Athena
=======

HDFS file - data
+
Schema - metadata

= Tabular view

Data will be kept in S3
+
Build Schema on top of it

= Tabular view

we can query it like normal sql table

we can query s3 data using sql style syntax.

EMR - we create a cluster of machines.

for Athena we do not have to create a cluster.

It is serverless.

No infrastructure is required.

In case of EMR we were paying based on nodes..

for each node we have cost associated per hour..

we will be charged based on amount of data scanned.

in S3 we can store our data in csv, json, parquet, orc ..

perfect for infrequent queries..

we can only focus on our actual problem statement, and we do not have to worry about managing the infrastructure.

pay for queries(based on amount of data scanned)

$5 per TB scanned..

100 mb

100 * 5

1000 * 1000

5 kb (atleast for 10 mb)

charged for minimum 10 mb per query.

there are no charges for DDL.

if the query fails we are not charged.

what if you cancel the query.

if we cancel the query it will charge us for the data that is already scanned.

problem 1: I want to find the sum of scores for each candidate.

for this we have to group it based on each student and then apply sum on scores.

problem 2: I want to calculate the avg marks scored in maths.

when the data is stored in csv format(non partitioned)

we have to go through full data scan which will incur more charges.

our goal should be to organize the data in a way that we do not have to scan all data..

we want to minimize the data scanned.

Athena session - 2
==========

we were scanning the entire data.

How to minimize the data scanning
==========

1. skip some of the columns from being scanned - column pruning

when we use row based file formats then we cannot skip some of the columns. we have to read all the columns.

column pruning is not possible when we go with row based file formats.

column based file formats - parquet, orc ..

2. skip some of the rows - partition

so that we have partitions of data and we can then skip some of the partitions..

partition pruning.

in our example we can have partitioning on subject so that we can only scan the math folder.

I will write a simple spark program to take csv as input

and give partitioned (subject) data in parquet format as output.

when we are saying

select year from trendytech.students_partitioned

then we see that we are just scanning 3 kb

first of all we are skipping all the columns.

parquet will be internally using dictionary encoding..

select subject, avg(score) as average_marks from trendytech.students_partitioned group by subject having subject = 'Math'

so in this case we are doing 2 levels of pruning

column skipping - parquet

partition skipping - partitioned the data..

3. use compression techniques like snappy, gzip..

in this session we created the schema manually..

in the next session we will see how to infer the schema of data using crawler..

Athena session - 3
==========

Last time we created the schema manually..

is there a automated way to infer the schema..

AWS Glue
=========

Glue acts as a metastore - the metadata is stored in
glue catalog (metastore)

Glue provides you a crawler as well which can crawl your data and infer schema from that..

AWS Glue is one service

S3 is another service

we have to give permission to AWS Glue to access S3

AWS Athena along with AWS Glue

Athena gives you a panel to query to your table

data of table - S3

Metadata is stored in - Glue
(we have inferred the schema using glue).

Athena
=======

how to create a normal table manually on csv data residing in s3

how to create a partitioned table on parquet file to get optimization so that we scan less data.

how to infer the schema using glue.

glue catalog holds the metadata - Hive Metastore

S3 holds the actual data.. - HDFS