Visualize Amazon S3 data using Amazon Athena and Amazon Managed Grafana

Tools are used in the project

AWS S3 **AWS ATHENA**

GRAFANA **LOACL WEB-SITE**

AWS S3 **NGROK**

share the website to other.

what is aws S3 and use of S3

Why we are using the s3 bucket in your Project ?

The main important using AWS S3 in your project .
> **Data Storage and Backup:** Storing all types of data such as documents, images, videos, application data, and backups.
>**Static Website Hosting:** Hosting static websites or web applications.
>**Content Delivery:**Storing and distributing static content.
> **Data Sharing and Collaboration:** Sharing data and collaborating with other users or systems.
>**Big Data and Analytics:**Storing large datasets for analysis.
>**Disaster Recovery and Archival:**Storing critical business data in S3 to ensure it can be recovered in case of a system failure or disaster.
>**Static Content:** Host the application's static web assets (like HTML, CSS, and JavaScript files) directly from S3.

Attache AWS S3 TO GLUE TO AWS ATHENA AND USE

Amazon S3 (Simple Storage Service)

> Storage: S3 is where your data resides. It's a highly scalable, secure, and durable object storage service offered by AWS.
> Data Lake: S3 serves as the foundation for your data lake, allowing you to store structured, semi-structured, and unstructured data in various formats like CSV, JSON, Parquet, etc

AWS Glue

> Data Catalog: Glue provides a fully managed data catalog service that catalogs and indexes your data stored in S3, making it searchable and queryable.
> ETL (Extract, Transform, Load): Glue offers ETL capabilities to prepare and transform your data before analysis. It automatically generates and executes ETL code to clean, enrich, and normalize your data.
> Schema Inference: Glue can automatically infer the schema of your data stored in S3, reducing manual effort.

Amazon Athena

> Query Service: Athena allows you to run SQL queries directly against the data stored in S3 without the need for setting up and managing servers.
> Serverless: It's a serverless service, meaning you don't need to provision or manage any infrastructure. You pay only for the queries you run.
> Interactive Analysis: Athena enables interactive, ad-hoc querying of data in S3, making it ideal for exploratory data analysis and business intelligence use cases.

Use of Attaching S3 to Athena with Glue

Query Data Directly: Athena can directly query the data stored in S3 using standard SQL syntax, without the need to load it into a separate database.

Schema Management: Glue provides a centralized data catalog where you can define and manage the schema of your data stored in S3. Athena can then use this catalog to understand the structure of your data and optimize query performance.

Cost-Effective Analysis: Since both Athena and Glue are serverless, you only pay for the queries you run and the resources consumed during data transformation, making it a cost-effective solution for data analysis.

The imaportant , attaching S3 to Athena with Glue enables seamless data analysis and querying capabilities on your data stored in S3, with Glue providing schema management and ETL capabilities, and Athena offering SQL-based querying on the data.

what is use of IAM USER & ROLE & POLICY IN PROJECT

- **Creating IAM (Identity and Access Management) users, roles, and policies is crucial when setting up a pipeline that involves transferring data from S3 to Athena and then visualizing it in Grafana. Here's why each component is essential**

IAM Role

Temporary Permissions: IAM roles are used to delegate permissions to entities like EC2 instances, Lambda functions, or even other AWS accounts. They provide temporary credentials with limited access, reducing the risk of unauthorized access.
Cross-Account Access: IAM roles enable cross-account access, allowing resources in one AWS account to access resources in another account securely.
Service Integration: IAM roles are often used for service-to-service communication, such as allowing Lambda functions to access S3 buckets or granting access to AWS Glue for data processing tasks.

IAM User

- **Granular Access Control:** By creating an IAM user, you can grant specific permissions only to the resources and actions required for the tasks the user needs to perform. This follows the principle of least privilege, enhancing security.
- **Audit Trail:** Actions performed by IAM users are logged, providing an audit trail for accountability and compliance purposes.
- **Separation of Concerns:** If multiple individuals or applications need access to AWS services, creating separate IAM users for each entity allows you to manage permissions independently.

IAM Policy

- **Permission Definition:** IAM policies define the permissions granted to users, roles, or groups. They specify what actions can be performed on which resources.
- **Resource Protection:** IAM policies help enforce security policies by restricting access to sensitive resources or actions. They allow you to control who can access, modify, or delete resources within your AWS environment.
- **Policy Management:** Policies can be attached to IAM users, roles, or groups, providing a centralized way to manage permissions across your AWS infrastructure.

Why Create IAM User, Role, and Policy for Your Pipeline:

1. **Security**: Creating IAM users, roles, and policies helps enforce the principle of least privilege, ensuring that only the necessary permissions are granted for each entity. This reduces the risk of unauthorized access or accidental data exposure.
2. **Compliance**: IAM controls help meet regulatory requirements and compliance standards by enforcing access controls and providing an audit trail of actions performed within your AWS environment.
3. **Manageability**: IAM allows you to manage access to AWS services centrally, making it easier to grant or revoke permissions as needed. This simplifies access management and reduces the risk of misconfigurations.
4. **Scalability**: As your pipeline grows and evolves, using IAM roles allows you to scale securely by granting permissions to new resources or services without compromising security.

Note: Important creating IAM users, roles, and policies is essential for maintaining security, compliance, manageability, and scalability in your AWS environment, especially when setting up pipelines involving data transfer and visualization across multiple services like S3, Athena, and Grafana.

Afert creating the IAM user, role , policys we need add some services and policys in IAM to see the data source in the grafana

The IAM policy grants Grafana access to Athena data stored in S3. Here's the policy and why it's needed:

IAM Policy for Grafana Access to Athena Data

The use of the json based on json code we give permisoon to access to Athena to the Grafana

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "athena:GetQueryExecution",
        "athena:GetQueryResults",
        "athena:StartQueryExecution"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::your-athena-query-results-bucket/*"
    }
```

IAM Policy: It's like a rulebook that tells AWS who can do what with which resources.
JSON Format: JSON (JavaScript Object Notation) is a way to write data in a structured format that computers can understand. In this case, it's used to write the IAM policy.
Version: Specifies the version of the IAM policy language being used. It tells AWS how to interpret the policy.
Statement: This is where you define the permissions. Each statement represents a set of permissions.
Effect: It can be "Allow" or "Deny". "Allow" means granting permissions, and "Deny" means blocking permissions.
Action: Lists the specific actions that are allowed or denied. Actions are operations that can be performed on AWS resources.
Resource: Specifies the AWS resources to which the actions apply. It defines where the actions can be performed.

Explanation of the Provided Policy

- **Allow Athena Actions:** This policy allows certain actions (GetQueryExecution, GetQueryResults, StartQueryExecution) on Athena. These actions are related to querying data in Athena.
- **Resource for Athena Actions:** The Resource field is set to *, which means all resources. So, this policy allows these actions on all Athena resources.
- **Allow S3 GetObject:** This policy also allows the GetObject action on S3. This action allows fetching objects (files) from an S3 bucket.
- **Resource for S3 GetObject:** The Resource field specifies the ARN (Amazon Resource Name) of an S3 bucket (arn:aws:s3:::your-athena-query-results-bucket/*). This means it allows fetching objects from a specific S3 bucket. You need to replace your-athena-query-results-bucket with the actual name of your S3 bucket where Athena query results are stored.

In short, this IAM policy grants permissions to fetch query results from Athena and S3, allowing Grafana to access and visualize the data.

how to install Grafana and use of Grafana .

Installing Grafana

> **Download**: Go to the Grafana website and download Grafana for your computer (Windows, Mac, or Linux).
> **Install**: Follow the instructions to install Grafana. It's usually just a matter of clicking "Next" a few times.
> **Start**: After installation, start Grafana. On Windows, it might be a program you can open. On Mac or Linux, you might need to run a command.

Using Grafana

Grafana helps you see and understand your data. You can:
> **See Data**: Grafana makes it easy to see your data by creating graphs and charts.
> **Connect Data**: You can connect Grafana to different places where your data is stored, like databases or online services.
> **Make Dashboards**: Dashboards are like custom screens where you can put all your graphs and charts in one place.
> **Get Alerts**: Grafana can alert you if something goes wrong with your data, like if a server is getting too hot or a website is too slow.

Why Use Grafana

Easy Visualization: Grafana makes it easy to turn your data into graphs and charts that are easy to understand.
Monitor Everything: You can use Grafana to monitor anything you want, from your computer's performance to your website's traffic.
Stay Informed: Grafana alerts you when something needs your attention, so you can fix problems quickly.
Note: Grafana helps you see, understand, and stay on top of your data, making it a valuable tool for many different tasks.

how Add an Athena data source in Grafana

1. **Open Grafana**: Open your Grafana instance in your web browser.
2. **Log In**: Log in to Grafana with your username and password if required.
3. **Go to Data Sources**: Click on the "Configuration" gear icon on the left sidebar, then select "Data Sources".
4. **Add Data Source**: Click on the "Add data source" button.
5. **Select Athena**: In the list of available data sources, find and click on "Athena".
6. **Configure Athena Data Source**:
   - **Name**: Enter a name for your data source, like "Athena".
   - **Auth Provider**: Choose "AWS" since you'll be using AWS authentication.
   - **AWS Region**: Select the region where your Athena queries are executed.(LOCATION)
   - **Access Key ID**: Enter your AWS Access Key ID.(from iam USER)
   - **Secret Access Key**: Enter your AWS Secret Access Key.(from iam USER)
   - **Session Token (Optional)**: Enter your AWS Session Token if using temporary credentials.
   - **Workgroup (Optional)**: Enter the name of the Athena workgroup if applicable.(FORM ATHENA)
7. **Test Connection**: Optionally, click the "Save & Test" button to test the connection to your Athena data source. This ensures that Grafana can connect to Athena using the provided credentials.
8. **Save Data Source**: Once the connection test is successful, click the "Save & Test" button again to save the data source configuration.
9. **Verify**: After saving, you should see your Athena data source listed in the data sources page.
10. **Explore and Visualize**: Now that you've added your Athena data source, you can start creating dashboards and visualizing your Athena query results in Grafana.

That's it! You've successfully added an Athena data source in Grafana. You can now use Grafana to explore, analyze, and visualize your data from Athena.

The data source in Grafana acts as a bridge between your data and Grafana's visualization tools. Here's how it works and why it's essential:

Use of Data Sources in Grafana:

1. **Connecting to Data**: Data sources allow Grafana to connect to various types of data repositories, databases, and monitoring systems where your data resides.
2. **Querying Data**: Once connected, Grafana can query the data source to retrieve information based on your configured queries.
3. **Visualizing Data**: Grafana then uses the retrieved data to create visualizations such as graphs, charts, and tables, allowing you to monitor and analyze your data effectively.
4. **Creating Dashboards**: Data sources are essential for creating dashboards in Grafana. Dashboards are customizable screens where you can arrange and display your visualizations, providing insights into your data.
5. **Alerting**: Grafana can also use data sources to set up alerts based on specific conditions or thresholds. For example, you can configure Grafana to send notifications if CPU usage exceeds a certain limit.

Why Data Sources are Important

- **Versatility:** Data sources in Grafana support a wide range of data types and sources, including databases (like MySQL, PostgreSQL), time-series databases (like Prometheus, InfluxDB), cloud services (like AWS CloudWatch), and more. This versatility allows you to monitor and analyze data from different sources in one place.
- **Consolidation:** Grafana's ability to connect to multiple data sources allows you to consolidate data from different systems into a single dashboard, providing a holistic view of your infrastructure or application performance.
- **Customization:** Each data source in Grafana can have its own configuration settings, allowing you to tailor data retrieval and visualization options to specific requirements.
- **Real-time Monitoring:** By connecting Grafana to real-time data sources, such as time-series databases or monitoring systems, you can monitor and react to changes in your data in real-time, helping to ensure the reliability and performance of your systems.

Note: data sources are a fundamental component of Grafana, enabling you to connect, query, and visualize data from various sources, providing valuable insights into your infrastructure, applications, and business metrics.

how to creat the dashboard in Grafana

Creating a dashboard in Grafana allows you to visualize and monitor your data in a customizable and interactive way. Here's how to create a dashboard and its use

Steps to Create a Dashboard:
1.Log in to Grafana: Open your Grafana instance in a web browser and log in with your credentials.
2.Access Dashboards: In the Grafana interface, click on the "Dashboards" icon in the left sidebar.
3.Create a New Dashboard:
Click on the "Manage Dashboards" button.
Click on the "+ Add new" button to create a new dashboard.
4. Add Panels:
Click on the "Add panel" button in the top right corner of the dashboard.
Choose the type of visualization you want to add (e.g., Graph, Singlestat, Table).
Configure the panel settings, including data source, query, and visualization options.
5. Customize Dashboard Layout:
Drag and drop panels to rearrange them on the dashboard.
Resize panels by dragging the edges to adjust their size.
Use the panel options menu (three dots in the panel header) to edit, duplicate, or delete panels.
6. Save Dashboard:
Once you've configured your dashboard, click on the "Save" button in the top right corner.
Enter a name for your dashboard and click "Save".
7. Access and Share Dashboard:
Your dashboard is now saved and accessible from the dashboard list.
You can share the dashboard with others by copying the dashboard URL or embedding it in a webpage.

Use of Dashboards

1. **Visualize Data**: Dashboards allow you to visualize your data using various types of charts, graphs, and tables. You can monitor metrics such as system performance, application health, or business KPIs.
2. **Monitor in Real-Time**: Dashboards provide real-time insights into your data, allowing you to monitor changes and trends as they happen.
3. **Identify Anomalies**: By setting up alerts on your dashboard, you can be notified when certain conditions or thresholds are met, helping you identify and address anomalies in your data.
4. **Track Key Metrics**: Dashboards help you track key metrics and performance indicators, enabling you to make informed decisions and take timely actions to optimize your systems or processes.
5. **Collaborate and Share**: Dashboards can be shared with team members or stakeholders, facilitating collaboration and communication around the data.

Note: dashboards in Grafana are powerful tools for data visualization, monitoring, and analysis. They provide a centralized and customizable view of your data, enabling you to gain insights, detect issues, and make data-driven decisions effectively

how to share the dasnboard into lcoal website

To share a Grafana dashboard on a local website, you can embed it using an iframe. Here's a step-by-step guide

Steps to Share a Grafana Dashboard on a Local Website

Generate the Embed URL:
Open the Dashboard: Log in to Grafana and open the dashboard you want to share.
Copy the URL: Click on the "Share" button (usually a paper plane icon or share icon) in the top-right corner of the dashboard.
Embed Tab: Select the "Embed" tab in the share dialog.
Copy Embed Code: You will see an iframe HTML code snippet. Copy this code.
Set Permissions:
Public Access: Ensure the dashboard is accessible to users who will view it on your local website. This might involve setting up public access or sharing it with specific users/roles.
Anonymous Access: If you want to allow anonymous access, you might need to adjust your Grafana settings to enable it.
Embed the Dashboard in Your Website:
HTML Page: Open the HTML file of your local website where you want to embed the dashboard.
Insert Iframe: Paste the copied iframe code into the HTML file at the desired location.
Adjust Size: Modify the width and height attributes of the iframe tag to fit your layout.

Example:
Here's how your HTML file might look:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Grafana Dashboard</title>
</head>
<body>
   <h1>My Grafana Dashboard</h1>
   <iframe src="http://your-grafana-instance/dashboard-solo/db/your-dashboard?panelId=1&fullscreen&orgId=1" width="100%"
height="800"></iframe>
</body>
</html>
```

Explanation:
src: Replace http://your-grafana-instance/dashboard-solo/db/your-dashboard?panelId=1&fullscreen&orgId=1 with the actual URL of your Grafana dashboard.
width and height: Adjust these values to fit the embedded dashboard within your website layout.
Ensure Access:
Network Access: Ensure that your Grafana instance is accessible from the machine where you are running your local web server.
Cross-Origin Requests: Depending on your setup, you might need to handle CORS (Cross-Origin Resource Sharing) issues if your Grafana instance is running on a different domain or port.
By following these steps, you can successfully embed a Grafana dashboard into your local website, allowing you to share real-time data visualizations easily.

Hosting a static website on AWS S3

Create an S3 Bucket:
Open the S3 Console: Go to the Amazon S3 console.
Create Bucket: Click the "Create bucket" button.
Bucket Name: Enter a unique bucket name (e.g., my-website-bucket).
Region: Choose the region where you want to create the bucket.
Settings: Leave the default settings or adjust them as needed.
Create: Click "Create bucket".
Upload Your Website Files:
Select Bucket: Click on the bucket name you just created.
Upload Files: Click the "Upload" button and select all your website files (HTML, CSS, JS, images, etc.).
Upload: Click "Upload" to upload the files to your S3 bucket.
Configure Bucket for Static Website Hosting:
Properties Tab: Go to the "Properties" tab of your S3 bucket.
Static Website Hosting: Scroll down and find the "Static website hosting" section. Click "Edit".
Enable: Select "Enable" and choose "Host a static website".
Index Document: Enter the name of your index file (e.g., index.html).
Error Document: Optionally, enter the name of your error document (e.g., error.html).
Save Changes: Click "Save changes".
Set Bucket Policy for Public Access:
Permissions Tab: Go to the "Permissions" tab of your S3 bucket.
Bucket Policy: Scroll down to the "Bucket Policy" section and click "Edit".
Policy JSON: Add the following JSON policy to make your bucket publicly accessible:
Access Your Website:
Bucket URL: Go back to the "Properties" tab and find the "Static website hosting" section. You'll see the URL of your website (e.g., http://my-website-bucket.s3-website-us-east-1.amazonaws.com).
Open URL: Open this URL in your web browser to see your hosted static website.
these steps, you can host your static website on AWS S3. This approach provides a scalable, reliable, and cost-effective way to serve static web content. Your website will be publicly accessible via the S3 bucket URL, and you can update the content by simply uploading new files to the bucket.

Hosting a static website on AWS S3 has several advantages and use cases

Uses and Benefits:
Cost-Effective:
Low Cost: AWS S3 charges are based on storage and bandwidth usage, making it a very cost-effective solution for hosting static websites.
No Server Management: You don't need to manage any servers or infrastructure, reducing maintenance costs and complexity.
Scalability:
Automatic Scaling: S3 automatically scales to handle large amounts of traffic without any intervention, making it suitable for high-traffic websites.
High Availability and Durability:
Reliability: S3 is designed for 99.99% availability and 99.999999999% durability of objects over a given year.
Global Infrastructure: AWS's global infrastructure ensures that your website can be accessed quickly from anywhere in the world.
Simplicity:
Easy Setup: Setting up a static website on S3 is straightforward, requiring minimal configuration.
Static Content Hosting: Perfect for static content like HTML, CSS, JavaScript, and images.
Integration with Other AWS Services:
CloudFront: You can integrate S3 with AWS CloudFront, a content delivery network (CDN), to further reduce latency and improve load times.
Route 53: Use AWS Route 53 for custom domain names and DNS management.
Security:
Fine-Grained Access Control: S3 provides robust security features, allowing you to control who can access your website and how.
SSL/TLS: Use HTTPS to secure your website by integrating with CloudFront.

Specific Use Cases:
Personal or Portfolio Websites:
Individuals can host their personal blogs, portfolios, or resume sites cost-effectively.
Documentation Sites:
Host static documentation sites for software projects, making it easy to share with users and developers.
Marketing or Landing Pages:
Companies can deploy landing pages or marketing websites quickly without the need for backend infrastructure.
Simple Web Applications:
For single-page applications (SPAs) built with frameworks like React, Angular, or Vue.js, S3 can serve the entire frontend.
Temporary or Event Websites:
Deploy temporary websites for events, campaigns, or promotions that require high availability and scalability without long-term hosting commitments.
Note: Hosting a static website on AWS S3 is useful because it is cost-effective, scalable, highly available, and easy to set up. It is ideal for serving static content and can integrate with other AWS services to enhance performance and security. This makes it an excellent choice for personal projects, business websites, documentation, and more.

what is the NGROK use of NGROK

Ngrok is a tool that provides secure tunnels to your localhost, making it accessible from the internet. It is particularly useful for developers who need to test and demonstrate web applications or APIs that are running on their local machine.

What is Ngrok?
Ngrok: A tool that exposes local servers behind NATs (Network Address Translators) and firewalls to the public internet over secure tunnels.

Uses of Ngrok:
Local Development and Testing:
Public URLs for Localhost: Ngrok provides a public URL that you can share with others to access your locally running web server.
Webhooks: Many third-party services require public URLs to send webhooks. Ngrok makes it easy to test these webhooks locally by providing a public endpoint that forwards requests to your local machine.
Testing Mobile Apps:
Access from Devices: Test web applications or APIs running on your local machine from mobile devices or remote systems without deploying the application.
Secure Tunnels:
HTTPS for Localhost: Ngrok provides HTTPS URLs, which is useful for testing applications that require secure connections (e.g., OAuth, secure APIs).
Authentication and Access Control: Ngrok includes features for basic authentication, adding another layer of security to your exposed endpoints.
Collaboration and Demos:
Share Local Work: Easily share your local development environment with teammates or clients for collaboration or demonstrations.
Temporary Public Access: Use Ngrok to temporarily expose a local application for a live demo or to get feedback without deploying it to a public server.
API Inspection and Debugging:
Traffic Inspection: Ngrok includes a web interface that allows you to inspect traffic passing through your tunnel. This is useful for debugging and monitoring HTTP requests and responses.
Replay Requests: You can replay requests from the Ngrok dashboard to test different scenarios without triggering the original source.

How to Use Ngrok

Download and Install Ngrok:
Go to the Ngrok website and download the appropriate version for your operating system.
Extract the downloaded file and place the ngrok executable in a directory that is in your system's PATH.
Start a Tunnel:
Open a terminal or command prompt.
Start a tunnel by specifying the port your local server is running on. For example, if your local server is running on port 8000: ngrok http 8000
Get Public URL:
After running the command, Ngrok will provide you with a public URL (e.g., http://randomstring.ngrok.io and https://randomstring.ngrok.io).
Use this URL to access your local application from anywhere.
Inspect Traffic:
Ngrok provides a web interface at http://localhost:4040 where you can inspect and replay the traffic going through your tunnel.
Note:Ngrok is a powerful tool for exposing local servers to the internet, facilitating development, testing, and collaboration. It provides secure tunnels, traffic inspection, and easy access to your local applications from anywhere.

final step

When you open the S3 link on another system for the first time, you might need to click on "Visit site" and refresh the page to see the Grafana dashboard.
Ensure that the Grafana dashboard is configured to update every minute to reflect the latest data.