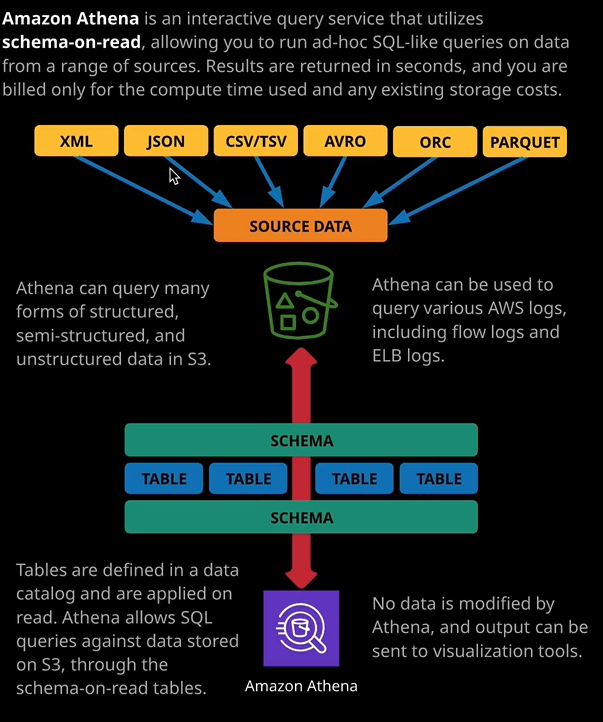
**With Athena, you can query huge data sets, which are stored on S3 with no preparation and only pay for the data volume that's queried by the service.** Now, if you're using traditional database engines, you need to create the table structure and the data structure in advance and this is known as the schema and once you've got the schema you need to put data inside those tables. So a traditional database management system can perform queries against it.



Now two problems that are associated with this way of doing things is that the database schema cannot easily change after the fact and if you're using a traditional database platform, then you've got to maintain the running costs of that database, even if you're not regularly performing queries on it.   
So they're two problems that Athena aims to address. Now, Athena handles things differently. With **Athena, your data lives on S3, and it's never changed or reformatted. It stays on S3 in its original format. Athena supports a wide range of data formats such as XML, JSON, CSV, and TSV, AVRO,** which I'm probably pronouncing it correctly ORC and PARQUET. Now, **since S3 is generally the best place to store large data sets within AWS anyway you're not double spending and having to maintain two copies of huge amounts of data.** Think about this for a second. How many AWS services have you learned about which store data in S3? Things like FlowLogs, CloudTrail.   
Even the logs of Elastic Load Balancers can be stored inside S3 and that's not even mentioning if you have to do any large scale data analytics using public data sets that are probably already stored in S3. **So the first main benefit of Athena is that your data is already probably going to be on S3.**   
The **second main benefit is that Athena uses a system known as schema on read.** **Now, what schema on read means is that you can define your schema in advance.** So this is what's indicated on this part of the diagram. You essentially define tables, attributes, and the overall structure of exactly what you want the data to look like. I'm being very specific with the words that I use. **The schema as it relates to Athena, not only defines what the structure is like on the source data store, so on S3, but it also defines exactly how you want it to look**. Unlike traditional relational database systems, though the schema is not persistent thing, you're not actually creating anything that exists constantly throughout the lifecycle of the product. The schema is only used when you use Athena to read data. So when Athena is performing queries you can imagine it that it's looking at the data through the lens of the schema. So the schema is only used when the data is being read through it and presented to you. **So essentially, you're creating a virtual table and you're reading your data through that virtual table, which applies the structure and this massively reduces the admin overhead needed to query in these large data sets because you don't need to perform any data manipulation when you initially added to Athena.** You just need to point Athena add the data and then run your queries against the schema and that schema was used while the query operate so that's a super powerful feature of the product.

Now, when you query, **Athena uses a serverless data processing engine, so it gathers the data that you request.** It reads it through the schema, and it presents it to you. You only pay for the amount of data queried for each operation as well as any existing data costs for storing the source data on S3 but just to reiterate, **it's likely that your source data already exists and Athena doesn't make any modifications to that source data.**

Now, I think the easiest way to demonstrate just how awesome Athena is, is to do a quick demonstration and I already have some good data to use for this. So in an earlier lesson, I enabled CloudTrail within this S3 account just to demonstrate how the product worked and I set my CloudTrail logs to be stored inside an S3 bucket called ac-cloudtrail234234 and if I go inside this bucket just demonstrate so AWS logs and then my account ID and then CloudTrail and then I can pick a particular region. So A.P. Southeast 2 which is the Sydney region going to 2019 08 which is this month and then I've got folders for each day that CloudTrail has been enabled. So CloudTrail stores this data in S3 and it's split across a lot of individual files, which all themselves compressed. Imagine that the effort required to load this into a traditional database. It will be a huge amount of admin overhead. Let's see how easy this is to do using Athena. So I'll move across to the Athena console. Now Athena uses the concepts of databases and tables, just like a traditional relational database but in the context of Athena, they're really just containers because everything happens when you actually query the data. So the first step is to define a table. A table which would be used to read the CloudTrail data through when I query it and I can create this table in one of two different ways, I could create the table from within Athena by just clicking on create table and then selecting the S3 bucket and Athena is capable of reading lots of different data formats as standard but for any AWS products which integrate with Athena, there's a better way of doing this, so I'm going to demonstrate that way. So I'm going to move across to the services dropdown and move back to the CloudTrail console. Once I'm there, I'm going to go to event history and I'm going to select run advanced queries in Amazon Athena. Now, I'll be asked to select the S3 bucket where I've got my CloudTrail log stores. So I'll select the dropdown and just select the ac-cloudtrail234234 bucket and then the statement definition below. What this does is essentially create this virtual table inside Athena. So this is essentially creating what is, in effect, the schema the thing that my raw data is going to be read through, and it defines the attributes of this table. So various fields event version, user identity, event time, event source, event name, and so on. So this is used to actually create the table inside Athena but it isn't actually putting any raw data into that table. It's just a lens which I can use to view the data that's already on S3. Now all of this is beyond the scope of what you'll need for the solutions architect associate exam. So I'm just going to go ahead and hit create table. So now the table has been created inside Athena but again, it's not a real table. It doesn't have any data stored in that table. It's just a lens, which we can use to query the actual data on S3. So I'm going to move across to the thing the console that I'm going to paste in a sample SQL query. It's going to select a number of attributes from this table that I've created inside Athena. So useridentity.username source IP address, event time, additional event data. All of those attributes are going to be selected and pulled from this table that's inside Athena, and it's going to be looking for individual rows where the event name equals console login. So it's going to try to pull all of the console logging events from any of the CloudTrail logs for any region stored in the S3 bucket. So I'm going to go ahead and run the query. Now note how quickly this runs through all the data in the S3 bucket. I can see the runtime 4.9 seconds as well as the amount of data consumed and that's the data that you're going to be billed for. So because Athena is a serverless product, there's no infrastructure charges. You're only going to be billed for the amount of data scanned. F

or the exam, I want you to go away with an understanding about how Athena works. So what it works with and in what situations you would use it. Now you know how it works because I've covered that already in this demonstration. I've created table and I have performed a SQL like query against that table. So now I want to talk about what it works with. **So Athena actually integrates with a lot of other AWS services. So you've seen on my screen how I can use it to query CloudTrail logs. It can also be used to query CloudFront logs or Elastic Load Balancer logs or VPC Flow Logs. It also integrates with AWS Glue and Glue is an ETL product or an extract transform and load service.** Now, this is well beyond the scope of this course, but **an ETL service essentially allows you to load data from one point to another and perform any number of modifications and just knowing that Athena can integrate with Glue** means you'll be able to answer any questions if you see any of these keywords listed but probably what you'll need to focus on if you do get any Athena questions is just how the product works, what its architecture is, and exactly what it can integrate with.

You could also **use Athena though, to query large public data sets that are already stored in S3 and you can also use it to query your own custom application logs**. As long as you can create a table and define exactly how the attributes in this table map onto the attributes in the physical data, you can query that from inside Athena. Now what's more likely to feature in the exam is when to use Athena.

Now **Athena is a serverless product. You don't pay for any time when you're not actually using the product, so you only pay for the data that's queried during your operations. Now that makes it great for ad hoc situations. So things where you need to do the odd query, maybe once a week or once a month, especially if the query is a large, resource intensive query. If you needed to maintain a large scale database platform 24/7 365 for potentially a once a week or once a month query, that would be a lot of resource wasted. Using Athena, you can just pay for the data queried and only pay while it's running and so it's great for ad hoc situations.**

So if you do face any exam questions that talk about **querying large data sets on a fairly infrequent basis then Athena is probably the solution to use**.