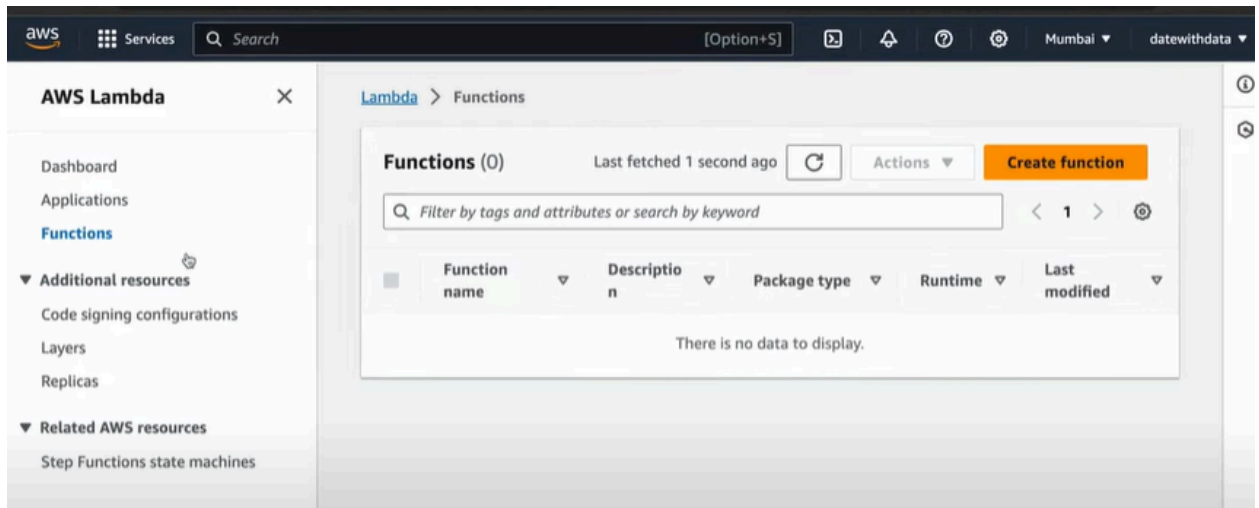


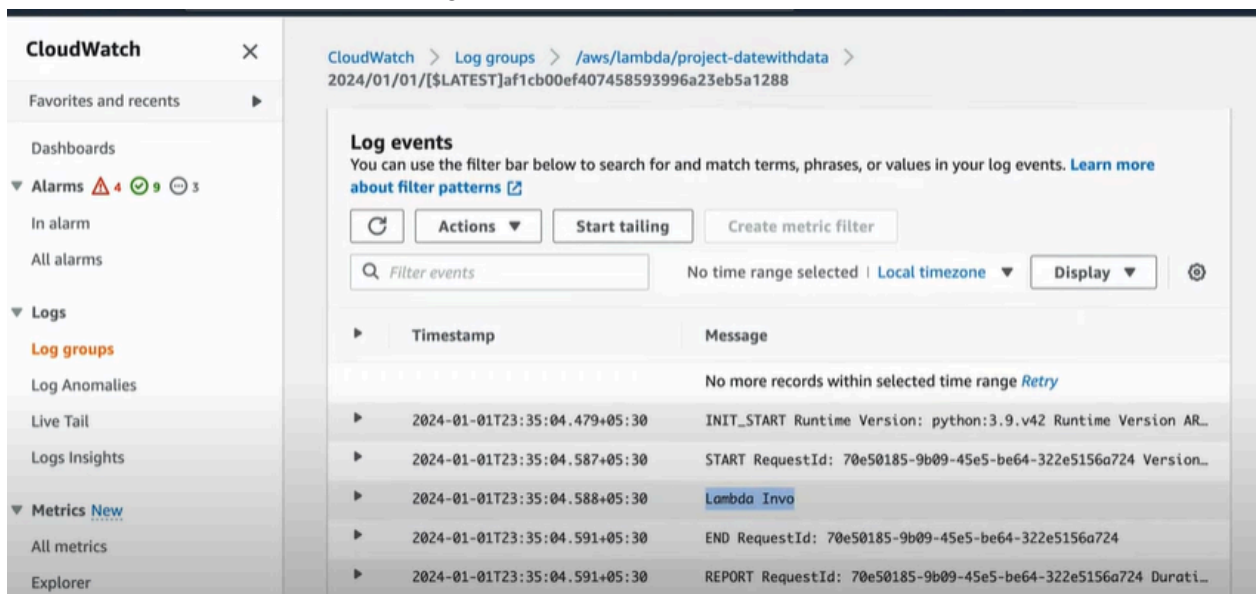
Lambda Function

1. Lets create a lambda function..that takes data from dynamoDB and stores it in S3 Bucket
2. Lets go to lambda and click on create function



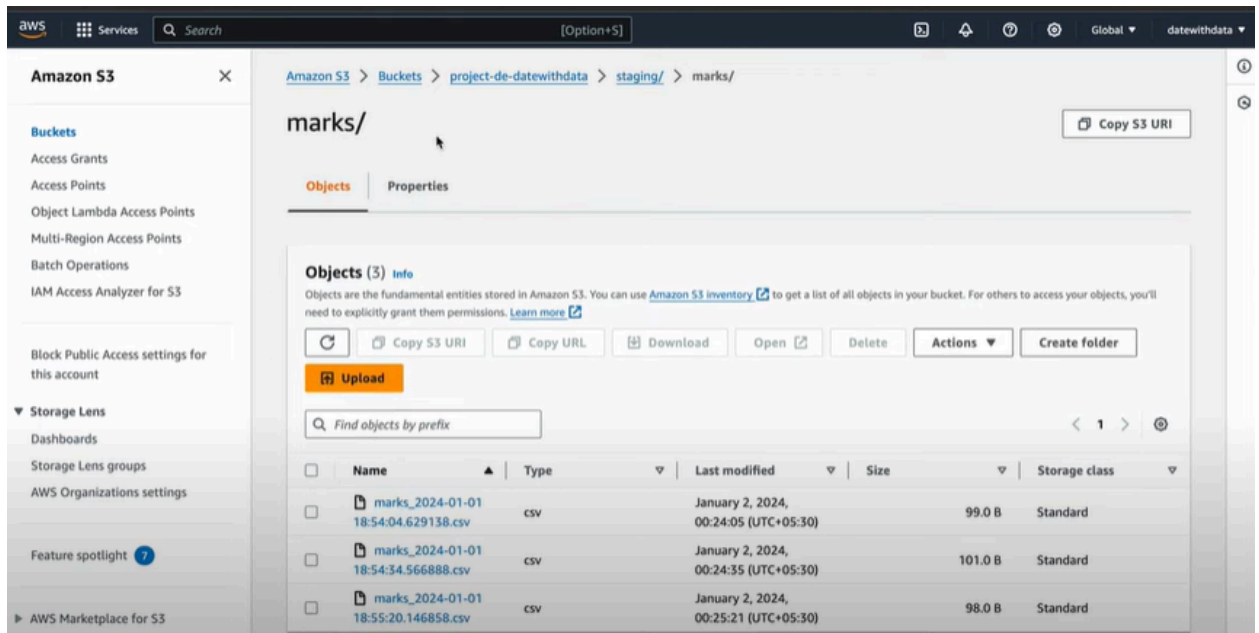
and we create a function with python as runtime

3. Lets setup a dynamoDB trigger
4. Now we need to give access to dynamoDB and S3 for our lambda function...using policies
5. We need to integrate cloud watch with our lambda ..to check the logs of the trigger
6. Next we'll add sample data in our dynamoDB and it triggers the lamda function...which we can see thru the cloudwatch logs



7. Now the data will be reflected in our S3 buckets

8. And we can see here..the data which we created in our marks table in dynamoDB is reflected in the marksS3 bucket



9. Now we have successfully created a lambda function that takes data from dynamoDB streams and load them in S3 buckets

AWS Glue Job

1. Now we'll create a Gluejob which takes data from s3buckets joins and save in data warehouse
2. Please watch the ETL code on youtube
3. Next we'll pste our code in Glue Script and provide all the required Job Details
4. After successfully running the GLue job...we can also see the output logs

CloudWatch

CloudWatch > Log groups > /aws-glue/jobs/output

/aws-glue/jobs/output

Actions View in Logs Insights Start tailing Search log group

Log group details

Log streams Tags Anomaly detection - new Metric filters Subscription filters

Log streams (2) Delete Create log stream Search all log streams

jr_ebd38f9d0c249418f068c4a! 1 match Exact match Show expired Info 1

Log stream	Last event time
jr_ebd38f9d0c249418f068c4a5ffd6c39ab206d...	2024-01-05 00:07:39 (UTC+05:30)

5.

Timestamp	Message
No more records within selected time range Retry	
2024-01-05T00:07:39.469+05:30	Read Mark Data
2024-01-05T00:07:39.658+05:30	Read Student Data
2024-01-05T00:07:39.861+05:30	Read DW Data
2024-01-05T00:08:01.683+05:30	Join Successful
2024-01-05T00:08:02.434+05:30	Union Successful
2024-01-05T00:08:02.489+05:30	Delete existing files
2024-01-05T00:08:05.537+05:30	Save the data
No more records within selected time range <i>Auto retry paused.</i> Resume	

6. Next we schedule the job run..to run the ETL job on regular intervals
7. We also can use version control in AWS glue
8. Now we have successfully created a ETL job..that takes data from s3 buckets and saves in data warehouse

ATHENA

- Run Glue Crawler
- Create tables and database
- Find the class topper

AWS Athena

1. Here we'll run Glue crawler...which helps us to create tables and databases
2. Then we can use AWS Athena...to query the data created by Glue crawler
3. Adding a Glue Crawler

The screenshot shows the AWS Glue console interface for adding a new crawler. The left sidebar contains the navigation menu with 'Crawlers' selected. The main area is titled 'Set crawler properties' and shows a five-step process. Step 1, 'Set crawler properties', is the active step. The 'Crawler details' section includes a 'Name' field with a placeholder 'Enter a unique crawler name', a 'Description - optional' field with a placeholder 'Enter a description', and a 'Tags - optional' section. The 'Next' button is highlighted in orange.

4. Next we'll add the data source...which is our warehouse

The screenshot shows the 'Choose S3 path' dialog in the AWS Glue console. It displays the path 'project-de-datewithdata' under 'S3 buckets'. A table titled 'Objects (1/2)' lists two folders: 'staging/' and 'warehouse/'. The 'warehouse/' folder is selected. The 'Choose' button is highlighted in orange.

5. Next we have to provide the IAM role

6. Next we have create a db and assign a db

The screenshot shows the 'Set output and scheduling' step in the AWS Glue console. On the left, a sidebar lists five steps: Step 1 (Set crawler properties), Step 2 (Choose data sources and classifiers), Step 3 (Configure security settings), Step 4 (Set output and scheduling), and Step 5 (Review and create). Step 4 is currently selected. The main panel is titled 'Set output and scheduling' and contains an 'Output configuration' section. This section includes a 'Target database' dropdown menu with 'dw' selected, a 'Clear selection' button, and an 'Add database' button with a link icon. Below this is a 'Table name prefix - optional' text input field with a placeholder 'Type a prefix added to table names'. Further down is a 'Maximum table threshold - optional' section with a text input field and a placeholder 'Type a number greater than 0'. At the bottom of the configuration section is a link for 'Advanced options'.

An AWS Glue crawler is an automated tool that discovers and analyzes data stored in various data sources. It acts like a librarian in a giant data warehouse, automatically sorting and cataloging your information. Here's how it works with an example:

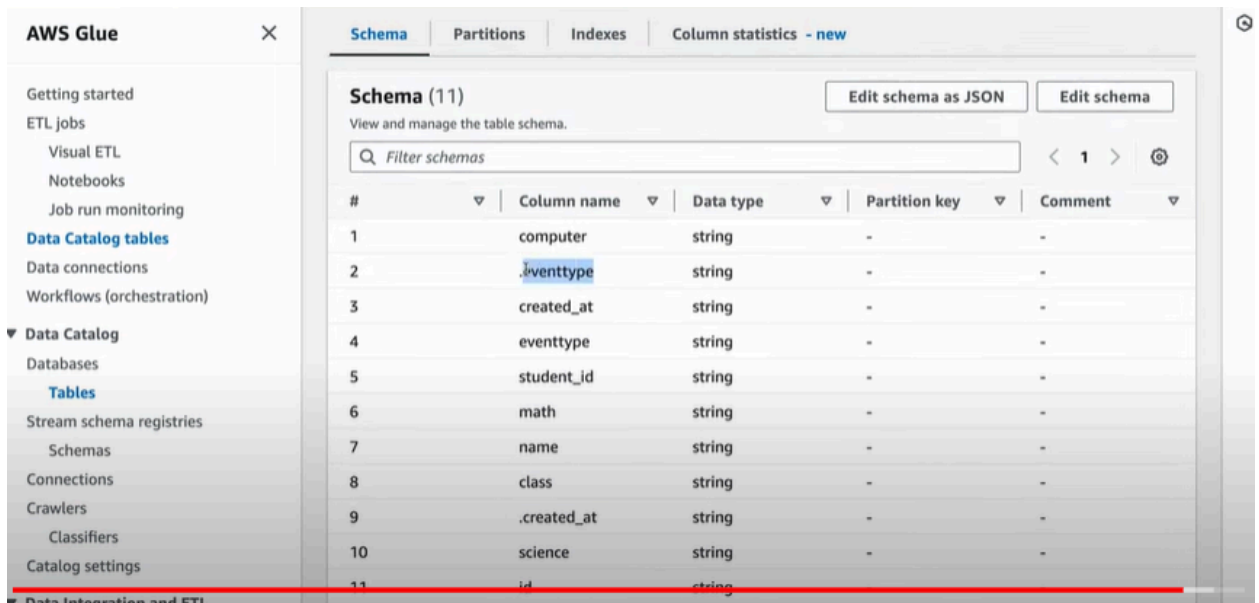
Imagine you have a bunch of CSV files containing sales data in an Amazon S3 bucket (think of it as a cloud storage space). The data is scattered across different folders, and there might be new files added every day.

An AWS Glue crawler can be configured to scan this S3 bucket. Here's what it typically does:

1. **Connects and Discovers:** The crawler connects to your S3 bucket and starts discovering the files.
2. **Classifies and Schemas:** It uses built-in classifiers (like a predefined format identifier) to understand the data format (CSV in this case). It then analyzes a sample of the data to figure out the structure, like which column represents "customer name" and which one represents "sales amount."
3. **Catalogs the Data:** Based on its findings, the crawler creates a schema (blueprint) describing your data and creates an entry in the AWS Glue Data Catalog. This catalog acts as an index, storing information about your data's location, format, and structure.
4. **Updates Regularly (Optional):** You can schedule the crawler to run periodically. This way, whenever new data files are added to the S3 bucket, the crawler automatically crawls them, updates the schema if needed, and keeps your Data Catalog synchronized.

- 7.
8. Now we'll create the crawler

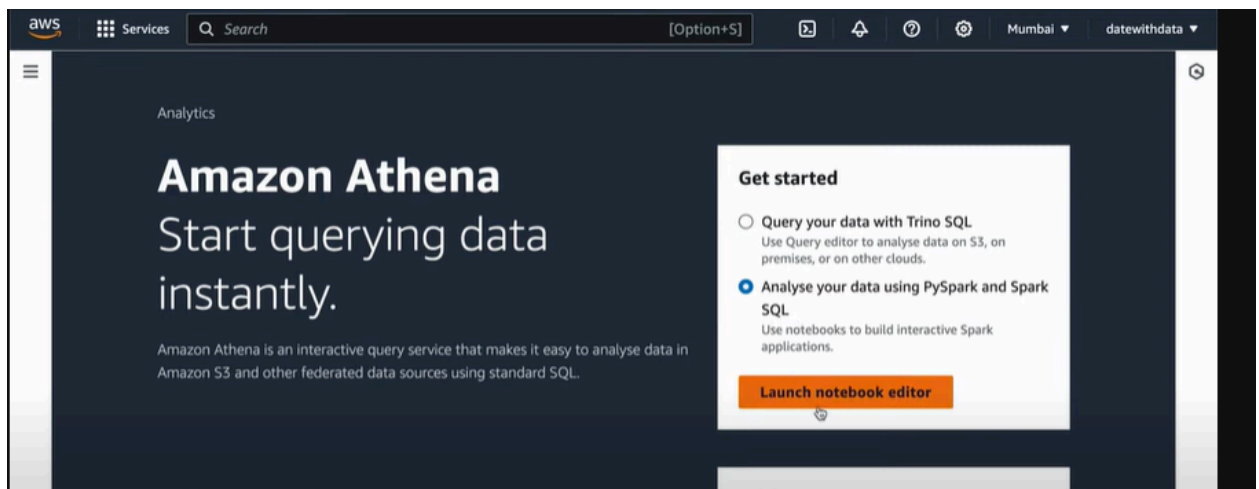
9. Once our table are created..we can go to athena and query our table



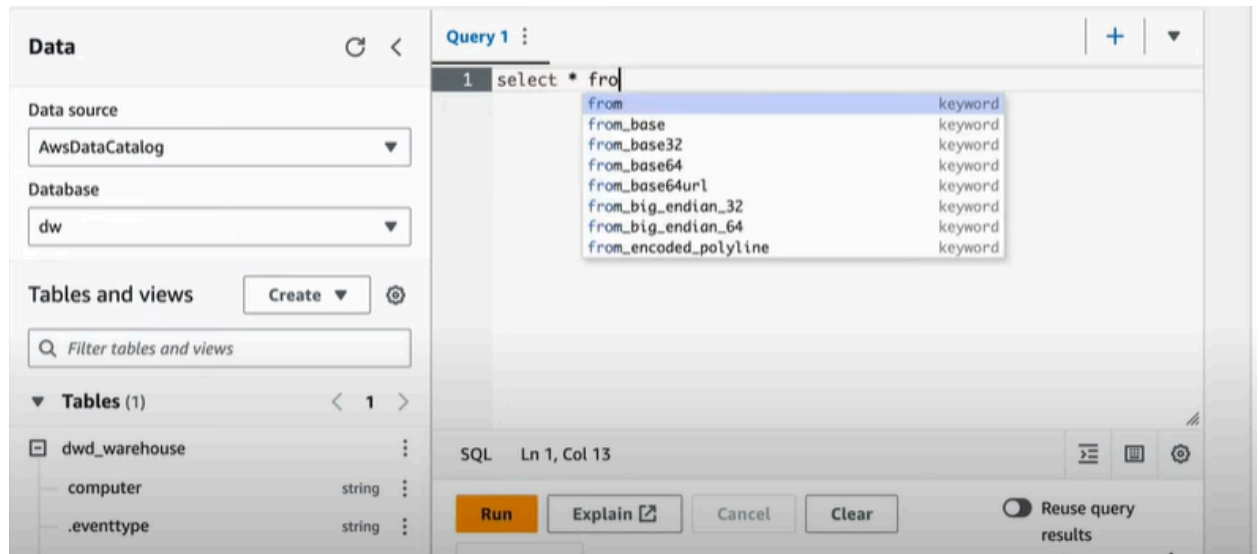
The screenshot shows the AWS Glue console interface. On the left is a navigation sidebar with options like 'Getting started', 'ETL jobs', 'Data Catalog tables', and 'Data Catalog'. The main panel is titled 'Schema (11)' and contains a table of schema columns. The table has columns for '#', 'Column name', 'Data type', 'Partition key', and 'Comment'. The columns listed are: computer (string), eventtype (string), created_at (string), eventtype (string), student_id (string), math (string), name (string), class (string), .created_at (string), science (string), and id (string). The 'eventtype' column is highlighted in blue. There are buttons for 'Edit schema as JSON' and 'Edit schema' at the top right of the schema view.

#	Column name	Data type	Partition key	Comment
1	computer	string	-	-
2	eventtype	string	-	-
3	created_at	string	-	-
4	eventtype	string	-	-
5	student_id	string	-	-
6	math	string	-	-
7	name	string	-	-
8	class	string	-	-
9	.created_at	string	-	-
10	science	string	-	-
11	id	string	-	-

10. Athena



11.



here by default we got AWS data catalog as source ..and now we can query(using SQL) our data from dw database

