Appendix B. Technology Certifications

By Noah Gift

The term “MLOps” directly implies a firm connection with IT, Operations, and other traditional technology disciplines via the phrase “Ops.” Technology certification has historically played an essential role in validating the skills of industry professionals. The salary for certified professionals is impressive. According to Zip Recruiter, in February 2021, an AWS Solutions Architect’s average wage was $155k.

One way to think about this topic is to consider the idea of a “Triple Threat.” In basketball this means a player that is so well-rounded they have achieved 10+ rebounds, 10+ assists, and 10+ points in a basketball game. You can apply this same approach to an MLOps career. Have a portfolio of examples of your work, get certified, and have either work experience or a relevant degree.

AWS Certifications

Let’s cover some AWS certification options.

AWS Cloud Practitioner and AWS Solutions Architect

I recommend certifications for MLOps specialists on the AWS Cloud. The AWS Cloud Practitioner is a more gentle introduction to the AWS certification world, similar to the AWS Solutions Architect certification. I have often taught this certification to students of many different types: Data Science Master’s students, nontechnical business professionals, and current IT professionals. Here are some of the more common questions and the answers related to both certifications, with a particular eye toward a person with a machine learning background aiming to pass the exam.

Even if you don't get the AWS Certification, these questions are critical for the MLOps practitioner and are worth testing your knowledge level.

> *Q: I am having trouble connecting RDS and sharing connections with a group of people. Is there a more straightforward method?*
>
> > *A*: You may find it more straightforward to use AWS Cloud9 as a development environment to connect to RDS. You can see a walk-through on Amazon.
>
> *Q: The cloud services model is confusing. What is PaaS, and how is it different from other models?*
>
> > *A*: One way to think about cloud offerings is to compare them to the food industry. You can purchase food in bulk at a store like Costco. It has a considerable scale, and it can pass the purchase price discounts down to the customer. As a customer, though, you may also need to take that food back to your home, prepare it, and cook it. This situation is similar to IaaS.
> >
> > Now let's look at a service like Grubhub or Uber Eats. Not only do you not have to drive to the store to pick up the food, but it has been prepared, cooked, and delivered for you. This situation is similar to PaaS. All you have to do is eat the food.
> >
> > If you look at PaaS (platform as a service), what it means is as the developer, you can focus on the business logic. As a result, many of the complexities of software engineering go away. An excellent example of two early PaaS services is Heroku and Google App Engine. A perfect example of a PaaS on AWS is AWS SageMaker. It solves many of the infrastructure issues involved with creating and deploying machine learning, including distributed training and serving predictions.
>
> *Q: What is the exact definition of an edge location? It isn't apparent.*
>
> > *A*: An AWS edge location is a physical location in the world where a server lives. Edge locations are different from data centers because they serve a more narrow purpose. The closer a user of the content is to the server's physical location, the lower the request's latency.

This situation is critical in content delivery like streaming videos and music and also for playing games. The most commonly referred to edge service on AWS is CloudFront. CloudFront is a CDN (Content Delivery Network). Cached or copies of the same movie file live in these locations worldwide via the CDN. This situation allows users to all have a great experience streaming this content.

Other services that use edge locations include Amazon Route 53, AWS Shield, AWS Web Application Firewall, and Lambda@Edge.

Q: What if one of the data centers in an availability zone (AZ) is affected by fire? How are data centers related to each other in terms of a natural or human-made disaster? How should a system be architected for data replication?

A: As part of the shared security model, Amazon is responsible for the cloud, and the customer is responsible for what is in the cloud. This situation means that data is safe against catastrophic unplanned failures like fires. Additionally, if there is an outage, the data may be unavailable during the outage in that region but restores eventually.

As an architect, it is the customer's responsibility to take advantage of multi-AZ architectures. An excellent example of this is Amazon RDS multi-AZ configuration. If an outage occurs in one region, the secondary failover database will already have the data replicated and handle the request.

Q: What is HA?

A: An HA (Highly Available) service builds with availability in mind. This situation means that failure is an expectation, and the design supports redundancy of data and services. An excellent example of an HA service is Amazon RDS. In addition, the RDS Multi-AZ design supports minimal interruption in service by allowing multiple versions of the database replication across availability zones.

Q: How do you decide between spot and on-demand?

A: Both spot and on-demand instances are billed a fixed amount for the first minute and then billed by the second. Spot instances are the most cost-effective because they can provide up to 90% savings. Use Spot instances when it doesn't matter when a task runs or is interrupted. In practice, this creates a critical use case for a spot instance. Here are some examples:

- Experimenting with an AWS service
- Training deep learning or machine learning jobs
- Scaling out a web service or other service in combination with on-demand instances

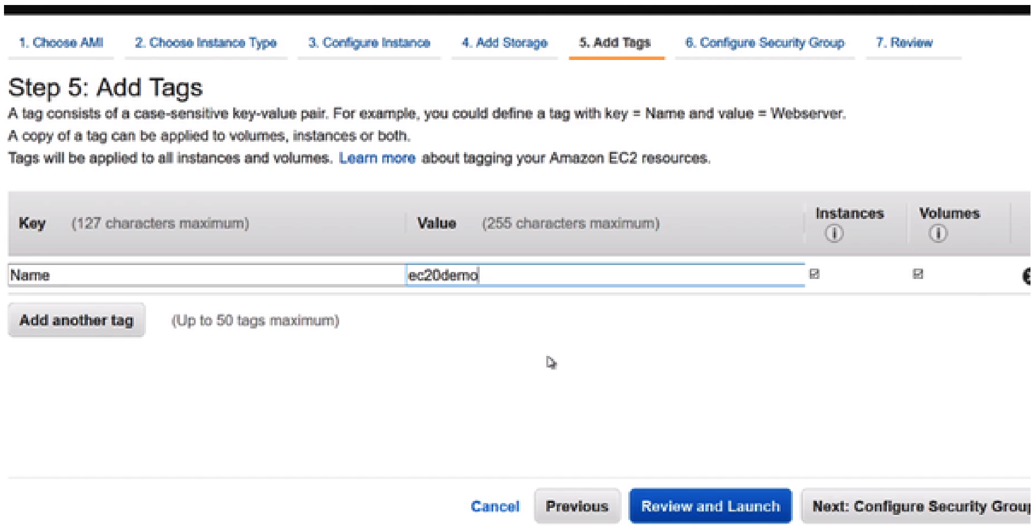
On-demand instances work when a workload runs in a steady state. So, for example, a web service in production wouldn't want only to use spot instances. Instead, it could start with on-demand cases, and when the usage of the service calculates (i.e., 2 c4.large instances), then reserved instances should be bought.

Q: How does spot hibernation work?

A: There are a few reasons for spot instance interruptions, including price (bid higher than maximum price), capacity (there are not enough spot instances unused), and constraints (i.e., the Availability Zone target size is too large). To hibernate, it must have an EBS root volume.

Q: What is a tag for in EC2?

A: In Figure B-1, an EC2 instance has a tag applied to it. This tag can group types of instances into a logical group like "web servers."



Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. Learn more about tagging your Amazon EC2 resources.

Key (127 characters maximum)	Value (255 characters maximum)	Instances	Volumes
Name	ec20demo	☑	☑

(Up to 50 tags maximum)

Figure B-1. EC2 tags

The main reason to use a tag for an EC2 resource is to attach metadata to a group of machines. Here is one scenario. Let's say 25 EC2 instances are running a shopping website, and they have no tags. Later a user spins up another 25 EC2 instances to temporarily do a task like training a machine learning model. It may be challenging in the console to determine which machines are temporary (and can be deleted) and the production machines.

Instead of guessing about machines' roles, it is best to assign tags to allow a user to identify a role quickly. This role could be: `Key="role"`, `Value="ml"` or it could be `Key="role"`, `Value="web"`. In the EC2 console, a user can query by tag. This process then allows for bulk operations like terminating instances. Tags can also play an important role in analyzing cost. If machine roles contain tags, then cost reports could determine if certain machine types are too expensive or using too many resources. You can read the official tag documentation on Amazon.

Q: What is PuTTY?

A: In Figure B-2, the PuTTY SSH tool allows for remote console access from Windows to a Linux virtual machine.

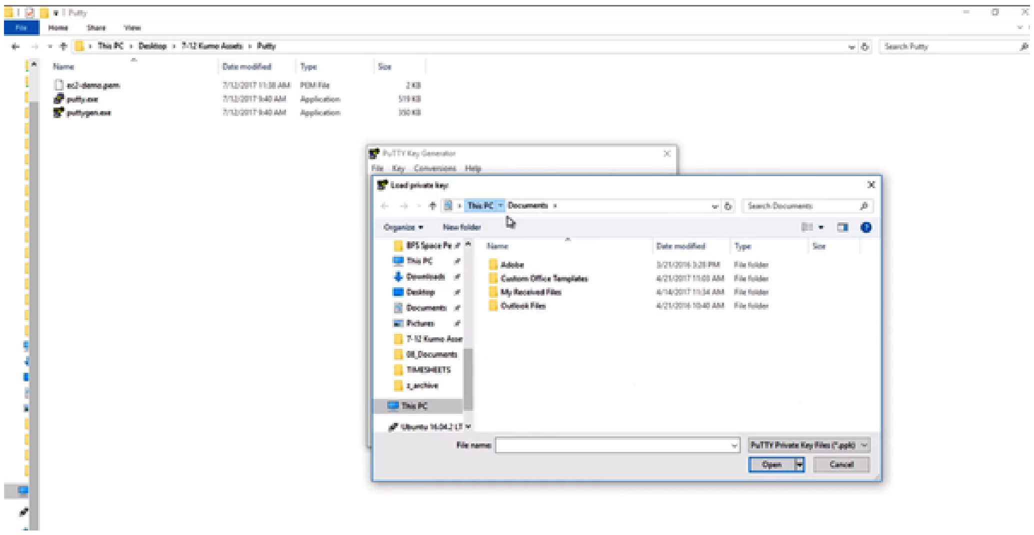


Figure B-2. PuTTY

PuTTY is a free SSH client used on the Windows operating system. MacOS and Linux have built-in support for SSH. What is SSH? It is a cryptographic network protocol for performing network operations. SSH is for logging in to a remote machine and managing the device via terminal commands.

You can read the official PuTTY documentation here.

Q: How is Lightsail different from EC2 or other services?

A: Lightsail is a PaaS or platform as a service. This platform means that a developer solely needs to worry about configuring and developing the WordPress application. EC2 is lower level and is called IaaS (infrastructure as a service). There is a spectrum with cloud computing where lower-level services are provided just like bulk ingredients at Costco. Those bulk ingredients create meals, but it requires skill. Likewise, a person can order meals to get delivered to their house. These meals are more expensive but need little expertise. PaaS is similar; the user pays more for higher-level services.

Other PaaS solutions (outside of AWS) are Heroku and Google App Engine. You can read more about cloud services types in the "Cloud Computing" chapter of *Python for DevOps* (O'Reilly).

Q: I read about a use case for the AMI that has the following description: "use it to copy that to a fleet of machines (deep learning cluster)." What *does a fleet of machine mean?*

> *A*: A fleet works the same way a rental car company works. When you ask for a car reservation, they will ask you to pick a group: compact, sedan, luxury, or truck. There is no guarantee of a specific model, only of a particular group. Likewise, because spot instances are an open market, it is possible that a particular machine, say C3.8XLarge is not available, but a similar combination is. You can request a group of resources identical in CPU, memory, and network capabilities by selecting a fleet. You can read more about EC2 Fleet on Amazon's blog.

Q: What does spikey mean for the sizing of on-demand instances?

> *A*: A "spikey" workload could be a website that suddenly gets 10X the traffic. Let's say this website sells products. It is usually a flat amount of traffic during the year, but around December, the traffic spikes to 10X the usual traffic. This scenario would be a fair use case for "on-demand" instances to scale out to meet this requirement. The expected traffic pattern should use reserved instances, but for the spike, this should use on-demand. You can read more about spikey traffic and reserved instances on Amazon's blog.

Q: What does the "SUBSECOND" mean for one of AWS Lambda's advantages?

> *A*: This means that you can design an efficient service and only incur charges for the duration of your request at 100ms intervals. This situation is different from an EC2 instance where you are billed for a continually running instance every second. With a Lambda function, you can design an event-based workflow where a lambda runs only in response to events. A good analogy would be a traditional light that turns off and on. It is easy to use more electricity because the light has a manual switch. A more efficient approach is motion-detection lighting. The lighting will turn off and on according to motion. This approach is similar to AWS Lambda; in response to events, it turns on, performs a task, and then exits. You can read more about Lambda in Amazon's documentation. You also can build a Python AWS Lambda project on GitHub.

Q: For AWS S3, there are several storage classes. Does the IA (Infrequent Access) tier of storage include both Standard IA and one-zone IA, or are there more types? I see only standard and one-zone in the section of INFREQUENT ACCESS on the AWS website.

> *A*: There are two types of IA (Infrequent Access). Standard IA, which store in three AZ (Availability Zones) and One Zone. A key difference in the One Zone is availability. It has 99.5% availability, less available than both three-zone IA and Standard. The lower cost reflects this diminished availability.

Q: How does Elastic File System (EFS) work?

> *A*: EFS conceptually works similar to Google Drive or Dropbox. You can create a Dropbox account and share data with multiple computers or friends. EFS works in a very similar way. The same filesystem is available to machines that mount it. This process is very different than EBS (Elastic Block Storage), which belongs to one instance at a time.

Q: For ELB's use case, I don't understand these two use cases: 1) what's "the single point of access" mean? Is it saying, if you can control your traffic by entering through one port or server, then it's more secure? 2) what does "decoupling application environment" mean?

> *A*: Let's look at a website as an example. The website will be running on port 443, which is the port for HTTPS traffic. This site could be *https://example.com*. The ELB is the only resource exposed to the outside world. When a web browser connects to *https://example.com*, it communicates only to the ELB. At the same time, the ELB will ask the web servers behind it for the information. It then sends that information back to the web browser.
>
> What is an analogy in the real world? It would be like a bank teller in a drive-through. You drive up to the window but connect only to the bank teller. Inside the bank, many people are working, yet you are interacting with only one person. You can read blog posts about ELB on the AWS blog.

Q: Why is the use case for ELB the same as the Classic Load Balancer?

A: They share the same traits: access through a single point of entry, decoupled application environment; they provide high availability and fault tolerance, and increase elasticity and scalability.

Elastic Load Balancing refers to a category of load balancers. There is Application Load Balancer, Network Load Balancer, and Classic Load Balancer. At a high level, the Classic Load Balancer is an older load balancer that has fewer features than the Application Load Balancer. It works in situations where older EC2 instances have been in service.

These are called EC2 classic instances. In a new scenario, an Application Load Balancer would be ideal for something like an HTTP service. You can read blog posts about ELB feature comparisons in Amazon's documentation.

AWS Certified Machine Learning - Specialty

Business schools and data science schools have completely embraced teaching certifications. At UC Davis, I teach students traditional for-credit material, like machine learning and cloud certifications, including both AWS Cloud Practitioner and AWS Certified Machine Learning - Specialty. With AWS, I work closely with many parts of its organization, including AWS Educate, AWS Academy, and AWS ML Hero.

As you might guess, yes, I recommend getting the AWS Machine Learning certification. Knowing many of the people involved in creating the AWS ML certification, and perhaps some influence on its creation, what I like about it is the heavy focus on MLOps. So let's dive into it in more detail.

The recommended candidate has knowledge and experience of at least 1–2 years developing, architecting, or running ML/deep learning workloads. In practice, this means the ability to express the intuition behind basic ML algorithms, perform basic hyperparameter optimization, experience with ML and deep learning frameworks, follow model-training best practices, and follow deployment and operational best practices. In a nutshell, reading this book is an excellent preparation step for getting certified!

The exam structure divides into several domains: Data Engineering, Exploratory Data Analysis, Modeling, and Machine Learning Implementation and Operations. The last section of the exam, in particular, is on what is essentially MLOps. So let's dive into these sections and see how they apply to concepts covered in this book.

Data Engineering

A central component of data engineering, analytics, and machine learning on AWS is the data lake, which also happens to be Amazon S3. Why use a data lake (Figure B-3)? The core reasons are as follows. First, it provides the ability to handle both structured and unstructured data. A data lake also allows both Analytics and ML workloads. Third, you can work on the data without moving it, which can be critical with big data. And finally, its low cost.

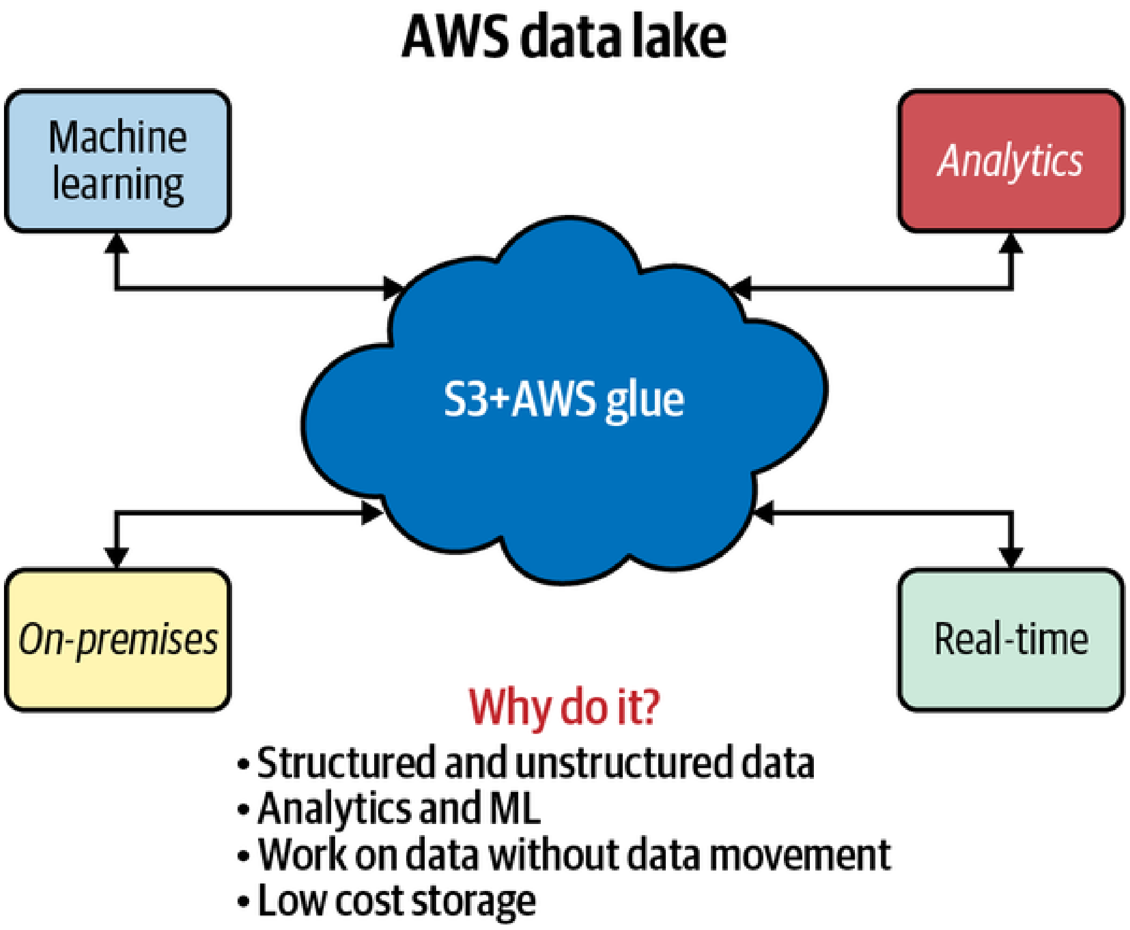


Figure B-3. Data lake

Another important topic on AWS Data Engineering is batch versus streaming data. Let's define streaming data first. Streaming data is typically small data sent from many sources. Examples include log-files, metrics, and time-series data like stock trading information. The leading service on AWS that handles streaming is Kinesis. Here are a few ideal scenarios: time-series analytics problems, real-time dashboards, and real-time metrics.

Batch versus streaming has significant effects on ML pipeline development. There is more control of model training in batch since you can decide when to retrain the ML model. Continuously retraining a model could provide better prediction results but at added complexity. For example, is A/B testing used to test for the accuracy of the new model? Model A/B testing is available in SageMaker endpoints, so this will need to factor into the architecture.

For batch processing, several tools can function as batch processing mechanisms. These include Amazon EMR/Spark, AWS Glue, AWS Athena, AWS SageMaker, and the aptly named AWS Batch service. In particular, for machine learning, AWS Batch solves a unique problem. For example, imagine you wanted to scale thousands of simultaneous and individual k-means clustering jobs. One way to do it would be with AWS Batch. Additionally, you can provide a simple interface to the batch processing system by writing a Python command line tool. Here is a snippet of code that shows what that could look like in practice:

```
@cli.group()
def run():
    """AWS Batch CLI"""

@run.command("submit")
@click.option("--queue", default="queue", help="Batch Queue")
@click.option("--jobname", default="1", help="Name of Job")
@click.option("--jobdef", default="test", help="Job Definition")
@click.option("--cmd", default=["whoami"], help="Container Override Commands")
def submit(queue, jobname, jobdef, cmd):
    """Submit a job to AWS Batch SErvice"""

    result = submit_job(
        job_name=jobname,
        job_queue=queue,
        job_definition=jobdef,
        command=cmd
    )
    click.echo(f"CLI:  Run Job Called {jobname}")
    return result
```

Another critical aspect of handling data engineering is using AWS Lambda to process events. It is important to note that AWS Lambda is a glue that integrates deeply into most AWS services. Therefore, doing AWS-based data engineering most likely will encounter AWS Lambda at some point.

The CTO of AWS believes that a "one size database doesn't fit anyone". What he means by this is clearly described in Figure B-4.

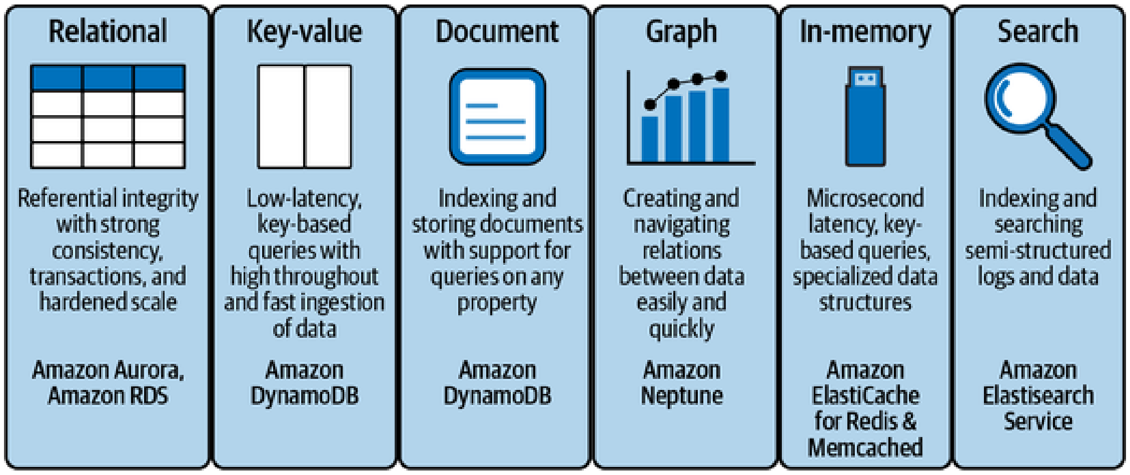

Relational
Referential integrity with strong consistency, transactions, and hardened scale
Amazon Aurora, Amazon RDS
Key-value
Low-latency, key-based queries with high throughout and fast ingestion of data
Amazon DynamoDB
Document
Indexing and storing documents with support for queries on any property
Amazon DynamoDB
Graph
Creating and navigating relations between data easily and quickly
Amazon Neptune
In-memory
Microsecond latency, key-based queries, specialized data structures
Amazon ElastiCache for Redis & Memcached
Search
Indexing and searching semi-structured logs and data
Amazon Elastisearch Service

Figure B-4. One size database

Another way of saying this is to use the best tool for the job. It could be a relational database; in others, it is a key/value database. The following example shows how simple a DynamoDB-based API in Python works. Most of the code is logging.

```
def query_police_department_record_by_guid(guid):
    """Gets one record in the PD table by guid

    In [5]: rec = query_police_department_record_by_guid(
        "7e607b82-9e18-49dc-a9d7-e9628a9147ad"
        )

    In [7]: rec
    Out[7]:
    {'PoliceDepartmentName': 'Hollister',
     'UpdateTime': 'Fri Mar  2 12:43:43 2018',
     'guid': '7e607b82-9e18-49dc-a9d7-e9628a9147ad'}
    """

    db = dynamodb_resource()
    extra_msg = {"region_name": REGION, "aws_service": "dynamodb",
        "police_department_table":POLICE_DEPARTMENTS_TABLE,
        "guid":guid}
    log.info(f"Get PD record by GUID", extra=extra_msg)
    pd_table = db.Table(POLICE_DEPARTMENTS_TABLE)
    response = pd_table.get_item(
        Key={
            'guid': guid
            }
    )
    return response['Item']
```

Three final items to discuss in data engineering are ETL, Data Security, and Data Backup and Recovery. With ETL, the essential services include AWS Glue, Athena, and AWS DataBrew. AWS DataBrew is the newer service, and it solves a necessary step in building production machine learning models, namely automating the messy steps in cleaning data. For example, in Figure B-5, a dataset, baby names, is profiled without writing a single line of code.

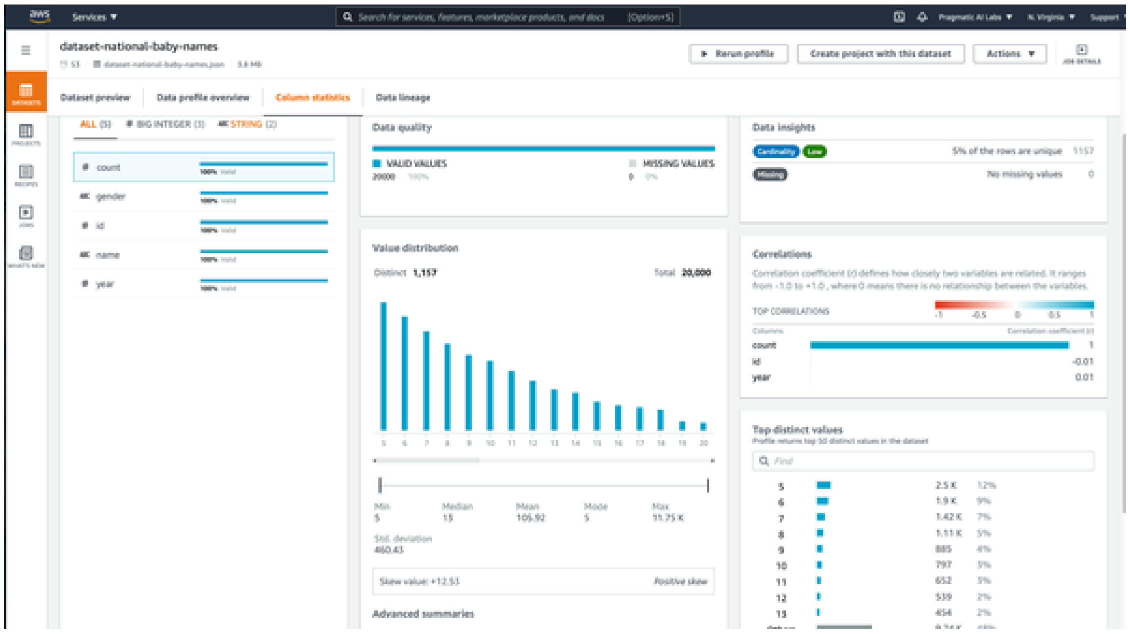


Figure B-5. DataBrew

Later, this same dataset could be a vital part of an MLOps project. One helpful feature is the ability to track the dataset's lineage, where it came from, and what actions involve the dataset. This functionality is available via the "Data lineage" tab (see Figure B-6).

Data governance is a concise way of addressing concerns due to data security and data backup and recovery. AWS, through the KMS (Key Management Service), allows for an integrated encryption strategy. This step is critical because it provides for the implementation of both encryptions at rest and in transit, as well as PLP (Principle of Least Privilege).

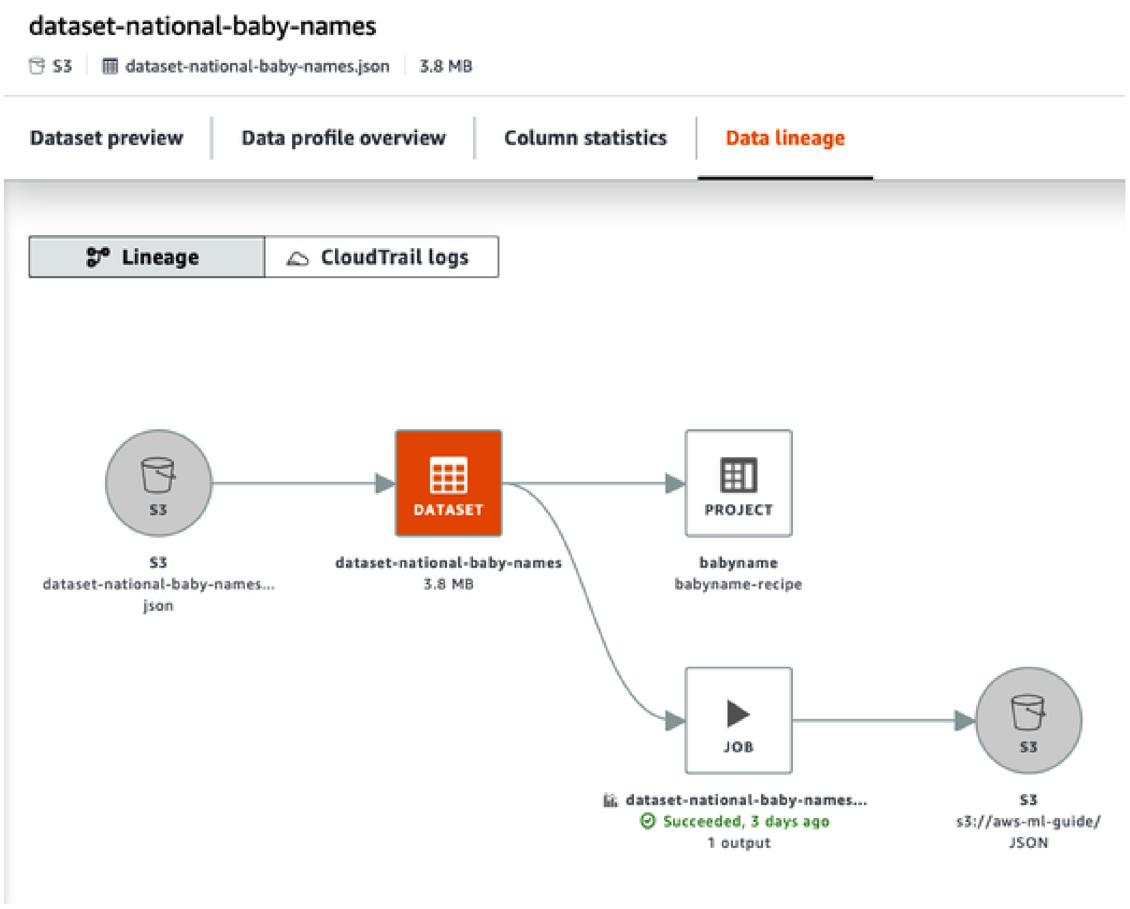


Figure B-6. DataBrew lineage

Another aspect of data security is the ability to log and audit access to data. Regular audit of data access is one way to identify risks and mitigate them. For example, you may flag a user regularly looking at an AWS Bucket that has nothing to do with their job and then realize this poses a significant security hole you need to close.

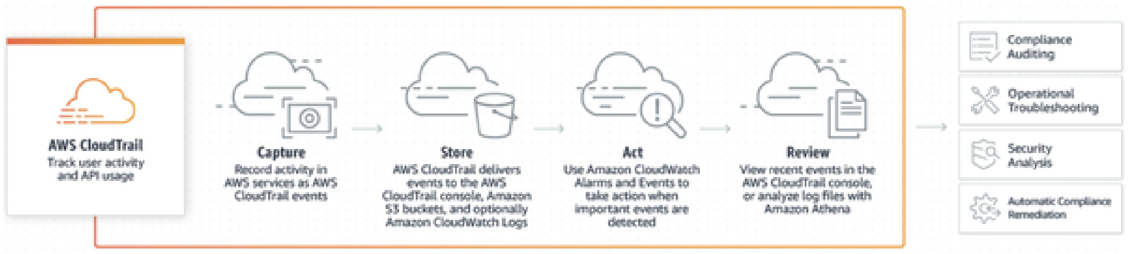

AWS CloudTrail
Track user activity and API usage
Capture
Record activity in AWS services as AWS CloudTrail events
Store
AWS CloudTrail delivers events to the AWS CloudTrail console, Amazon S3 buckets, and optionally Amazon CloudWatch Logs
Act
Use Amazon CloudWatch Alarms and Events to take action when important events are detected
Review
View recent events in the AWS CloudTrail console, or analyze log files with Amazon Athena
Compliance Auditing
Operational Troubleshooting
Security Analysis
Automatic Compliance Remediation

Figure B-7. AWS Cloud-Trail

Finally, data backup and recovery is perhaps one of the most important aspects of data governance. Most AWS services have snapshot capabilities, including RDS, S3, and DynamoDB. Designing a helpful backup, recovery, and lifecyle for data that archives to Amazon Glacier are essential for best practices compliance in data engineering.

Exploratory Data Analysis (EDA)

Before you can do machine learning, first, the data needs exploration. On AWS, several tools can help. These include the DataBrew example from earlier, as well as AWS QuickSight. Let's take a look at what you can accomplish with AWS QuickSight in Figure B-8.

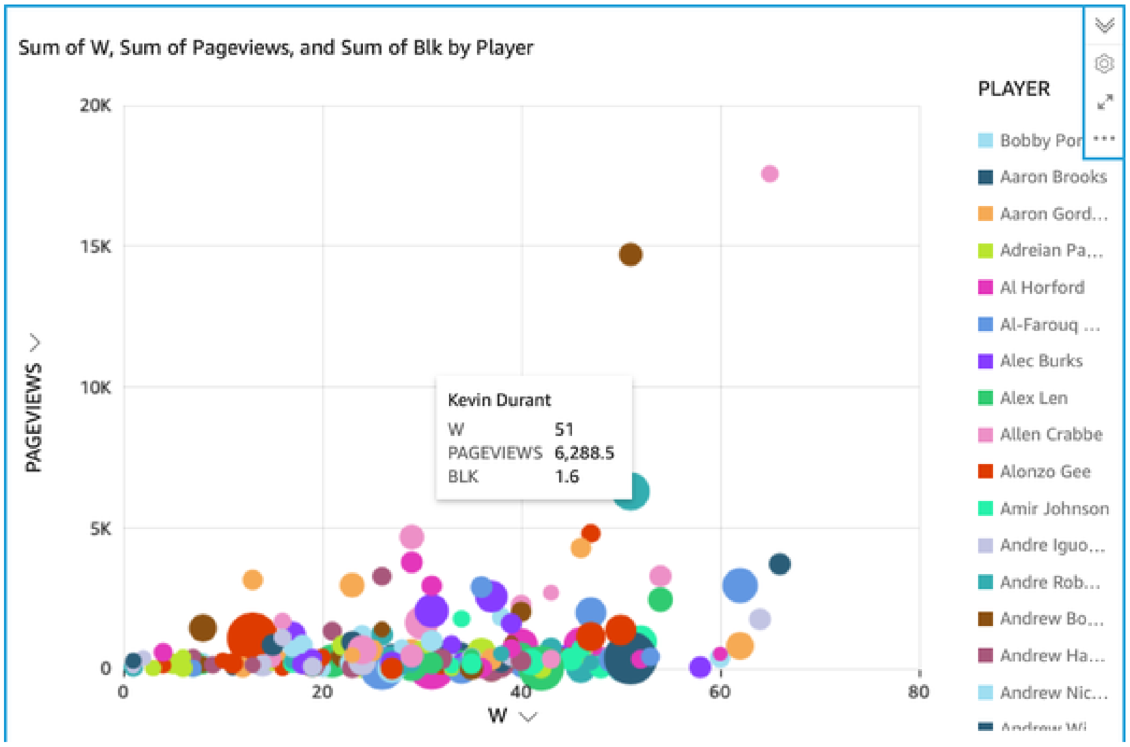


Figure B-8. AWS QuickSight

Notice how this no-code/low-code approach exposes a power-law relationship between wins and popularity on social media, i.e., pageviews from Wikipedia. Fans may gravitate most closely with “winners,” take notice of them, and want to read more information about these players. This initial EDA step could immediately lead toward developing a machine learning model that uses fan behavior to predict which teams are more likely to win the NBA season.

Replicating this chart yourself is straightforward: download the CSV file, tell QuickSight to perform a new analysis, make a new dataset using the CSV file, and then select “create an analysis.”

It is essential to understand the role of EDA in MLOps. EDA helps detect outliers, find hidden patterns (via clustering), see data distributions, and create features. When using clustering, it is essential to remember that data needs to be scaled. Scaling data normalizes the magnitude. For example, if two friends ran “50,” importance is vital to consider. One friend could have run 50 miles, and another could have run 50 feet. They are immensely different things. Without scaling, the results of machine learning distort from the magnitude of one variable or column. The following example shows how scaling looks in practice:

```
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler

scaler = StandardScaler()

print(scaler.fit(numerical_StandardScaler(copy=True,
    with_mean=True, with_std=True)

# output
# [[ 2.15710914 0.13485945 1.6406603 -0.46346815]
```

Another concept in EDA is the idea of preprocessing data. Preprocessing is a broad term that can apply to more than one scenario. For example, machine learning requires data to be numerical, so one form of preprocessing is to encode categorical variables to numerical format.

NOTE

Encoding categorical data is an essential part of machine learning. There are many different types of categorical variables.

- Categorical (Discreet) Variables
  - Finite set of values: {green, red, blue} or {false, true}
- Categorical Types:
  - Ordinal (ordered): {Large, Medium, Small}
  - Nominal (unordered): {green, red, blue}
- Represented as text

Another form of preprocessing is the creation of new features. Let's take this example of NBA players' ages. A plot shows there is a normal distribution of age with the median around 25 years old (Figure B-9):

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv(
  r"https://raw.githubusercontent.com/noahgift/socialpowernba" \
  r"/master/data/nba_2017_players_with_salary_wiki_twitter.csv")

sns.distplot(df.AGE)
plt.legend()
plt.title("NBA Players Ages")
```

Figure B-9. NBA Players Age

We can take this knowledge to create a new feature. The feature can convert age into several categories: Rookie, Prime, Post Prime, and Pre-Retirement. These categories may lead to future insights:

```
def age_brackets (age):
  if age >17 and age <25:
      return 'Rookie'
  if age >25 and age <30:
      return 'Prime'
  if age >30 and age <35:
       return 'Post Prime'
  if age >35 and age <45:
      return 'Pre-Retirement'
```

We can then use these new categories to group the data and figure out each bracket's median salary:

```
df["age_category"] = df["AGE"].apply(age_brackets)
df.groupby("age_category")["SALARY_MILLIONS"].media
```

Notice, there are dramatic differences in the median salary. When players start, they get paid the least, but their salary roughly triples in their prime. Once they retire, there is a dramatic shift as their pay is half what they were compensated in their peak:

```
age_category
Post-Prime 8.550
Pre-Retirement   5.500
Prime            9.515
Rookie           2.940
Name: SALARY_MILLIONS, dtype: float64
```

Machine Learning Implementation and Operations (MLOps)

Let's discuss some of the critical components of MLOps in the concept of the AWS ML Certification exam. The following key concepts are essential to consider when building models:

- Monitoring
- Security
- Retraining Models
- A/B Testing
- TCO (Total Cost of Ownership)

What about MLOps itself? The following are critical to consider:

- Are you using a simple enough model?
- Are you using the data lake or wired directly into production SQL DB?
- Do you have alerts set up for prediction threshold failures?
- Do you have development, stage, and production environments?

Finally, two topics are worth discussing: troubleshooting production deployment and efficiency of ML systems. For a production deployment, these concepts include using CloudWatch, searching CloudWatch logs, alerting on critical events, using auto-scale capabilities, and using enterprise support.

For ML systems' cost and efficiency, both on the exam and in the real world, it is essential to understand the following key concepts:

- Spot Instances (show spot code)
- Proper use of CPU versus GPU resources
- Scaling up and scaling down
- Time to market
- AI API versus "Do it Yourself"

Other Cloud Certifications

In addition to AWS, Azure and GCP have notable certifications.

Azure Data Scientist and AI Engineer

Azure has a few certifications that are worth looking at, including the Azure Associate Data Scientist and Azure AI Engineer Associate. These certifications are broken down into three levels (in order of expertise): fundamentals, associate, and expert. In addition, there are several *learning paths* related to Azure's AI platform that are closely related to MLOps:

- The get-started guide
- Create machine learning models
- Create no-code predictive models with Azure Machine Learning
- Build AI solutions with Azure Machine Learning

A good starting point is to go through the training guide, which has a compelling list of journeys (for example, Data and AI professionals) and explicit certifications like Azure AI Fundamentals. These *journeys* allow you to decide the best strategy to pursue the objective that works best.

Additionally, there are free (or mostly free) resources to help you get started on Azure if you are student or a faculty member. The offerings tend to change with time, but you can sign up without a credit card and get started right away as of this writing. If you are an educator, there are additional offers and resources available to you that can also be helpful.

GCP

A few notable certifications for MLOps practitioners include the Professional Machine Learning Engineer and Professional Cloud Architect. Finally, a very specific MLOps-related certification is the TensorFlow Developer Certificate. It allows you to demonstrate your proficiency in using TensorFlow to solve deep learning and ML problems.

SQL-Related Certifications

Minimal knowledge of SQL is necessary to be successful with MLOps. Later though, it is recommended to go deeper on SQL and master it both theoretically and in an applied fashion. Here some recommended resources:

- Databricks Certified Associate Developer for Apache Spark 3.0
  - Study Material: Databricks website, and O'Reilly Learning Platform
    - O'Reilly Learning Platform: Learning Spark
    - O'Reilly Learning Platform: Spark: The Definitive Guide
- Microsoft Certified: Azure Data Fundamentals
  - Study Material: Coursera, Microsoft Learn and O'Reilly Learning Platform
    - Coursera Course Microsoft Azure DP-900 Data Fundamentals Exam Prep
    - O'Reilly Learning Platform: Exam Ref DP-900

- Oracle Database SQL Certified Associate Certification
    - Study Material: O'Reilly Learning Platform and Oracle
        - O'Reilly Learning Platform: OCA Oracle Database SQL Exam Guide
        - Oracle: Oracle Database SQL Certified Associate Certification
- Google Data Analytics Professional
    - Study Material: Coursera and O'Reilly Learning Platform
        - O'Reilly Learning Platform: Data Governance: The Definitive Guide
        - O'Reilly Learning Platform: Data Science on the Google Cloud Platform
        - O'Reilly Learning Platform: Google BigQuery: The Definitive Guide
        - Coursera: Google Data Analytics Professional

You can find additional references on Coursera.