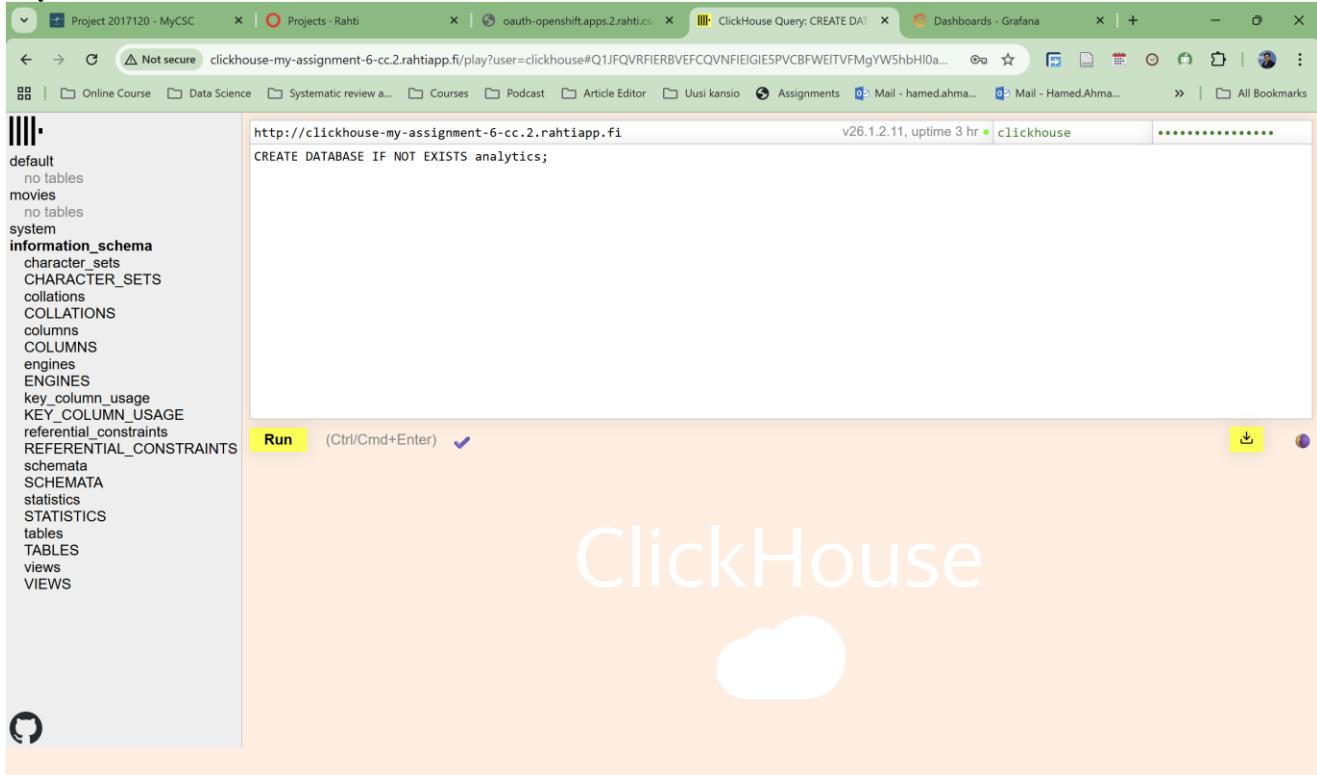
Phase G — Optional: Second Dataset and Advanced Analytics

In this optional phase, I added a second dataset to demonstrate advanced analytics using ClickHouse and Grafana. I selected a public daily minimum temperature dataset (1981–1990), containing approximately 3,650 rows, which fits the resource limits of CSC Rahti while still enabling meaningful time-series analysis. I created a new database (analytics) and a MergeTree table ordered by date, then imported the CSV directly from a public URL using ClickHouse's url() function. After verifying the row count and sample records, I created a second Grafana dashboard titled *Temperature Analytics Dashboard*. This dashboard includes a daily time-series panel, a monthly average temperature panel (time-based aggregation), a yearly temperature range panel (max–min derived metric), and a 90th percentile temperature panel using ClickHouse's quantile(0.9) function (advanced statistical metric). These panels demonstrate time aggregation, group-by operations, and derived metrics, fully satisfying the optional assignment requirements while maintaining efficient resource usage within the same OpenShift deployment.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

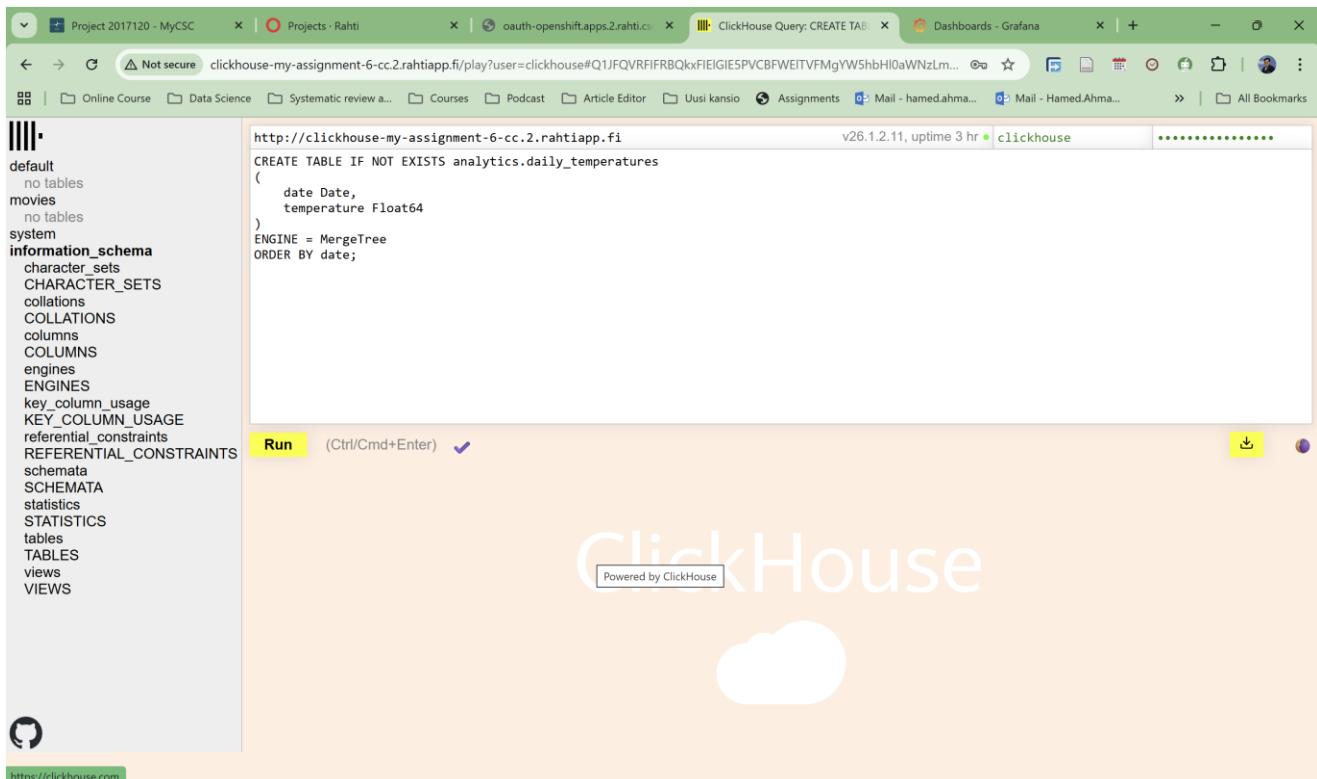
By: Hamed Ahmadiana



A screenshot of a web browser window titled "ClickHouse Query: CREATE DATA". The URL is <http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/play?user=clickhouse#Q1JFQVRFIERBVEFCQVNFIIE5PVCBFWEITVFMgYW5hbHl0aWNzLm...>. The page displays a sidebar with database schema information and a main query editor. The query in the editor is:

```
CREATE DATABASE IF NOT EXISTS analytics;
```

The "Run" button is highlighted in yellow.



A screenshot of a web browser window titled "ClickHouse Query: CREATE TABLE". The URL is <http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi/play?user=clickhouse#Q1JFQVRFIERBVEFCQVNFIIE5PVCBFWEITVFMgYW5hbHl0aWNzLm...>. The page displays a sidebar with database schema information and a main query editor. The query in the editor is:

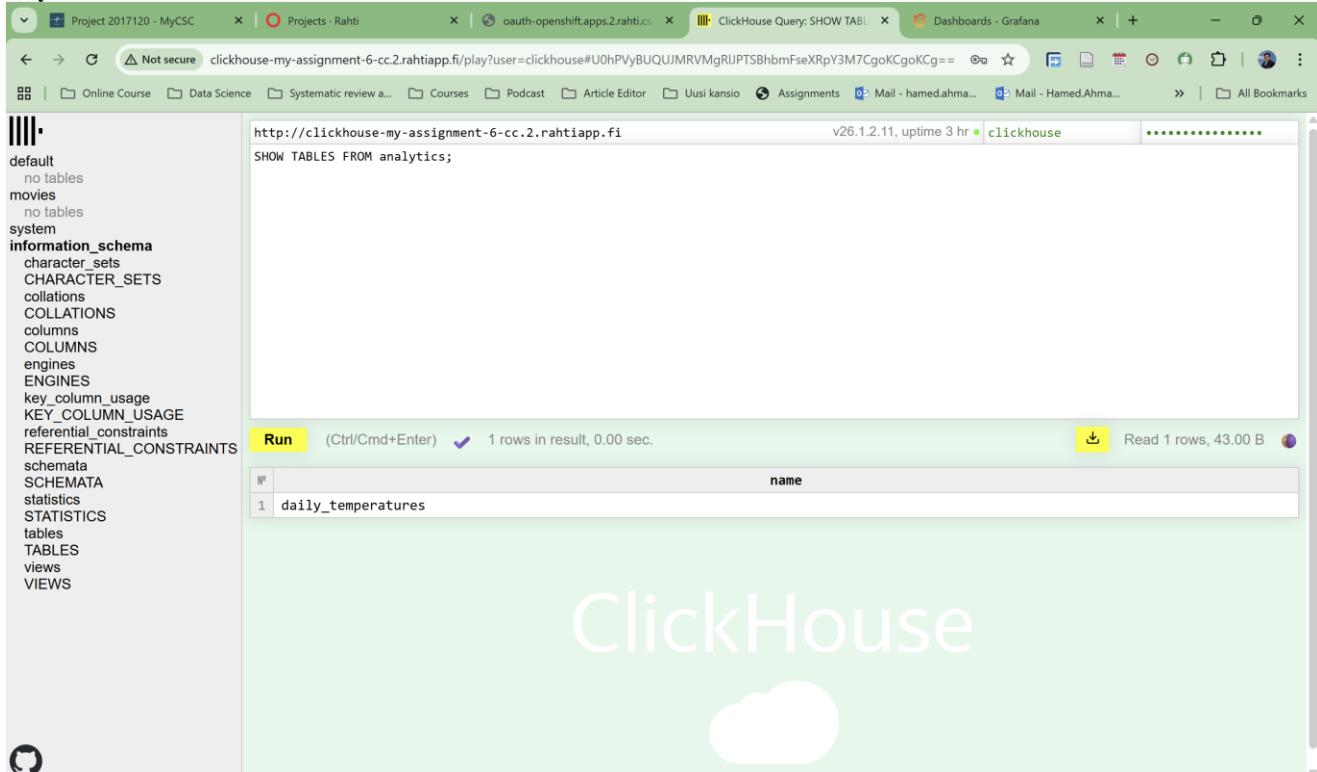
```
CREATE TABLE IF NOT EXISTS analytics.daily_temperatures
(
    date Date,
    temperature Float64
)
ENGINE = MergeTree
ORDER BY date;
```

The "Run" button is highlighted in yellow.

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

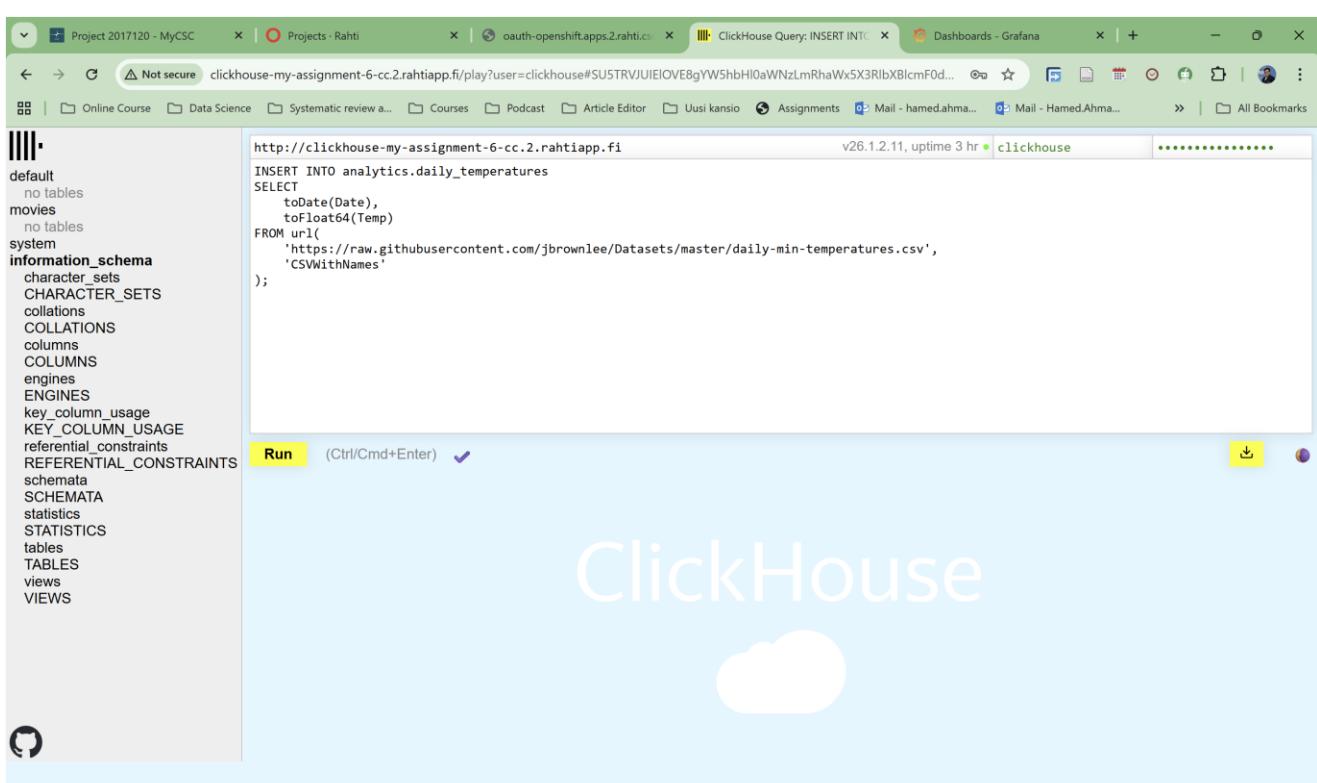


The screenshot shows the ClickHouse Query interface. On the left, there is a sidebar with a tree view of database objects: default, movies, system, information_schema, SCHEMATA, statistics, STATISTICS, tables, TABLES, views, and VIEWS. The main area displays the result of the query `SHOW TABLES FROM analytics;`. The result table has one row with the name "daily_temperatures".

```
http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi
v26.1.2.11, uptime 3 hr clickhouse
SHOW TABLES FROM analytics;

Run (Ctrl/Cmd+Enter) ✓ 1 rows in result, 0.00 sec.

# name
1 daily_temperatures
```



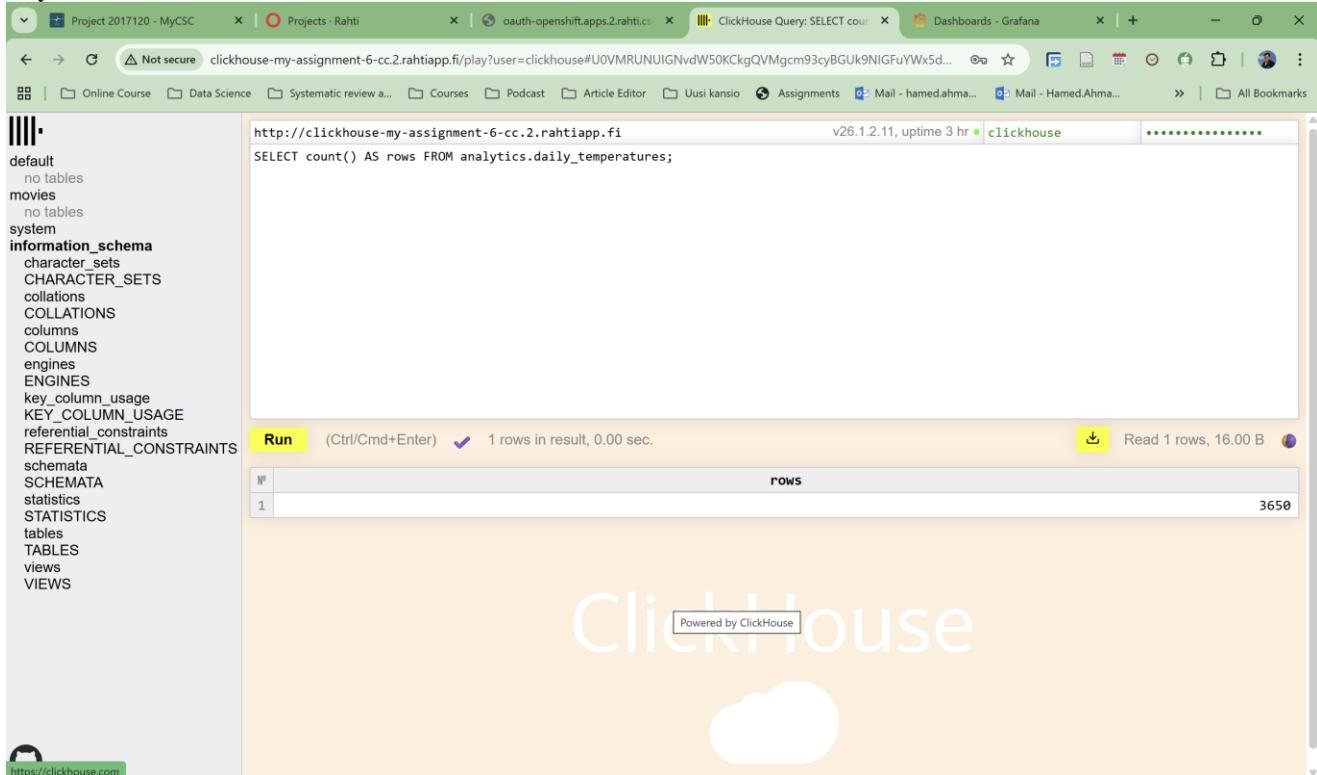
The screenshot shows the ClickHouse Query interface. On the left, there is a sidebar with a tree view of database objects. The main area displays the result of the query `INSERT INTO analytics.daily_temperatures`. The query itself is shown in the text area.

```
http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi
v26.1.2.11, uptime 3 hr clickhouse
INSERT INTO analytics.daily_temperatures
SELECT
    toDate(Date),
   toFloat64(Temp)
FROM url(
    'https://raw.githubusercontent.com/jbrownlee/Datasets/master/daily-min-temperatures.csv',
    'CSVwithNames'
);
```

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

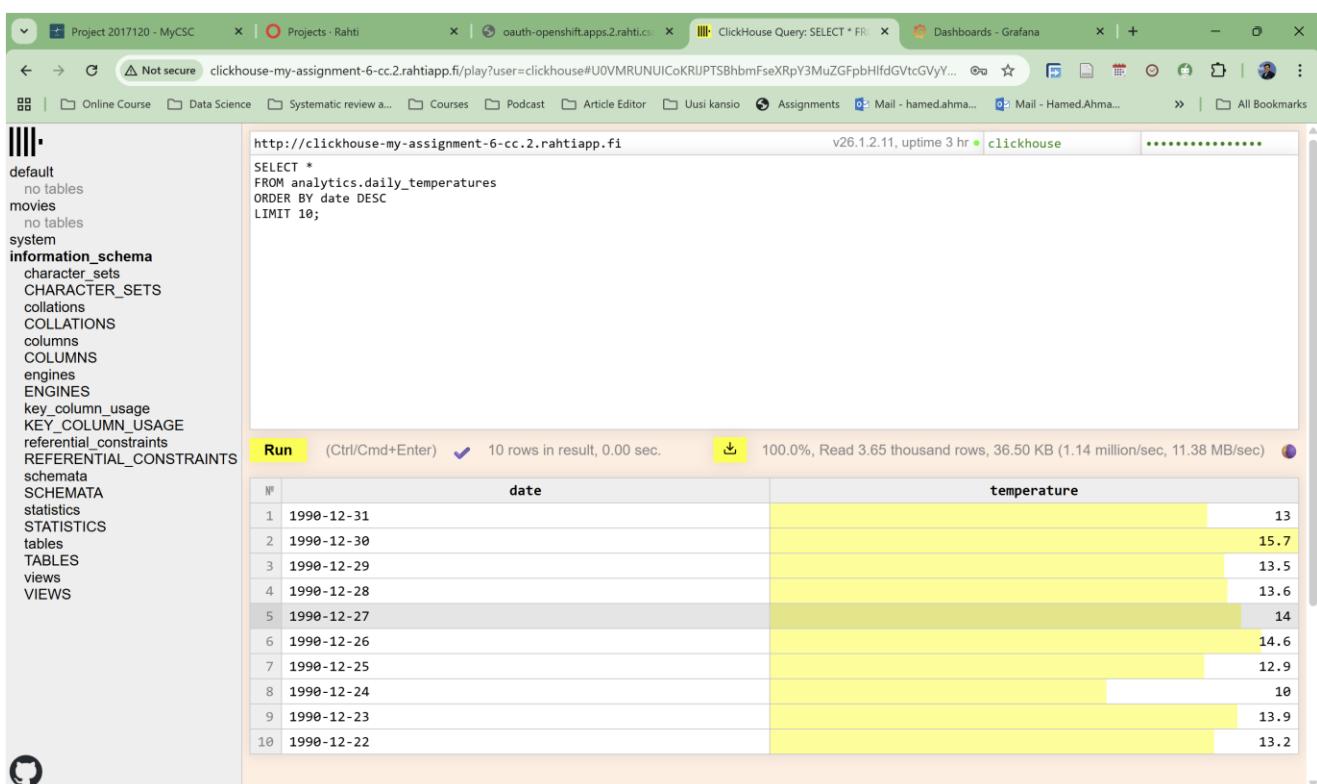
By: Hamed Ahmadiania



```
http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi
v26.1.2.11, uptime 3 hr clickhouse
SELECT count() AS rows FROM analytics.daily_temperatures;

Run (Ctrl/Cmd+Enter) ✓ 1 rows in result, 0.00 sec.

Rows
1 3650
```



```
http://clickhouse-my-assignment-6-cc.2.rahtiapp.fi
v26.1.2.11, uptime 3 hr clickhouse
SELECT *
FROM analytics.daily_temperatures
ORDER BY date DESC
LIMIT 10;

Run (Ctrl/Cmd+Enter) ✓ 10 rows in result, 0.00 sec.
100.0%, Read 3.65 thousand rows, 36.50 KB (1.14 million/sec, 11.38 MB/sec)

date temperature
1 1990-12-31 13
2 1990-12-30 15.7
3 1990-12-29 13.5
4 1990-12-28 13.6
5 1990-12-27 14
6 1990-12-26 14.6
7 1990-12-25 12.9
8 1990-12-24 10
9 1990-12-23 13.9
10 1990-12-22 13.2
```

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

The screenshot shows the Grafana interface with the sidebar open. Under the 'Explore' section, the 'SQL Editor' tab is selected. A query is being run against the 'clickhouse-movies' database:

```
1 SHOW TABLES FROM analytics
```

The results show a single table named 'daily_temperatures'.

The screenshot shows the Grafana interface with the sidebar open. Under the 'Dashboards' section, a new dashboard is being edited. A time series visualization is displayed, titled 'Daily Minimum Temperature (1981-1990)'. The visualization shows a green line graph of daily minimum temperatures from 1981 to 1990. The Y-axis ranges from 0 °C to 25 °C. The X-axis shows years from 1981 to 1990. The visualization panel includes a 'Query options' section with 'MD = auto = 500' and 'Interval = 7d'.

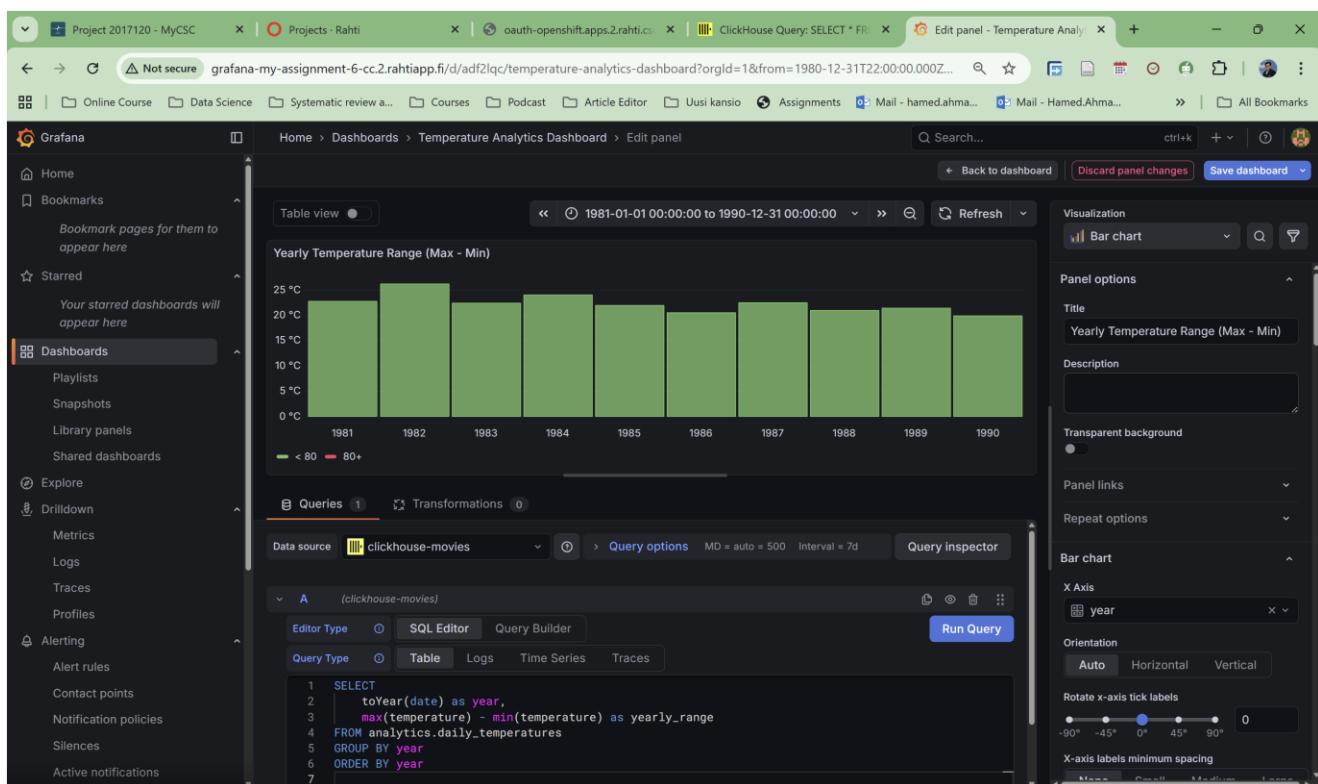
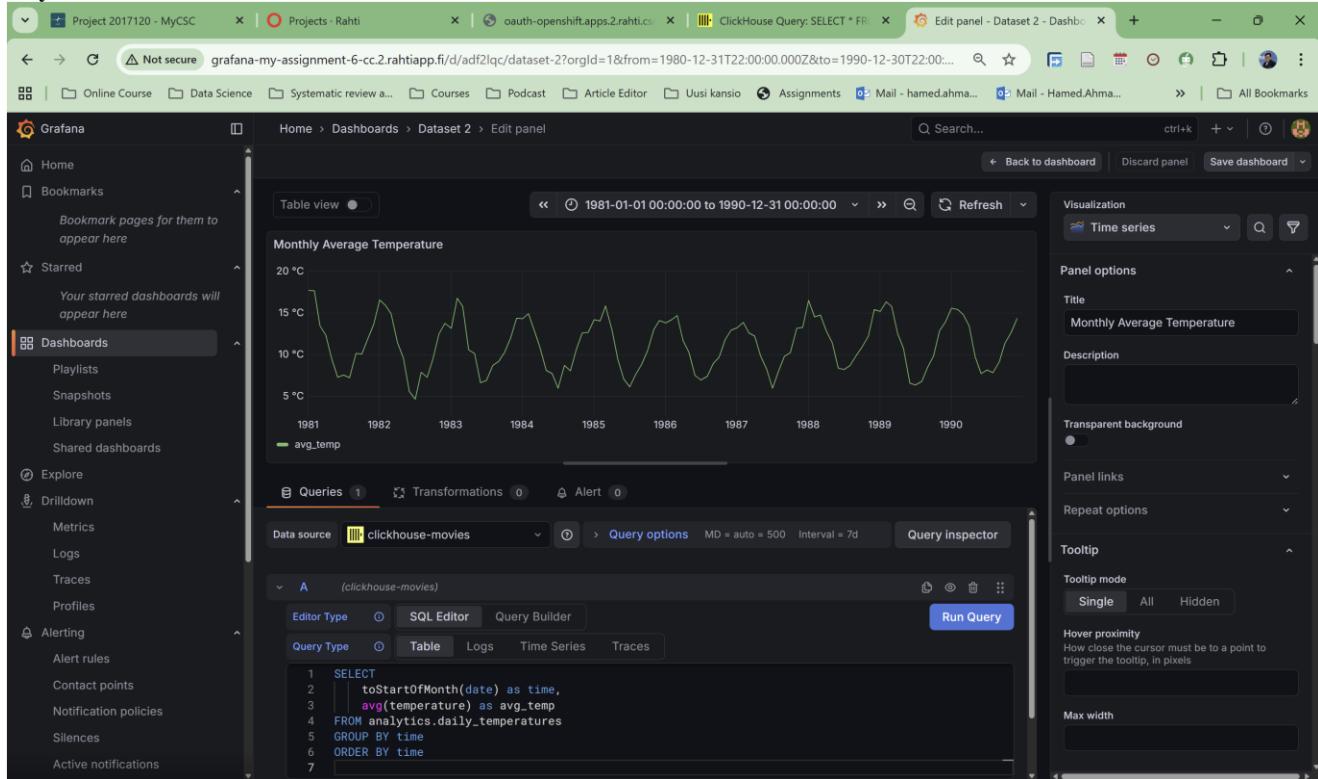
The underlying query in the SQL Editor is:

```
1 SELECT
2     date as time,
3     temperature
4 FROM analytics.daily_temperatures
5 ORDER BY date
```

Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania



Assignment 6: Cloud Native Visual Analytics

Course: Cloud Computing and Data Engineering

By: Hamed Ahmadiania

