Introduction

Introduction

1. Big Data on Azure Databricks

2. Core Azure Databricks Workloads

3. Scaling Azure Databricks Workloads

4. Data Pipelines with Azure Databricks

5. Machine Learning Architectures

Conclusion

Optimize data pipelines

Selecting transcript lines in this section will navigate to timestamp in the video

- [Lynn] Have you been working with data that's growing in volume and complexity and wondering how you're going to compute against this data? We'll be taking a look at managed Apache Spark clusters on Databricks Azure. We'll look at cluster set-up, different types of notebooks and a number of data workflows. These notebooks will include data processing with common scenarios such as Spark SQL, visualization and machine-learning scenarios with Spark ML, third-party libraries such as TensorFlow and Scikit-learn. We'll also look at a data pipelining and architectural patterns. I'm Lynn Langit. We have lots to cover, so let's get started.

What you should know

Selecting transcript lines in this section will navigate to timestamp in the video

- [Instructor] In order to get the most out of this course, there are some concepts that you should have some familiarity with. These include Core Azure concepts, such as an understand of Azure Storage, such as Blob Storage, Azure Compute, such as working with Virtual Machines, and Azure Security. In addition, it will be helpful if you have an understand of Core Database concepts, in particular, working with the SQL Query Language, and some familiarity with Database Storage. Additionally, we will be working with programming and scripting languages, particularly Python, as well as some other languages, such as R and Scala. We're not going deep into the languages, however, if you have some familiarity, you'll get more out of this course. If you want a refresher on some of these concepts, we have some other course in our library that cover Core Spark Concepts, Azure Storage Deep Dive's, and Azure SQL Querying.

About using cloud services

Selecting transcript lines in this section will navigate to timestamp in the video

- [Instructor] In this course we're going to be working with cloud services. Although you can start with an Azure trial account and that's always a good way to get started. Some of the features that we're going to be looking at do require a premium Azure Databricks account and I'll call those out as we go through this course. Also when you're done studying be sure to turn off and delete unused services so that you're not getting unexpected charges on your Azure bill.

Meet Databricks Apache Spark clusters

Selecting transcript lines in this section will navigate to timestamp in the video

- [Instructor] To get us started working with Azure Databricks let's consider this quote from one of the Databricks company founders, Matei Zaharia. He said, I think that by 2020 most data will be in either public clouds or cloud-like private environments. And the key words here are not only cloud but also most data. The volume of data that we need to work with is growing exponentially because of new methods of collection. Whether it's informational data from our phones or genetic data from our bodily fluids we are having more and more data that is interesting and we need to process. So Matei vision started actually when he was a student back at UC Berkeley in California, and he envisioned distributed computing that could help us to process these volumes of data, and that's called Apache Spark. So when we're thinking about this new world of data we go beyond the typical relational databases and we think in terms of streaming systems, data lakes and data warehouses. And as we'll see in this course Azure Databricks allows us to account for these new types of data needs all across the spectrum. Databricks itself is built on top of the open source Apache Spark Core, and it includes in this implementation integration with Microsoft Azure, auto scaling, multi-language notebooks, built-in machine learning frameworks and integrated security. Now as we start to work with this system of course this is built on the open source Apache Spark distributed compute framework. So this framework not only allows for distributed compute it's designed to process its computation in-memory of the various nodes. And it's designed to work with a distributed file system. Spark itself is an ever growing ecosystem. As of this recording the Spark Core API includes integration with the following programming languages, R, SQL, Python, Scala, and Java. Sitting on top of that are higher level frameworks where we can work with Spark SQL and DataFrames, Streaming, Mllib for Machine Learning and GraphX for Graph Computation.

Business scenarios for Spark

Selecting transcript lines in this section will navigate to timestamp in the video

- [Instructor] So as a working cloud architect, what types of business scenarios have I found that are a best fit for Apache Spark technologies? In a nutshell, those are around distributed compute, and really what's driving it is the volume of data. For example, I've been doing quite a lot of work recently in genomic sequencing and analysis of genomic information. The kinds of tasks that I've used Spark for in these types of workflows included data cleansing, or Extract, Transform, and Load; fast data serving pipelines; scalable complex processing; and distributed machine learning. You can think of Azure Databricks as a set of three components. You have the Databricks tools, services, and optimizations that surround the core open source Apache Spark distribution, and Apache Spark itself provides the distributed computation needed for these intensive workloads, and this sits on top of some sort of file system. Now natively in Databricks, you have the DBFS, or the Databricks file system, which is somewhat similar to HDFS, which is somewhat similar to HDFS, Hadoop File System, upon which Apache Spark was originally built; however, it's been many years since Hadoop was open source, and so there's a great number of enhancements. Interestingly, as an alternative to working with DBFS file systems, when integrated with Azure, there are some new ways that you can work with distributed file systems, given some of the Azure services, such as Azure storage blobs, and many more, as we'll see in this course.

Understand Spark key components

Selecting transcript lines in this section will navigate to timestamp in the video

- Now as mentioned in the introduction to this course, if you're unfamiliar or new to working with Apache Spark, you may want to review some other courses in the library to get some foundational learning. If you have done some work with opensource Spark, you will quickly see how the integration really enhances the capability of Spark's key components, which are shown here. And just to review them, you'll see that the core of Spark is a user program and you can see on the left side you instantiate a Spark Context and then you set up a RDD or a structure which will hold the distributed data. That's a resilient data set. And the RDDs then can work to have operations performed on them in a distributed manner. And some of the examples shown here are a Cassandra table, which is a type of key value database, mapping, filtering, keying by, reducing by key, caching, so on and so forth. And you can see, once the data is loaded into the memory of the worker nodes, as RDDs or some higher level structure, such as data tables or other types of data contexts which can sit on top of RDDs, then you'll see that there is a DAG, or a Directed Acyclic Graph, which is used in scheduling the operations in a distributed fashion. And so much of the optimization is handled not only by Spark, but when you're working with the Databricks distribution, you're working with a more highly optimized version of the opensource Spark. It is interesting. Most of the Apache Spark committers, or many of them, are employees of Databricks but not all of them. So the Spark Core is slightly different in terms of performance from the Databricks product, which builds on the Spark Core. And I know, in my case, I have actually run performance testing on various vendor clouds and in this course, we're focusing on Azure, of course, with Spark Core in contrast to Databricks and I do see performance enhancements with Databricks. It's interesting. So you can see that the RDDs and the DAGScheduler live in the Spark Context, which lives in the Spark application. That interacts with the cluster manager, which then sends the scheduled jobs out to the worker nodes, which use executors and have cached data, again, Spark is designed to work in memory, to execute the various data processing tasks. So it is a rather complex environment and, as we work through some of the examples in this course, it will be beneficial if you get a little bit of background with key Spark operations.

Azure Databricks concepts

Selecting transcript lines in this section will navigate to timestamp in the video

- [Instructor] So the Databricks offering around Apache Spark has three component areas, as I mentioned previously. So let's dive a little bit deeper into them. So, at the top level, you have the Databricks collaborative workspace and that's expressed as Databricks Interactive where you can work with your cluster through a Jupyter-style notebook. So this is a browser-based integration that we'll be working with very frequently in this course. I like to think of it as a smart IVE because it has information about not only the execution, but also visualizations and documentation. There's also a Databricks production layer at this level and there are APIs, which you can work with. The Databricks runtime includes Apache Spark core, also Databricks I/O, and now Databricks Serverless, which we'll be exploring in addition to running Databricks on more traditional cloud-based VMs. And surrounding that is Databricks Enterprise Security, which is more and more an absolute requirement for many of my customers. Now in this course, we're going to look at Databricks running on Azure. Databricks is optimized for AWS, or Azure at the time of this recording. So, the Azure implementation provides you with managed, scalable Apache Spark servers, integrated tooling and examples, and as I mentioned, is optimized to run in the Azure cloud. So understanding a bit more about what this type of solution provides for you, Databricks Azure integration gives you fully managed Spark clusters hosted on Azure, Databricks workspaces for collaboration, with the Databricks notebooks, enterprise security including very important Active Directory integration if needed. And also, very key in this implementation is integration with Azure services such as SQL Data Warehouse, Cosmos DB, Azure Blob Storage, Power BI, and others. And if we look at this architectural diagram, it gives us kind of an all up view of what the Azure Databricks offering looks like. So you can see, kind of like the core Databricks offering, it has a layer around integration for working with people, and applications. And then it's surrounding Spark core. So you can see in the center we have the Databricks workspace, Databricks workflows, the runtime, Databricks I/O, Databricks Serverless, and Databricks Enterprise Security. Surrounding the core, we have our target users, our data scientists, engineers, line of business, and others. Our target applications in the Azure ecosystem. Those are applications such as Deep Learning/ML, streaming, data warehousing, Power BI, and others. And notice at the bottom layer, we have several different file storage options. And this is what I mentioned earlier. It's really interesting, this era of cloud. Now we can go beyond just a single storage choice. And this really can impact performance, cost, scalability. You can see this is designed to work with Azure Blob Store, Azure Data Lake Store, Azure SQL Data Warehouse, Apache Kafka, and Hadoop. So given the fact that Apache Spark is open-source, why would it be a good choice for a customer to select managed Apache Spark and cloud-based, so Azure Databricks in this case. Well I see it, from my architect standpoint, as faster time to value. It's a much quicker setup for prototyping experiments, as you'll see when we do some examples in this course. The included Jupyter-style notebooks really provide a lot of value in terms of not only getting your data scientists up and going, but actually translating their work into usable dataflows that can be deployed by your DevOps people. Intuitive and integrated console, very easy to use, and the Apache Spark core committers work for Databricks, as I mentioned earlier, which results in a very performant implementation. It also has granular and integrated security, which is required for so many of my customers. And to summarize, so that we can understand differences between the offerings of Databricks Azure versus Apache Spark. In the world of compute, if you use Azure Databricks you get managed, distributed compute. In Spark, you get distributed compute, but you will be doing the management. You will be managing your VMs, you'll be setting up the operating systems, you be patching them, it's quite a lot more work. In terms of the environment, if you use Azure Databricks, you'll use Azure managed virtual machines or containers. So they'll be patched, they'll be optimized. If you set up Apache Spark on your own then you have to size and select your virtual machine sizes, your docker containers, and the container management, and so on and so forth. The notebook interface provided by Databricks is a sophisticated Jupyter-style notebook, it's not the same thing as Jupyter, they actually have done many enhancements, and again I'll point those out in this course. For regular Apache Spark implementations you would have to set this up yourself again. The command-line interface Databricks includes CLI, or command-line interface. Of course Spark has its own Spark-submit CLI, but Databricks CLI allows you to automate some of the enterprise applications such as management of security.