1
An Introduction to Data Engineering

Data engineering continues to be a fast-growing career path and a role in high demand, as data becomes ever more critical to organizations of all sizes. For those that enjoy the challenge of putting together the "puzzle pieces" that build out complex data pipelines to ingest raw data, and then transform and optimize that data for varied data consumers, it can be a really rewarding career.

In this chapter, we look at the many ways that data has become an important, and increasingly valuable, corporate asset. We also review some of the challenges that organizations face as they deal with increasing volumes of data, and how data engineers can use cloud-based services to help overcome these challenges. We then set the foundations for the hands-on activities in this book by providing step-by-step details on creating a new **Amazon Web Services (AWS)** account.

Throughout this book, we are going to cover a number of topics that teach the fundamentals of developing data engineering pipelines on AWS, but we'll get started in this chapter with these topics:

- The rise of big data as a corporate asset
- The challenges of ever-growing datasets
- The role of the data engineer as a big data enabler
- The benefits of the cloud when building big data analytic solutions
- Hands-on – create or access an AWS account for following along with the hands-on activities in this book

Technical requirements

You can find the code files of this chapter in the GitHub repository using the following link: https://github.com/PacktPublishing/Data-Engineering-with-AWS-2nd-edition/tree/main/Chapter01.

The rise of big data as a corporate asset

You don't need to look too far or too hard to hear about the many ways that big data and data analytics are transforming organizations and having an impact on society as a whole. We hear about how companies such as *TikTok* analyze large quantities of data to make personalized recommendations about which video clip to show a user next. We also read countless articles about how the new generation of chatbots (like *ChatGPT* from *OpenAI* or *Bard* from *Google*) have been trained on massive datasets, and as a result, are able to have human-like conversations on a wide range of topics. We experience how companies like *Amazon* and *Netflix* are able to recommend products or videos we may be interested in, based on our purchase and viewing history. All of these companies have innovated and added customer value by performing complex analyses on very large datasets.

We also see the importance of data in large companies, as demonstrated by those companies creating a new executive C-level position – the **Chief Data Officer (CDO)**. According to an article (https://hbr.org/2021/08/why-do-chief-data-officers-have-such-short-tenures) in the Harvard Business Review, the role of CDO was first established by Capital One (a technology-driven U.S. bank) in 2002. By 2012, it was estimated that 12% of firms had a CDO according to a NewVantage Partners survey, and by 2021, this had grown to 65% of firms having a CDO.

There is no doubt that data, when harnessed correctly and optimized for maximum analytic value, can be a game-changer for an organization. At the same time, those companies that are unable to effectively utilize their data assets risk losing a competitive advantage to others

that do have a comprehensive data strategy and effective analytic and machine learning programs.

Organizations today tend to be in one of the following three states:

- They have an effective and modernized data analytics and machine learning program that differentiates them from their competitors.
- They are conducting proof of concept projects to evaluate how modernizing their analytic and machine learning programs can help them achieve a competitive advantage.
- Their leaders are having sleepless nights worrying about how their competitors are using new analytics and machine learning programs to achieve a competitive advantage over them.

No matter where an organization is in its data journey, if it has been in existence for a while, it has likely faced a number of common data-related challenges. Let's look at how organizations have typically handled the challenge of ever-growing datasets.

The challenges of ever-growing datasets

Organizations have many assets, such as physical assets, intellectual property, the knowledge of their employees, and trade secrets. But for too long, organizations did not fully recognize that they had another extremely valuable asset, and they failed to maximize the use of it – the vast quantities of data that they had gathered over time.

That is not to say that organizations ignored these data assets, but rather, due to the expense and complex nature of storing and managing this data, organizations tended to only analyze and keep a subset of their data.

Initially, data may have been stored in a single database, but as organizations and their data requirements grew, the number of databases

exponentially increased. Today, with the modern application development approach of microservices, companies commonly have hundreds, or even thousands, of databases. Faced with many data silos, organizations invested in data warehousing systems that would enable them to ingest data from multiple siloed databases into a central location for analytics. But due to the expense of these systems, there were limitations on how much data could be stored, and some datasets would either be excluded or only aggregate data would be loaded into the data warehouse. Data would also only be kept for a limited period of time, as data storage for these systems was expensive, and therefore, it was not economical to keep historical data for long periods. There was also a lack of widely available tools and compute power to enable the analysis of extremely large, comprehensive datasets.

As organizations continued to grow, multiple data warehouses and data marts would be implemented for different business units or groups, and organizations still lacked a centralized, single-source-of-truth repository for their data. Organizations were also faced with new types of data, such as semi-structured or even unstructured data, and analyzing these datasets using traditional tools was a challenge.

As a result, new technologies were invented that were better able to work with very large datasets and different data types. **Hadoop** was a technology created in the early 2000s at Yahoo as part of a search engine project that wanted to index 1 billion web pages. Over the next few years, Hadoop, and the underlying **MapReduce** technology, became a popular way for all types of companies to store and process very large datasets. However, running a Hadoop cluster was a complex and expensive operation requiring specialized skills.

The next evolution for big data processing was the development of **Spark** (later taken on as an Apache project and now known as **Apache Spark**), a new processing framework for working with big data. Spark showed significant increases in performance when working with large datasets due to the fact that it did most processing in memory, signifi-

cantly reducing the amount of reading and writing to and from disks. Today, Apache Spark is often regarded as the gold standard for processing large datasets and is used by a wide array of companies.

In parallel with the rise of Apache Spark as a popular big data processing tool was the rise of the concept of data lakes – an approach that uses low-cost object storage as a physical storage layer for a variety of data types, Apache Hive as a central catalog of all the datasets, and makes that data available for processing with a wide variety of tools, including Apache Spark. AWS's own website uses the following definition of data lakes:

> *A data lake is a centralized repository that allows you to store all your structured and unstructured data at any scale. You can store your data as-is, without having to first structure the data, and run different types of analytics—from dashboards and visualizations to big data processing, real-time analytics, and machine learning to guide better decisions.*

Data lakes were great for centralizing data but were often run by a centralized team that would look to ingest data from all the data silos in an organization, transform and aggregate the data, and make it available centrally for use by other teams. While this was a definite improvement on having databases and data warehouses spread out with no central repository or governance, there was still room for improvement.

By centralizing all the data and having a single team manage this repository of data, the teams working to transform and extract additional value out of the data were often not the people most familiar with the business context behind the data.

To address this, Zhamak Dehghani (a data consultant working for Thoughtworks at the time) developed a new approach, which eventually became known as a **data mesh**. While we will cover a brief introduction to data mesh architecture here, we will cover this topic in more detail in *Chapter 15, Implementing a Data Mesh Architecture*.

With a data mesh architecture, the idea is to make the teams that generate the data responsible for creating an analytics version of the data, and then make that data easily accessible to the rest of the organization without needing to make multiple copies of the data.

By 2022, this concept had gained widespread appeal, and many companies were working to implement a data mesh approach for their data. Some took a limited view of what a data mesh was and considered it primarily a means of sharing data between teams without needing to physically move or copy the data. However, a full data mesh implementation went well beyond the technology of how to share data. A data mesh implementation meant a change to the processes of how operational data was converted into analytical data, and along with it, the personas that were responsible for the data. A data mesh implementation was not just a technical implementation but rather a change to the culture and operation of teams within an organization.

Whereas in many organizations, large-scale analytics had been done by a centralized team, with a data mesh approach, the team that owns an application that generates data must *productize* that data to make it available to the rest of the business. Much like DevOps had changed how development teams worked to create and support software, the data mesh approach meant that product teams needed to work differently in how they generated and shared analytical data. They would appoint a data product manager who would be responsible for the task of taking operational data and creating an analytics data product from that. For the product they created, they would take ownership of ensuring data quality, the freshness of data, communication around

changes to the product (such as schema changes), and so on. They would effectively be product managers, outlining a roadmap for the data analytics product they would create, and they would be responsive to customer feedback on the data product.

With a data mesh, there would still be a central data team, but this team would be responsible for creating a centralized data platform, which all the different data product teams could then use. So rather than being data engineers that transformed data, the centralized team would focus on being data platform engineers, creating a standardized platform that met best practices. They would then make this platform available to individual development teams to use in order to create and share their own data analytic products.

Having looked at how data analytics became an essential tool in organizations, let's now look at the roles that enable maximizing the value of data for a modern organization.

The role of the data engineer as a big data enabler

Amid the increasing recognition of data as a valuable corporate asset and the introduction of new technologies to store and process vast amounts of data, there has been an increase in the opportunities and roles available for data-related careers.

Let's look at a sample use case where a sales manager for a consumer goods organization wants to better understand which alternate products a customer considers before purchasing their product. In addition, they also want to have a better way of predicting product demand by category based on external factors, such as the expected weather.

Achieving the desired outcomes as specified by the sales manager will require bringing in data from multiple internal and external sources.

Datasets that could be relevant to this scenario may include the following:

- Customer, product, and order relational databases
- Web server logs from the consumer-facing storefront
- Third-party sales data from online marketplaces where relevant products are sold (such as Amazon.com)
- Other relevant third-party datasets that may influence sales (for example, weather-related data)

Multiple teams would need to be involved in the project, with the following 3 roles playing a primary part in implementing the required solution:

- Data engineer
- Data scientist
- Data analyst

Let's take a look at how these three roles would contribute to this new project.

Understanding the role of the data engineer

The role of a **data engineer** is to do the following:

- Design, implement, and maintain the pipelines that enable the ingestion of raw data into a storage platform
- Transform that data to be optimized for analytics, based on data consumer requirements
- Make that data available for various data consumers using their tool of choice

A data engineer must also ensure that they comply with all required security and governance requirements while performing the above tasks.

In our scenario, the data engineer will first need to design the pipelines that ingest raw data from various internal and external sources. To achieve this, they will use a variety of tools, depending on the source system and whether it will be scheduled batch ingestion or near real-time streaming ingestion (as discussed in *Chapter 6*, *Ingesting Batch and Streaming Data*).

The data engineer is also responsible for transforming the raw input datasets to optimize them for analytics, using various techniques (as discussed in *Chapter 7, Transforming Data to Optimize for Analytics*). The data engineer must also create processes to verify the quality of data, add metadata about the data to a data catalog, and manage the lifecycle of code related to data transformation.

Finally, the data engineer may need to assist in integrating various data consumption tools with the transformed data, enabling data analysts and data scientists to use their preferred tools to draw insights from the data.

The data engineer uses tools such as **Apache Kafka**, **Apache Spark**, and **Presto,** as well as other commercially available products, to build the data pipeline and optimize data for analytics.

The data engineer is much like a civil engineer for a new residential development. The civil engineer is responsible for designing and building the roads, bridges, train stations, and so on to enable commuters to easily commute in and out of the development. In a similar way, the data engineer is responsible for designing and building the infrastructure required to bring data into a central source and for optimizing the data for use by various data consumers.

Understanding the role of the data scientist

The role of a **data scientist** is to draw complex insights and make predictions based on various datasets, using machine learning and artifi-

cial intelligence. The data scientist will combine a number of skills, including computer science, statistics, analytics, and math, in order to help an organization answer complex questions and make informed decisions using data.

Data scientists need to understand the raw data and know how to use that data to develop and train complex machine learning models that will help recognize patterns in the data and predict future trends. In our scenario, the data scientist may build a machine learning model that uses past sales data, correlated with weather information for each day in the reporting period.

They can then design and train this model to help business users get predictions on the likely top-selling categories for future dates based on the expected weather forecast (ice creams sell better on hot days, and umbrellas sell better on rainy days).

Where the data engineer is like a civil engineer building infrastructure for a new development, the data scientist is developing new and advanced products for the residents of the development, such as advanced types of transport in and out of the development. Data scientists create the machine learning models that enable data consumers and business analysts to draw new insights and predictions from data. However, much like the designer of a new airplane is dependent on having an airport where the plane can land and take off, the data scientist is dependent on data engineers creating data pipelines to bring in the data required to train new machine learning models.

Understanding the role of the data analyst

The role of a **data analyst** is to examine and combine multiple datasets in order to help a business understand trends in the data and to make more informed business decisions. While a data scientist develops models that make future predictions or identifies non-obvious patterns in data, the data analyst works with well-structured and mod-

eled data to understand current conditions and to highlight recent patterns from the data.

A data analyst may answer questions such as which menu item sold best in different geographic regions over the past month, or which medical procedure had the best outcome for patients of different ages. These insights help an organization make better decisions for the future.

In our scenario, the data analyst may run complex queries against the different datasets that are available (such as an orders database or web server logs), joining together subsets of data from each source to gain new insights. For example, the data analyst may create a report highlighting which alternate products are most often browsed by a customer before a specific product is purchased. The data analyst may also make use of advanced machine learning models developed by the data scientists to gain further valuable insights.

Where the data engineer is like a civil engineer building infrastructure, and the data scientist is developing new, advanced forms of transportation, the data analyst is like a skilled pilot, using their expertise to get users to their end destination.

Understanding other common data-related roles

Organizations may have other role titles and job descriptions for data-related positions, but generally, these will be a subset of the roles described in the preceding sections.

For example, a **big data architect** or **data platform architect** could be a subset of the **data engineer** role, focused on designing the architecture for big data pipelines, but not building the data-specific pipelines. Or, a **data visualization developer** may be focused on building

out visualizations using business intelligence tools, but this is effectively a subset of the **data analyst** role.

Larger organizations tend to have more focused job roles, while in a smaller organization, a single person may take on the role of data engineer, data scientist, and data analyst.

In this book, we will focus on the role of the data engineer, and dive deep into how a data engineer is able to build complex data pipelines using the power of cloud computing services. Let's now look at how cloud computing has simplified how organizations are able to build and scale out big data processing solutions.

The benefits of the cloud when building big data analytic solutions

For a long time, organizations relied on complex systems that they would run in their own data centers to help them capture, store, and process large amounts of data. But over the last decade or so, there has been a trend of an increasing amount of data that organizations want to store and analyze, and on-premises systems have struggled to scale to keep up with demand. Scaling up these traditional tools for managing ever-increasing dataset sizes has been expensive, complex, and time-consuming, and organizations have been seeking alternative solutions to cope with the increasing data volumes.

Ever since Amazon launched **AWS** in 2006, organizations have been realizing the benefits of running their workloads in the cloud. Cloud computing enables scalability, cost efficiency, security, and automation that most companies find impossible to achieve within their own data centers, and this applies to the area of data analytics as well. One of the first AWS services was **Amazon Simple Storage Service** (**Amazon S3**), a cloud-based object store that offers essentially unlimited scalability at low cost, and yet provides durability and availability that most data center managers could only dream of achieving. Today,

Amazon S3 has become the physical storage layer for many thousands of data lake projects, and a wide ecosystem of analytic tools has been created to work with the service.

Successful data engineers need to understand the tools available in the cloud for building out complex data analytic projects and understand which set of tools is best to achieve the outcome needed for their project. In this book, you will learn more about AWS services for working with big data, and you will gain hands-on experience in developing a data engineering pipeline in AWS.

To get started, you will need either an existing AWS account, or you will need to create a new AWS account so that you can follow along with the practical examples. In the next section, we provide step-by-step instructions for creating a new AWS account.

Hands-on – creating and accessing your AWS account

The projects in this book require you to access an AWS account with administrator privileges. If you already have administrator privileges for an AWS account and know how to access the AWS Management Console, you can skip this section and move on to *Chapter 2*, *Data Management Architectures for Analytics*.

If you are making use of a corporate AWS account, you will want to check with your AWS cloud operations team to ensure that your account has administrative privileges. Even if your daily-use account does not allow full administrative privileges, your cloud operations team may be able to create a **sandbox** account for you.

What is a sandbox account?

A sandbox account is an account isolated from your corporate production systems with relevant guardrails and

governance in place and is used by many organizations to provide a safe space for teams or individual developers to experiment with cloud services.

If you cannot get administrative access to a corporate account, you will need to create a personal AWS account or work with your cloud operations team to request specific permissions needed to complete each section. The exercises in this book assume you have administrative access and the full details of required granular permissions will not be covered, but you can review the AWS documentation for information on granular permission requirements for each service.

An important note about costs associated with hands-on tasks in this book

If you are creating a new personal account or using an existing personal account, you will incur and be responsible for AWS costs as you follow along in this book. While some services may fall under AWS Free Tier usage, some of the services covered in this book will not. **We strongly encourage you to set up budget alerts within your account and to regularly check your billing console.**

See the AWS documentation on setting up billing alarms to monitor your costs at https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/monitor_estimated_charges_with_cloudwatch.html.

Creating a new AWS account

To create a new AWS account, you will need the following things:

- An email address (or alias) that has not been used before to register an AWS account
- A phone number that can be used for important account verification purposes
- Your credit or debit card, which will be charged for AWS usage outside of the Free Tier

> **A tip regarding the phone number you use when registering**
>
> It is important that you keep your contact details up to date for your AWS account as, if you lose access to your account, you will need access to the email address and phone number registered for the account to restore access. If you expect that your contact number may change in the future, consider registering a virtual number that you will always be able to access and that you can forward to your primary number. One such service that enables this is Google Voice (`http://voice.google.com`).

The following steps will guide you through creating a new AWS account:

1. Navigate to the AWS landing page at `http://aws.amazon.com`.
2. Click on the **Create an AWS Account** link in the top right-hand corner.
3. Provide an **email address**, provide a **name** for your account, and then click on **Verify email address**. You will be emailed a verification code to verify your email, which you need to enter on the form to continue.

> **A tip about reusing an existing email address**
> Some email systems support adding a "+" sign followed by a few characters to the end of the username portion of your email address in order to create a

> unique email address that still goes to your same mailbox. For example, `atest.emailaddress@gmail.com` and `atest.emailaddress+dataengineering@gmail.com` will both go to the primary email address inbox. If you have used your primary email address previously to register an AWS account, you can use this tip to provide a unique email address during registration but still have emails delivered to your primary account.

4. Once you verify using the code emailed to you, specify a new secure **password** for your account (one that you have not used elsewhere). Then click on **Continue**.
5. Select **Business** or **Personal** for the account type (note that the functionality and tools available are the same no matter which one you pick).
6. Provide the requested personal information and then, after reviewing the terms of the **AWS Customer Agreement**, click the checkbox if you agree to the terms, and then click **Continue**.
7. Provide a credit or debit card for payment information and select **Verify and Continue**.
8. Provide a **phone number** for a verification text or call, enter the characters shown for the **security check**, and complete the verification.
9. Select a **support plan** (basic support is free, but only provides self-service access to support resources) and complete the signup.
10. You will receive a notification that your account is being activated. This usually completes in a few minutes, but it can take up to 24 hours. Check your email to confirm account activation.

> **What to do if you don't receive a confirmation email within 24 hours**

If you do not receive an email confirmation within 24 hours confirming that your account has been activated, follow the troubleshooting steps provided by AWS Premium Support at https://aws.amazon.com/premiumsupport/knowledge-center/create-and-activate-aws-account/.

Accessing your AWS account

Once you have received the confirmation email confirming that your account has been activated, follow these steps to access your account and create a new admin user:

1. Access the AWS console login page at http://console.aws.amazon.com.
2. Make sure **Root user** is selected, and then enter the email address that you used when creating the account.
3. Enter the password that you set when creating the account.

Best practices for securing your account

When you log in using the email address you specified when registering the account, you are logging in as the account's *root user*. It is a recommended best practice that you do not use this login for your day-to-day activities but rather, only use this when performing activities that require the root account, such as creating your first **Identity and Access Management (IAM)** user, deleting the account, or changing your account settings. For more information, see https://docs.aws.amazon.com/IAM/latest/UserGuide/id_root-user.html.

It is also strongly recommended that you enable **Multi-Factor Authentication (MFA)** on this and other administrative accounts. To enable this, see https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa_enable_virtual.html.

In the following steps, we are going to create a new IAM administrative user account:

1. In the AWS Management Console, confirm which **Region** you are currently in. You can select any region, such as the region closest to you geographically.

Selecting a region

The region you select in the AWS console is the geographical area of the world where the AWS resources you create will be deployed. It generally makes sense to deploy to the region closest to where you are located; however, this is not always the case. For example, not all AWS services are available in all regions

infrastructure/regional-product-services/).

Another factor to consider is that the pricing for AWS services differs from region to region, so also take this into account when selecting a region to use for the exercises in this book. Finally, make sure that you are always in the same region when working through the exercises in each chapter.

For more information on selecting a region, refer to the AWS blog post *What to Consider when Selecting a Region for your Workloads* at

https://aws.amazon.com/blogs/architecture/what-to-consider-when-selecting-a-region-for-your-workloads/.

In the following screenshot, the user is in the **Ohio** region (also known as **us-east-2**).

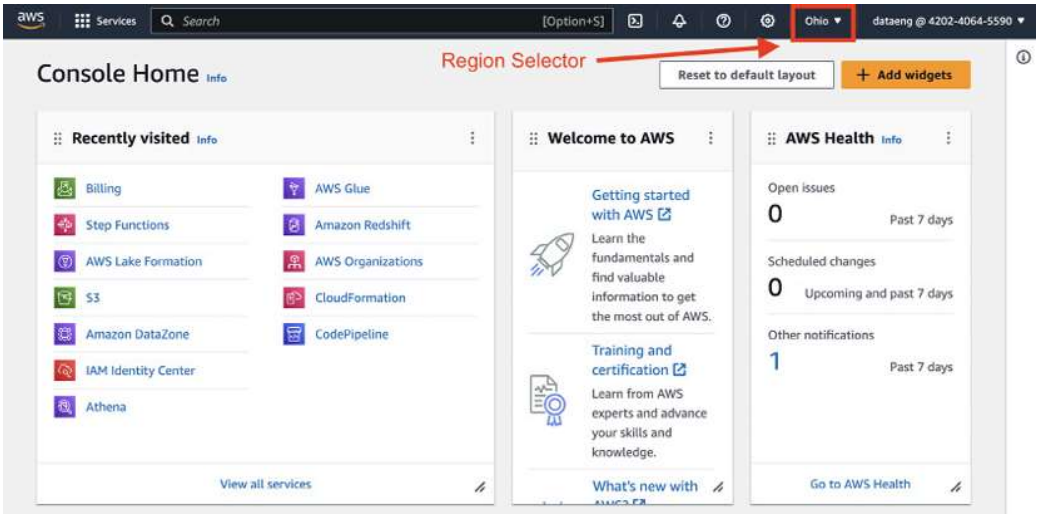

Figure 1.1: AWS Management Console

2. In the **Search** bar at the top of the screen, type in `IAM` and press *Enter*. This brings up the console for **Identity and Access Management** (**IAM**).
3. On the left-hand side menu, click **Users** and then **Add users**.
4. Provide a username, and then select the checkbox for **Enable console access - *optional***.
5. Select **Custom password**, provide a password for console access, select whether to force a password change on the next login, then click **Next**.

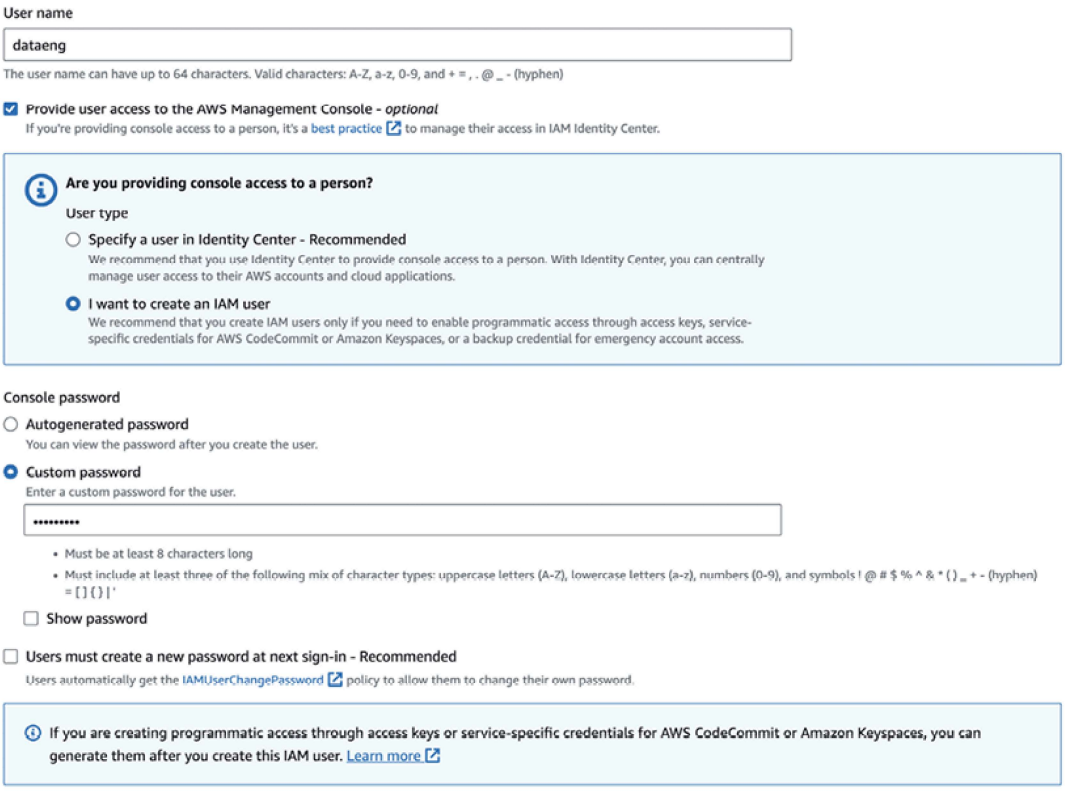

Figure 1.2: Creating a new user in the AWS Management Console

6. For production accounts, it is best practice to grant permissions with a policy of least privilege, giving each user only the permissions they specifically require to perform their role. However, AWS managed policies can be used to cover common use cases in test accounts, and so to simplify the setup of our test account, we will use the **AdministratorAccess** managed policy. This policy gives full access to all AWS resources in the account.
   On the **Set permissions** screen, select **Attach policies directly** from the list of policies, select **AdministratorAccess**, then click **Next: Tags**.

7. Optionally, specify tags (key-value pairs), then click **Next**.
8. Review the settings, and then click **Create user**.
9. Take note of the **Console sign-in URL** link that you will use to sign into your account.

For the remainder of the tutorials in this book, you should log in using the URL link provided and the username and password you set for your IAM user. You should also strongly consider enabling MFA for this account, a recommended best practice for all accounts with administrator permissions.

Summary

In this chapter, we reviewed how data is becoming ever more important for organizations looking to gain new insights and competitive advantage, and introduced some core big data processing technologies and approaches. We also looked at the key roles related to managing, processing, and analyzing large datasets, and highlighted how cloud technologies enable organizations to better deal with the increasing volume and variety of data.

In our first hands-on exercise, we provided step-by-step instructions for creating a new AWS account, which can be used through the remainder of this book as we develop our own data engineering pipeline.

In the next chapter, we dig deeper into current approaches, tools, and frameworks that are commonly used to manage and analyze large datasets, including data warehouses, data marts, data lakes, and data mesh. We also get hands-on with AWS again, but this time we will use the AWS **Command Line Interface** (**CLI**) to create new Amazon S3 buckets.

Learn more on Discord

To join the Discord community for this book – where you can share feedback, ask questions to the author, and learn about new releases – follow the QR code below:

https://discord.gg/9s5mHNyECd