8
Identifying and Enabling Data Consumers

A data consumer can be defined as a person, or application, within an organization that needs access to data. Data consumers can vary from staff that pack shelves and need to know stock levels to the CEO of an organization that needs data to make a decision on which projects to invest in. A data consumer can also be a system that needs data from a different system.

Everything a data engineer does is to make datasets useful and accessible to data consumers, which, in turn, enables the business to gain useful insights from their data. This means delivering the right data, via the right tools, to the right people or applications, at the right time, to enable the business to make informed decisions.

Therefore, when designing a data engineering pipeline (as covered in *Chapter 5*, *Architecting Data Engineering Pipelines*), data engineers should start by understanding business objectives, including who the data consumers are and what their requirements are.

We can then work backward from these requirements to ensure that we use the appropriate tools to ingest data at the required frequency (streaming or batch, for example). We can also ensure that we create transformation pipelines that transform raw data sources into data that meets the consumer's specific requirements. And, finally, understanding our data consumers will guide us in selecting a target location and format for our transformed data that is compatible with the tools that best enable our data consumers.

Maintaining an understanding of how the data is consumed, as well as knowledge of any downstream dependencies, will also help data engineers support different types of data consumers as they work with a variety of datasets.

In this chapter, we will do a deep dive into data consumers by covering the following topics:

- Understanding the impact of data democratization
- Meeting the needs of business users with data visualization
- Meeting the needs of data analysts with structured reporting
- Meeting the needs of data scientists and ML models
- Hands-on – transforming data using AWS Glue DataBrew

Technical requirements

For the hands-on exercise in this chapter, you will need permission to use the AWS Glue DataBrew service. You will also need to have access to the AWS Glue Data Catalog and any underlying Amazon S3 locations for the databases and tables that were created in the previous chapters. If you are using the administrative user created in *Chapter 1*, then you will have the necessary permissions.

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/Chapter08**

Understanding the impact of data democratization

At a high level, business drivers have not changed significantly over the past few decades. Organizations are still interested in understanding market trends and customer behavior, increasing customer retention, improving product quality, and improving speed to market. However, the analytics landscape, the teams and individual roles that

deliver business insights, and the tools that are used to deliver business value have evolved.

Data democratization – the enhanced accessibility of data for a growing audience of users, in a timely and cost-efficient manner – has become a standard expectation for most businesses. Today's varied data consumers expect to be able to get access to the right data promptly, using their tool of choice to consume the data.

In fact, as datasets increase in volume and velocity, their gravity will attract more applications and consumers. This is based on the concept of *data gravity*, a term coined by Dave McCrory, which suggests that data has mass. That is, as datasets increase in size, they attract more users and become more difficult to move.

The takeaway here is that as data volumes grow, and the velocity at which data is ingested increases, the more business users demand easy access to high-quality data. Delivering on this becomes critical for enabling a business to stay competitive.

A growing variety of data consumers

Over the past few years, we have seen an increase in the number and type of data consumers within an organization, and these data consumers are constantly looking for new data sources and tools. As a result, in today's modern organizations, we can expect to find a wide variety of data consumers – from traditional business users and data analysts to data scientists, machine-to-machine applications, as well as new types of business users (as organizations become more data-driven, it's not only senior managers that need access to data, but employees at all levels of the organization need access to relevant, high-quality data).

Beyond just the ability to run SQL queries and generate scheduled reports based on a pre-existing dataset, we see data analysts who also

want the ability to do ad hoc data cleansing and exploration, as well as the ability to join structured data with semi-structured data or metadata extracted from unstructured data. For example, they may want to examine how social media drives sales trends.

And business users now expect dashboards to be refreshed with real, or near-real-time, data. They also want these dashboards to be accessible from anywhere, on many different types of mobile devices. Furthermore, they are interested in more than just sales or ERP data. Analysts and business users are interested in social media data to identify consumer trends, and insurance and real estate companies are looking for data to be extracted from documents (such as medical reports or property appraisals). In the manufacturing industry, a variety of data consumers want access to data that's been collected from machines, devices, and vehicles for use cases such as proactively anticipating maintenance requirements.

Data consumers are also no longer limited to individual humans or teams. We are seeing a growing need for business applications to access data, be fed data, or be triggered based on an event or trend in the data. Call centers are interested in real-time transcripts of audio calls for sentiment analysis and tagging calls for manager review. They are also looking for applications and integrations that would use real-time call transcriptions, or full-text analysis of corporate documents, to reduce the time agents spend searching for answers. Engagement platforms map the customer journey and use every event (for example, email opened or email ignored) to tailor the customer experience.

Finally, organizations have a growing need for data science, and as such, the number of data scientist roles within those organizations is increasing. Data scientists develop **machine learning** (**ML**) models that can identify non-obvious patterns in large datasets or make predictions about future behaviors based on historical data. Data scientists usually require access to raw, non-aggregated data, and require

large volumes of data to train a machine learning model and test the model for accuracy.

How a data mesh helps data consumers

With the increase in the number of datasets within an organization, and with a growing set of users wanting access to data across an organization, it can be very difficult for data consumers to find the data they need. Once a data consumer does find an available dataset, they need to be able to easily understand whether the dataset they found will meet their needs, and also need to get access to the data without complicated processes and the need to create multiple copies of the dataset.

A data mesh strategy (which we cover in more detail in *Chapter 15, Implementing a Data Mesh Strategy*), includes some core principles that help make datasets more easily discoverable, understandable, and accessible by data consumers. One of the data mesh principles is that of a self-serve data platform, and this platform is intended to streamline the experience of data consumers discovering, accessing, and using datasets.

The self-serve data platform often includes a business data catalog as the mechanism to make data discoverable. Data consumers can go to the catalog to search and discover all the datasets available in an organization. Each dataset published to the catalog should include additional metadata that helps the data consumer better understand the data (such as where the data comes from, who owns it, and business metadata providing more context around the dataset). The catalog should also provide a mechanism for a consumer to request access to a dataset, and then, if needed, the ability to automatically route that request to the appropriate person to approve the access. Once approved, the self-serve data platform should make it easy for the consumer to access the dataset, without needing to physically copy the dataset.

Let's now take a deeper dive into some of the different types of data consumers that we can find in today's organizations. We will also look at how data engineers can help enable each of these data consumers.

Meeting the needs of business users with data visualization

Some roles within an organization, such as data analysts, have always had easy access to data. For a long time, these roles were effectively gatekeepers of the data, and any "ordinary" business users that had custom data requirements would need to go through the data gatekeepers.

However, over the past decade or so, the growth of big data has expanded the thirst and need for custom data among a growing number of **business users**. Business users are no longer willing to tolerate having to go through long, formal processes to access the data they need to make decisions. Instead, users have come to demand easier, and more immediate, access to wider sets of data.

To remain competitive, organizations need to ensure that they enable all the decision-makers in their business to have easy and direct access to the right data. At the same time, organizations need to ensure that good data governance is in place and that data consumers only have access to the data they need (as we discussed in *Chapter 4*, *Data Governance, Security, and Cataloging*). Data engineers are key to enabling this.

AWS tools for business users

Business users have mixed skill sets, ranging from those that are Excel power users and are comfortable with concepts such as pivot tables, to executives who want easy access to dashboards that provide visualizations that summarize complex data.

As a data engineer, you need to be able to provide solutions that meet the needs of these diverse business users. Within AWS, the primary tool that's used by business users is **Amazon QuickSight**, a cloud-based **Business Intelligence** (**BI**) application. QuickSight enables the creation of easy-to-access visualizations and reports but also provides functionality for advanced users to dig deeper into the data while providing strong security and governance controls. Amazon QuickSight is cloud-based and can easily be provisioned for hundreds, or even thousands, of users in an organization.

A quick overview of Amazon QuickSight

We will do a deep dive into **Amazon QuickSight** in *Chapter 12*, *Visualizing Data with Amazon QuickSight*, but in this section, we will have a brief look at some of the primary ways that business users can use this tool.

Amazon QuickSight provides interactive access to data for business users, with many different types and styles of charts supported. A dashboard can display data from multiple different data sources, and users can filter data, sort data, and even drill down into specific aspects of a dataset. In addition to dashboards, QuickSight can also be used to generate multi-page reports.

Business users can elect to receive **dashboards** via regular emails or can access and interact with dashboards on-demand via the QuickSight portal or the QuickSight mobile app. Dashboards can also be embedded into existing web portals and apps, making these rich data visualizations accessible via existing tools that business users have access to.

With a feature called **Amazon QuickSight Q**, business users can ask questions in natural language and receive answers, along with visualizations. This feature uses the power of **Natural Language Processing**

(**NLP**) based machine learning models to understand the intent of a query and to then automatically identify appropriate data sources and create relevant visuals. For example, a sales manager could type in a query such as "*show me sales this month by segment*," and QuickSight will create a chart showing the relevant information.

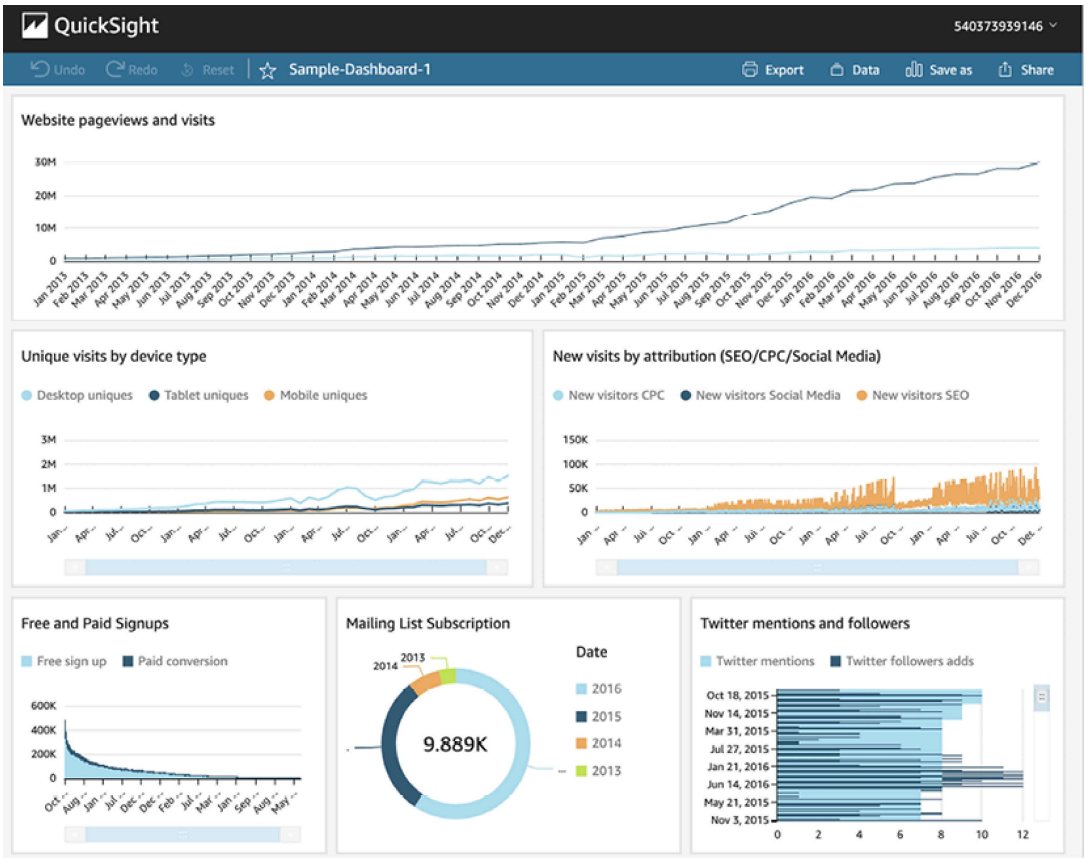

Figure 8.1: A sample QuickSight dashboard

While some users may have previously used spreadsheets to explore datasets using custom-built charts and pivot tables, QuickSight can provide the same functionality but in a much easier-to-use way. QuickSight also provides security, governance, and auditability, which is not possible when users share ad hoc spreadsheets.

QuickSight can use data from many different sources, including directly from an S3-based data lake, databases (such as Redshift, MySQL, and Oracle), SaaS applications (including Salesforce, ServiceNow, Jira, and others), as well as numerous other sources.

As a data engineer, you may be involved in helping set up QuickSight and may need to configure access to the various data sources. QuickSight users with relevant access can combine different data sources directly, thereby enabling them to build the visualizations the business requires without going through traditional data gatekeepers. However, there may also be times when you are asked to create new datasets in a data lake or data warehouse (such as Redshift or Snowflake) so that QuickSight users can access the required data without needing to combine and transform datasets themselves.

We are now going to move on and explore a different type of data consumer – the data analyst. But for a deeper dive into QuickSight, including a hands-on exercise on creating a QuickSight visual, refer to *Chapter 12*, *Visualizing Data with Amazon QuickSight*.

Meeting the needs of data analysts with structured reporting

While business users make use of data to make decisions related to their job in an organization, a data analyst's full-time job is all about the data – analyzing datasets and drawing out insights for the business.

If you look at various job descriptions for **data analysts**, you may see a fair amount of variety, but some elements will be common across most descriptions. These include the following:

- Cleansing data and ensuring data quality when working with ad hoc data sources.
- Developing a good understanding of their specific part of the business (sometimes referred to as becoming a domain specialist for their part of the organization). This involves understanding what data matters to their part of the organization, which metrics are important, and so on.

- Interpreting data to draw out insights for the organization (this may include identifying trends, highlighting areas of concern, and performing statistical analysis on data). The data analyst also needs to present the information they've gathered, as well as their conclusions, to business leaders.
- Creating visualizations using powerful BI software (such as Amazon QuickSight) that other business users can then interact with.
- Doing an ad hoc analysis of data using structured query languages such as SQL.

A data analyst is often tasked with doing complex data analysis to answer specific business questions. Examples, as described earlier in this book, include identifying which products are the most popular by different age or socio-economic demographics. Another example is what percentage of customers have browsed the company's e-commerce store more than 5 times, for more than 10 minutes at a time, in the last 2 weeks, but have not purchased anything.

At times, a data analyst may make use of data in the data lake that has already been through formal data engineering pipelines, which means it has been cleaned and checked for quality. At other times, a data analyst may need to ingest new raw data, and in these cases, they may be responsible for data cleansing and performing quality checks on the data.

Some of the work a data analyst does may be to use ad hoc SQL queries to answer very specific queries for a certain project, while at other times they may create reports, or visualizations, that run on a scheduled basis to provide information to business users.

AWS tools for data analysts

Data analysts may use a variety of tools as they work with diverse datasets. This includes using query languages, such as SQL, to explore

data in a data warehouse such as Redshift or data in a traditional database. A data analyst may also use advanced toolsets such as **Python** or **R** to perform data manipulation and exploration. Visual transformation tools may also be used by the data analyst to cleanse and prepare data when working with ad hoc data sources that have not been through formal data engineering pipelines.

Data analysts also use BI tools, such as **Amazon QuickSight**, to create advanced visualizations or multi-page reports for business users. We covered Amazon QuickSight previously, so let's explore some of the other tools in AWS that can be used by data analysts.

Amazon Athena

Amazon Athena is a service that enables users to run complex SQL queries against a variety of data sources. This can be used to perform ad hoc exploration of data, enabling the data analyst to learn more about the data and test out different queries. Users also have the ability to use an integrated notebook environment to run Spark code for complex queries.

Using Athena, a data analyst can run queries that join data from across tables in different data sources. For example, using Athena, you can run a single query that brings data in from S3 and joins that with data from Redshift.

In *Chapter 11*, *Ad Hoc Queries with Amazon Athena*, we will do a deeper dive into the Athena service.

AWS Glue DataBrew

Data analysts often need to use new sources of data to answer new questions and may need to perform some data transformation on these datasets. While creating these new insights, the data analyst may work closely with business users to develop the reports, visualizations, metrics, or other data as needed. Part of this iterative process may in-

volve creating ad hoc transformation pipelines to ingest, cleanse, join, and transform data.

Once the deliverable has been finalized (data sources identified, transformations determined, and so on), the data analyst may work with their data engineering team to formalize the pipeline. This is a recommended best practice to ensure that all pipelines are contained in a source control system, are part of formal deployment processes, and so on. As such, data engineers should work closely with data analysts, and always be ready to help formalize the ad hoc pipelines that a data analyst may create and that the business has come to depend on.

One of the AWS tools that is very popular with data analysts is the **AWS Glue DataBrew** service. Using DataBrew, data analysts can easily cleanse new data sources and transform and join data from different tables to create new datasets.

This can all be done with the Glue DataBrew **visual interface**, without the data analyst needing to write any code.

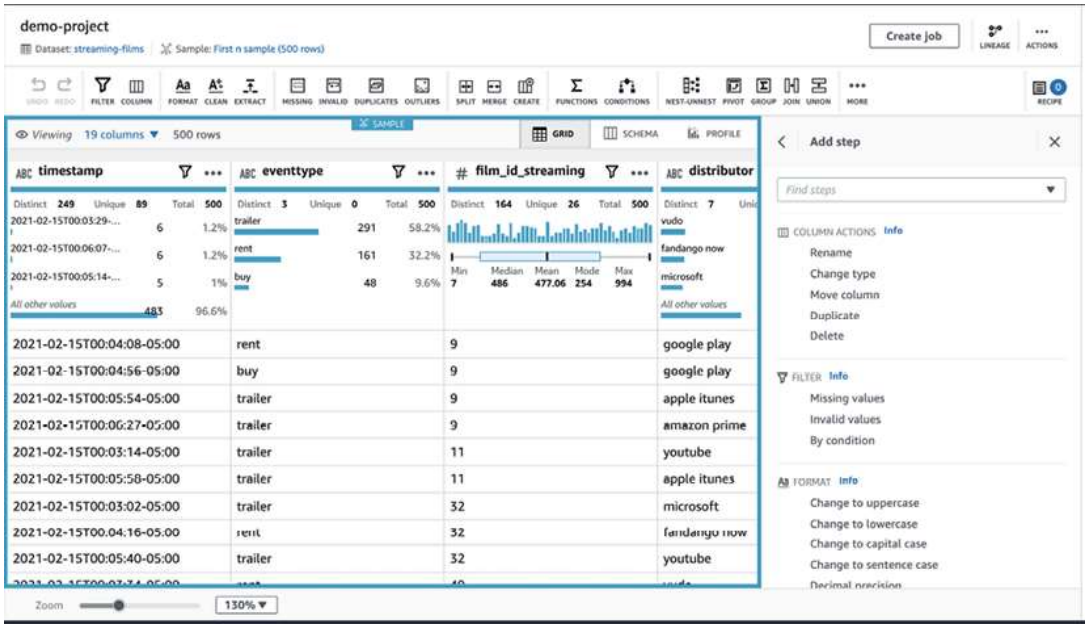

Figure 8.2: The AWS Glue DataBrew visual transform designer

Glue DataBrew can connect to many different data sources, including Redshift and Snowflake, JDBC databases, S3, Glue/Lake Formation tables, as well as other Amazon services such as AWS Data Exchange and Amazon AppFlow. DataBrew also includes over 250 **built-in transforms** that can be used by data analysts to easily perform common data cleansing tasks and transformations. In the hands-on section of this chapter, you will get to use some of these built-in transforms.

Running Python or R in AWS

Some data analysts have advanced coding skills that they put to use to explore and visualize data using popular programming languages such as Python and R. These languages include many functions for statistically analyzing datasets and creating advanced visualizations.

Python code can be run using multiple services in AWS, including the following:

1. **AWS Lambda**: Can run Python code in a serverless environment, for up to a maximum of 15 minutes of runtime
2. **AWS Glue Python shell**: Can run Python code in a serverless environment, with no limit on how long it runs
3. **Amazon EC2**: A compute service where you can install Python and run Python code

In addition, if working with large datasets where a single compute node does not provide the needed processing power, *AWS Glue for Ray* is an engine option on AWS Glue that enables processing large datasets with Python and popular Python libraries. AWS Glue for Ray supports the popular `ray.io` open-source compute framework that enables running Python code over multi-node clusters.

RStudio, a popular IDE that can be used for creating data analytic projects based on the R programming language, can also be run using multiple services in AWS:

1. **RStudio** can be run on Amazon EC2 compute instances, enabling data analysts to create R-based projects for data analysis. See the AWS blog titled *Running R on AWS* (https://aws.amazon.com/blogs/big-data/running-r-on-aws/) for more information on how to set this up.
2. If you're working with very large datasets, RStudio can also be run on Amazon EMR, which uses multiple compute nodes to process large datasets. See the AWS blog titled *Statistical Analysis with Open-Source R and RStudio on Amazon EMR* (https://aws.amazon.com/blogs/big-data/statistical-analysis-with-open-source-r-and-rstudio-on-amazon-emr/) for more information on how to use R with Amazon EMR.

Data engineers can help enable data analysts who have strong Python or R skills by helping them configure these coding environments in AWS. Data engineers can also help formalize data transformation pipelines in those cases where a data analyst has created an ad hoc pipeline for processing that the business has subsequently come to use on an ongoing basis.

While data analysts are primarily responsible for deriving insights out of data that reflect current trends, as well as the current state of the business, data scientists generally use data to predict future trends and requirements. In the next section, we will dive deeper into the role of the data scientist.

Meeting the needs of data scientists and ML models

Over the past decade, the field of **ML** has significantly expanded, and the majority of larger organizations now have **data science** teams that use ML techniques to help drive the objectives of the organization.

Data scientists use advanced mathematical concepts to develop ML models that can be used in various ways, including the following:

1. Identifying non-obvious patterns in data (based on the results of a blood test, what is the likelihood that this patient has a specific type of cancer?)
2. Predicting future outcomes based on historical data (is this consumer, with these specific attributes, likely to default on their debt?)
3. Extracting metadata from unstructured data (in this image of a person, are they smiling? Are they wearing sunglasses? Do they have a beard?)

Many types of ML approaches require large amounts of raw data to train the **machine learning model** (teaching the model about patterns in data). As such, data scientists can be significant consumers of data in modern organizations.

AWS tools used by data scientists to work with data

Data scientists use a wide variety of tools for many different purposes, such as tools for developing ML models, tools for fine-tuning those models, and tools for preparing data to train ML models.

Amazon SageMaker is a suite of tools that helps data scientists and developers with the many different steps required to build, train, and deploy ML models. In this section, we will only focus on the tools that are used in data preparation, but in *Chapter 13*, *Enabling Artificial Intelligence and Machine Learning*, we will do a deeper dive into some of the other AWS tools related to ML and AI.

SageMaker Ground Truth

Most ML models today rely on training the model using labeled data. That is, a dataset that includes the attribute that we are trying to predict is available to help train our model.

Let's use an example of a data scientist named Luna who is looking to create an ML model to identify if an image is of a dog or a cat. To train the model, Luna would need loads of pictures of dogs and cats and would need each image to be labeled to indicate whether it is a picture of a dog or a cat. Once Luna has this labeled dataset, she could train her ML model to recognize both dogs and cats.

For our example, let's imagine that Luna was able to acquire a set of 10,000 images of dogs and cats, but the images are unlabeled, which means they cannot be used to train the model. And it would take weeks for Luna to go through the 10,000 images on her own to label each one correctly.

Luckily, Luna has heard about **SageMaker Ground Truth**, a fully managed service for labeling datasets. Ground Truth uses its own ML model to automatically label datasets, and when it comes across data that it cannot confidently label, it can route that data to a team of human data labelers to be manually labeled. You can route data to either your pre-selected team of data labelers, or make use of the over 500,000 independent contractors that are part of the **Amazon Mechanical Turk** program and have them label the data according to your instructions.

Using Amazon SageMaker Ground Truth, Luna can quickly and accurately get her 10,000 images of dogs and cats labeled, ready to help train her ML model.

SageMaker Data Wrangler

It has been estimated that data scientists can spend up to 70% of their time cleaning and preparing raw data to be used to train ML models. To simplify and speed up this process, **Amazon SageMaker Data Wrangler** can be used to aggregate and prepare data for machine learning purposes.

In most organizations, there will be formal datasets that data engineering teams have prepared for consumption by the organization. However, the specific data that a data scientist needs for training a specific model may not be available in this repository, may not be in the required format, or may not contain the granular level of data that is needed. To best enable data scientists to be self-sufficient without needing to depend on other teams, many organizations enable their data science teams to directly ingest and process raw data.

Data Wrangler supports directly ingesting data from sources, including Amazon S3, Athena, Redshift, as well as the Snowflake data warehouse. Once imported, a data scientist can use the SageMaker Studio interface to transform the data, selecting from a library of over 300 built-in data transformations. Data Wrangler also supports writing custom transformations using PySpark and popular Python libraries such as pandas.

Once a Data Wrangler flow has been created in the **SageMaker Studio visual interface**, a user can export the Data Wrangler flow into a **Jupyter notebook** and run it as a Data Wrangler job, or even export the code as Python code and run it elsewhere.

SageMaker Clarify

SageMaker Clarify is a tool for examining raw data to identify potential bias in data that is going to be used to train ML models. For example, let's say that you were developing a new ML model to detect credit risk for new customers. If your proposed training dataset contains data mostly on middle-aged people, then the resulting ML model may be less accurate when making predictions for younger or older people.

SageMaker Clarify has been integrated with **SageMaker Data Wrangler**, enabling users to evaluate their datasets for potential bias as part of the data preparation process. Users can specify the attributes that they want to evaluate for bias (such as gender or age) and SageMaker Clarify will use several built-in algorithms to detect poten-

tial bias. SageMaker Clarify also provides a visual report with details on the measurements and potential bias identified.

So far, we have had a look at several types of data consumers that are common in organizations, as well as the types of tools that these data consumers may use. Now, we will move on to this chapter's hands-on exercise – creating a simple data transformation using AWS Glue DataBrew.

Hands-on – creating data transformations with AWS Glue DataBrew

In *Chapter 7, Transforming Data to Optimize for Analytics*, we used AWS Glue Studio to create a data transformation job that took in multiple sources to create a new table. In this chapter, we discussed how **AWS Glue DataBrew** is a popular service for data analysts, so we'll now make use of Glue DataBrew to transform a dataset.

Differences between AWS Glue Studio and AWS Glue DataBrew

Both AWS Glue Studio and AWS Glue DataBrew provide a visual interface for designing transformations, and in many use cases, either tool could be used to achieve the same outcome. However, Glue Studio generates Spark code that can be further refined in a code editor and can be run in any compatible environment. Glue DataBrew does not generate code that can be further refined, although Glue DataBrew recipes can also be run from a Glue Studio job. Glue Studio has fewer built-in transforms, and the transforms it does include are generally aimed at data engineers. Glue DataBrew has over 250 built-in transforms, and these are generally aimed at data

analysts. For more information on running Glue DataBrew recipes from within Glue Studio, see the following blog post: https://aws.amazon.com/blogs/big-data/use-aws-glue-databrew-recipes-in-your-aws-glue-studio-visual-etl-jobs/.

In this hands-on task, we will be playing the role of a data analyst who has been tasked with creating a mailing list that can be used to send marketing material to the customers of our now-closed video store, to make them aware that our catalog of movies is now available for streaming.

Configuring new datasets for AWS Glue DataBrew

To start with, we're going to access the Glue DataBrew console and connect to two existing S3-based data sources (the *customer* and *address* tables that we ingested from our MySQL database in *Chapter 6*, *Ingesting Batch and Streaming Data*):

1. Log in to the **AWS Management Console** and access the **Glue DataBrew** service at https://console.aws.amazon.com/databrew.
2. From the left-hand side menu, click on **Datasets**.
3. Click on **Connect new dataset**.
4. Provide a **Dataset name** for the customer table (such as `customer-dataset`).
5. In the **Connect to new dataset** section of the window, click on **Data Catalog S3 tables** on the left-hand side. Then, click on **sakila** from the list of Glue databases.

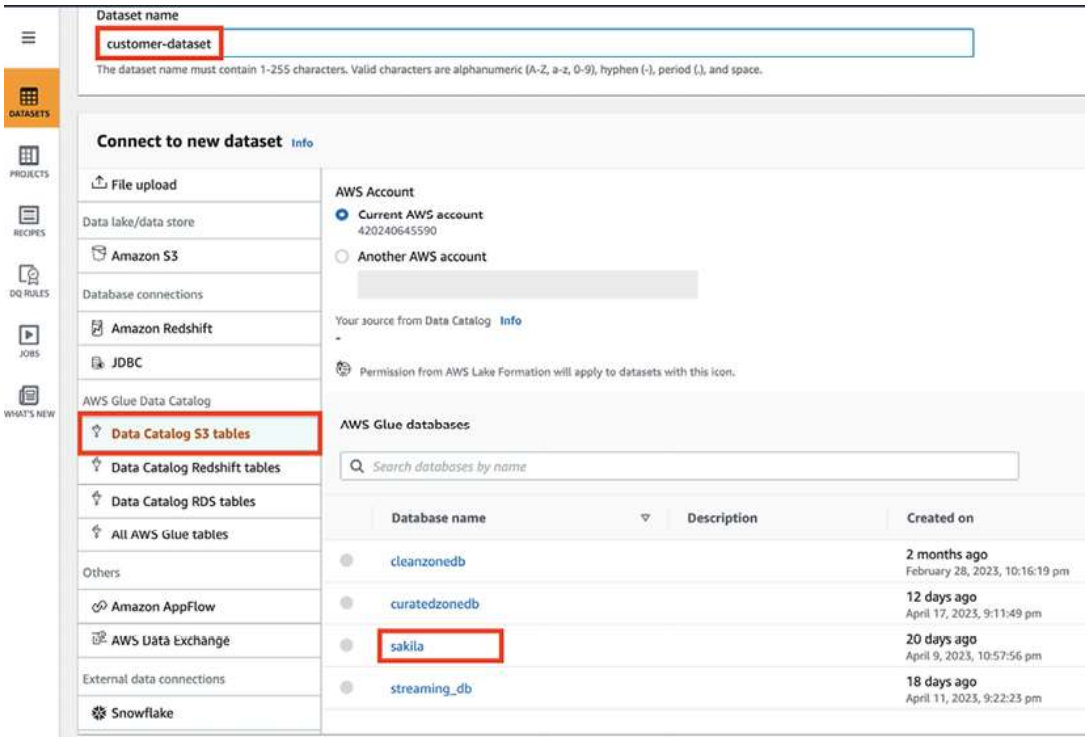

Figure 8.3: Glue DataBrew – Dataset console

6. From the list of tables, click the selector for the `customer` table, and then click **Create dataset** at the bottom right.
7. Repeat *Steps 1–6*, but this time, name the dataset `address-dataset` , select **Data Catalog S3 tables** and **sakila** again, but select the `address` table, and then **Create dataset**.

Now that we have configured the two datasets we plan to use, we will start creating the transform steps in a new DataBrew project.

Creating a new Glue DataBrew project

Now, let's create a new Glue DataBrew project where we can join our customer and address tables, and then clean the dataset:

1. In the AWS Glue DataBrew console, click on **PROJECTS** from the left-hand side menu. Then, click **Create project**.
2. For **Project name**, provide a name (such as `customer-mailing-list`).

3. Under **Recipe details**, leave the default of **Create new recipe** as is.
4. Under **Select a dataset**, select **customer-dataset**:

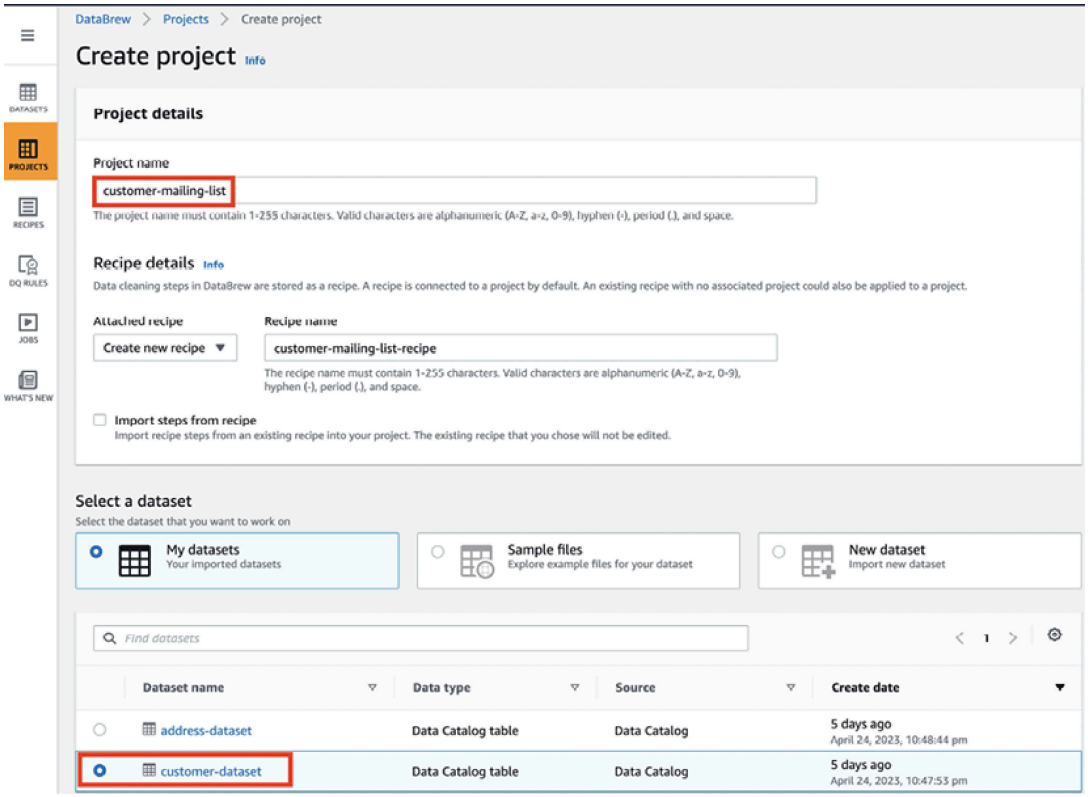

Figure 8.4: Creating a new Glue DataBrew project (1)

5. Under **Permissions**, from the drop-down list, select **Create new IAM role**.
6. For **New IAM role suffix**, provide a suitable suffix, such as `dataengbook` .
7. At the bottom right, click on **Create project**:

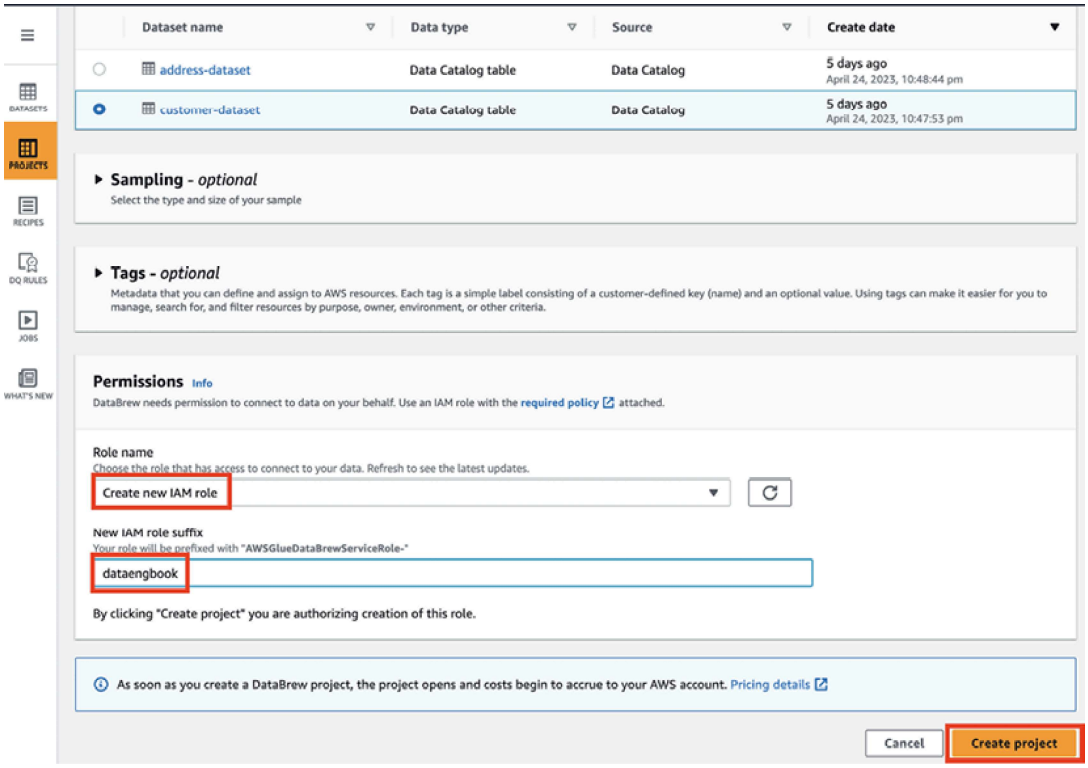

Figure 8.5: Creating a new Glue DataBrew project (2)

Note that there are session costs associated with Glue DataBrew projects ($1.00 per 30-minute session). However, at the time of writing, AWS is offering the first 40 sessions at no charge to new Glue DataBrew customers. For the current pricing, see https://aws.amazon.com/glue/pricing/.

Building your Glue DataBrew recipe

We can now use the interactive Glue DataBrew project session to build out a recipe for our transformation (a recipe is the steps that are taken to transform our data). Note that it may take a few minutes before the session is provisioned and ready.

In the interactive project session window, as shown in the following screenshot, we can see a sample of our customer table data and a panel to the right that allows us to build our recipe:

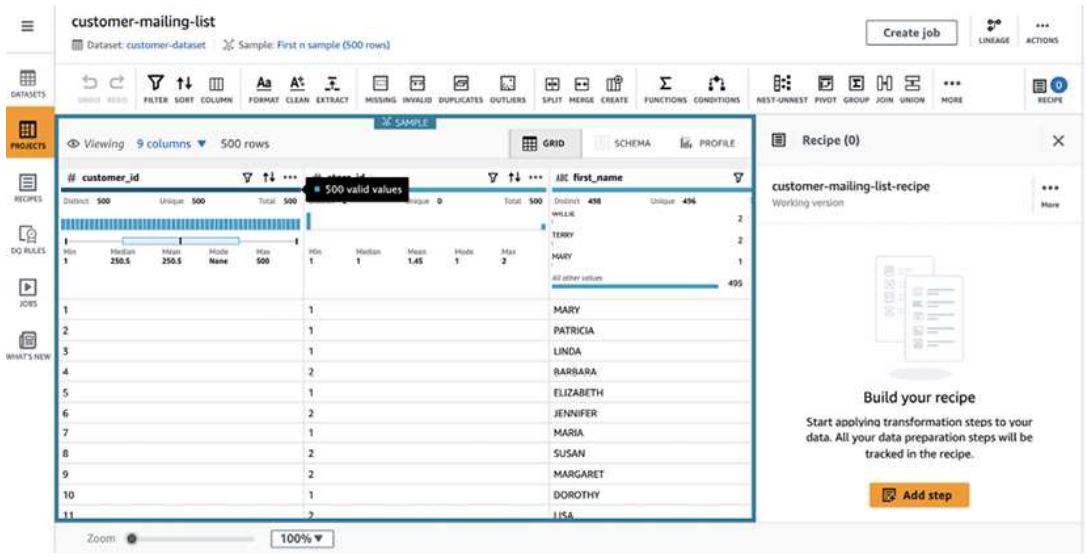

Figure 8.6: AWS Glue DataBrew interactive project session

For our recipe, we want to join this data with our address table, and then make the following changes to the dataset to create a mailing list for our marketing team:

- Change the `first_name` and `last_name` columns to capital case.
- Change the email addresses so that they're all in lowercase.

Follow these steps to create the recipe. The first steps will join our data with the address table:

1. Click on **Add step** in the recipe panel on the right-hand side of the console.
2. Scroll down through the list of transformations and select **Join multiple datasets**.
3. From the **Select dataset** dropdown, select **address-dataset**. Dataset metadata, as well as a sample of the dataset, will be displayed. Click on **Next** at the bottom right.
4. For **Select join type**, select **Left join**. This takes all the rows in our left-hand table (the customer table) and joins each row with the matching row in the address table, based on the join keys we specify.

5. For **Join keys**, for **Table A**, select `address_id`. For **Table B**, also select `address_id`.
6. Under **Column list**, deselect all the columns, and then select only the following columns (these will be the only columns that our marketing team needs for the mailing list):
    1. **Table A**, `customer_id`
    2. **Table A**, `first_name`
    3. **Table A**, `last_name`
    4. **Table A**, `email`
    5. **Table B**, `address`
    6. **Table B**, `district`
    7. **Table B**, `postal_code`
7. Click **Finish**.

We will now see a preview of our new table, with the customer and address tables joined, and only the columns selected previously showing.

You may notice that our customer list includes addresses from many different countries (look at some of the entries under the `district` column), and yet we don't have a column for the country. This is because our original data source (a MySQL database) was highly normalized. The address table has a `city_id` field, and we could have included that and then joined our new dataset with the city table to include the city name and `country_id` fields. However, we would need to have joined that dataset with the country table (joining on the `country_id` column) to get the country name. We will not be covering those steps here, but feel free to give that a try on your own.

All the first names and last names were captured in all uppercase in the original data source (MySQL), so let's transform these into capital case, and transform the email address into all lowercase.

1. In the **Recipe** panel, click on **Add step icon** next to **Applied steps**.

2. From the list of transforms, scroll down and select the **FORMAT / Change to capital case** transform.
3. For **Source column**, select the `first_name` column. Ensure that **Format column to** has **Capital case** selected and then click **Apply**.
4. Repeat *Steps 1–3*, but this time select the `last_name` column as **Source column**.
5. Repeat *Steps 1–3*, but this time select the **FORMAT / Change to lowercase** transform and select the `email` column as the **Source** column, and then click **Apply**.

Your Glue DataBrew recipe should look as follows:

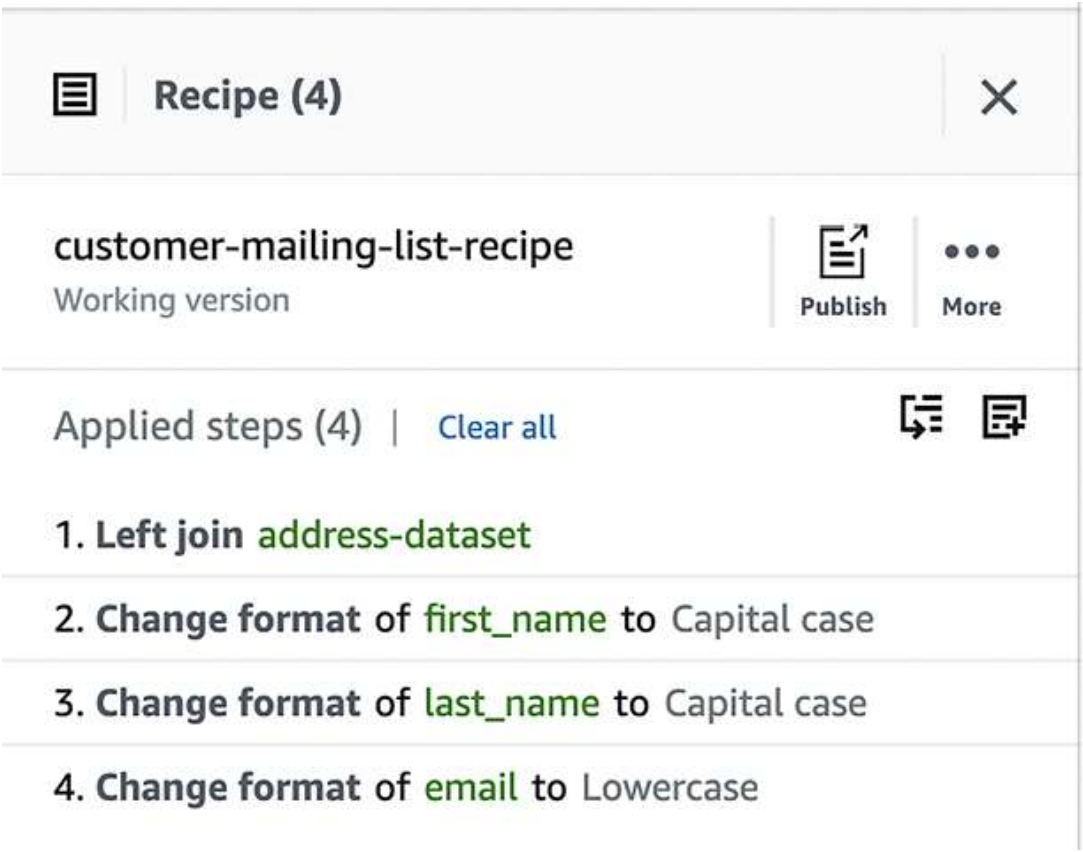

Figure 8.7: Completed Glue DataBrew recipe

With that, we have created our recipe and have been able to preview the results of our transform. Our final step will be to run our recipe in a Glue DataBrew job and write out the results to Amazon S3 so that we can provide the mailing list file to our marketing team.

Creating a Glue DataBrew job

In this final section of our hands-on activity, we will run our recipe in a job and write the results of our transform to a file in Amazon S3:

1. In the AWS Glue DataBrew console, click on **Jobs** from the left-hand side menu. Then, click **Create job**.
2. For **Job name**, provide a name for your job (such as `mailing-list-job`).
3. For **Job input**, select **Project**, and then select your `customer-mailing-list` project.
4. For **Job output settings**, leave the default settings as is (output to Amazon S3, with CSV set as the file type, the delimiter as a comma, and no compression).
5. For **S3 location**, select a location (such as `s3://dataeng-clean-zone-<initial>/mailing-list` but changing the initials to match your unique bucket name).
6. For **Permissions**, select **Create new IAM role** and provide a suffix (such as `mailing-list-job`). By having Glue DataBrew create a new role for this job, DataBrew will automatically provide write access to the location you specified for S3 output.
7. Click **Create and run job**.
   When the job finishes running, the **Job run history** screen will be displayed, showing the status of the job:

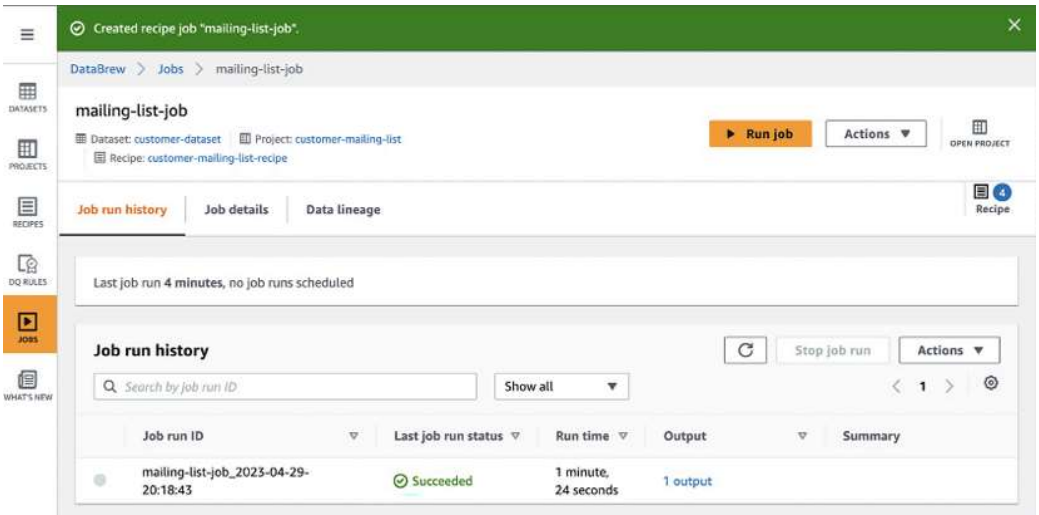

Figure 8.8: Job run history screen showing the job's status

8. Click on **1 output** in the **Output** column to view the S3 destination that you selected for this job. Click on **S3 destination path** to open a new browser tab showing the output's location in the S3 console. Download the CSV file and open it with a text editor or spreadsheet application to verify the results.

In this hands-on exercise, you created a new Glue DataBrew job that joined two tables (customer and address). You then ran various transforms on the dataset to format the columns as needed by the marketing team and created a new CSV output file in Amazon S3.

Summary

In this chapter, we explored a variety of data consumers that you are likely to find in most organizations, including business users, data analysts, and data scientists. We briefly examined their roles and then looked at the types of AWS services that each of them is likely to use to work with data.

In the hands-on section of this chapter, we took on the role of a data analyst, tasked with creating a mailing list for the marketing department. We used data that had been imported from a MySQL database into S3 in a previous chapter, joined two of the tables from that database, and transformed the data in some of the columns. Then, we wrote the newly transformed dataset out to Amazon S3 as a CSV file.

In the next chapter, *Loading Data into a Data Mart*, we will look at how data from a data lake can be loaded into a data warehouse, such as Amazon Redshift.

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