Cost Modeling Data Lakes for Beginners

How to start your journey into data analytics

November 2020

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

Abstract

This whitepaper focuses on helping you understand and address the first challenge of your data lake journey: how much will it cost? Understanding cost considerations can inform your organizational business case and decision-making process when evaluating the value realization outcomes for a data lake project.

This whitepaper discusses the challenges of using traditional methodologies for costing data lake projects. It then outlines an approach that enables you to move at speed, realizing value early on the project cycle.

Introduction

Customers want to realize the value held in the data their organization generates. Common use cases include helping them expedite decision making, publishing data externally to foster innovation, or creating new revenue streams by monetizing the data.

Organizations that successfully generate business value from their data, will outperform their peers. An Aberdeen survey saw organizations who implemented a Data Lake outperforming similar companies by 9% in organic revenue growth. These leaders were able to do new types of analytics like machine learning over new sources like log files, data from click-streams, social media, and internet connected devices stored in the data lake. This helped them to identify, and act upon opportunities for business growth faster by attracting and retaining customers, boosting productivity, proactively maintaining devices, and making informed decisions.

To best realize this value using data lakes, customers need a technology cost breakdown for their budget to build a solution. But without building the solution, they don't know how much it will cost. This is a common paradox that delays many customers from starting their data lake projects.

Customers need a platform that increases agility, lowers cost of experimentation, and provides a technical breadth to support all their use cases, through an innovative platform. Ideally, the platform can rapidly validate or discount solutions against business objectives. This encourages a culture of failing fast, which enables further experimentation to optimize solution matching against business imperatives.

A data lake is a common way to realize these goals. There are many considerations along this journey, such as team structure, data culture, technology stack, governance risk, and compliance.

Costing data lakes requires a different approach than delivering them. Customers must focus on identifying and measuring business value early on so that they can start their projects quickly and demonstrate the value back to the business quickly and incrementally.

What should the business team focus on?

- **Think big –** It's important to create a vision that your organization can rally around to help guide decision making in-line with a common goal. At Amazon, we have a phrase: "Stubborn on vision; flexible on details". This sums up the importance of creating an environment that allows autonomous decision-making, while everyone pulls in the same direction and is flexible about the implementation details.
- **Measure business value** – Without measuring business value, it's hard to justify any expenditure or drive any value from your testing early on in the project.
- **Prototype rapidly** – Focus your energy on driving business outcomes with any experiments you run.
- **Understand what influences costs** – Analytics projects generally have similar stages, ingestion, processing, analytics, and visualization. Each of these stages has key factors that influence the cost.
- **Cost model a small set of experiments** – Your analytics project is a journey. As you expand your knowledge, your capabilities will change. The sooner you can start experimenting, the sooner your knowledge will grow. Build a cost model that covers to smallest amount of work to impact your business outcomes and iterate.
- **Avoid wasting energy building technology stacks** – Building solutions from the ground up is expensive, time-consuming, and very rarely provides any direct value to your organization.

Defining the approach to cost modeling data lakes

IT projects historically have well-defined milestones that make cost modeling a fairly simple process. Selected software is usually a commercial off-the-shelf (COTS) product, which can be costed (including licenses) and based on predictable metrics, such as number of users and number of CPUs.

The effort to implement the software is well within the standard skill sets of the IT department, with plenty of experience in similar deployment models to draw on to get an accurate picture of the implementation time. This will all feed into a large design document that can be used to calculate costs.

Therefore, you can expect to use the same cost modeling you have previously used for an analytics project. The challenge here is that an analytics project often doesn't have a clear end. It's a journey where you explore and prepare data in line, with some aspirational business outcomes. This makes it's difficult to know the following:

- How much data will be ingested
- Where that data will come from
- How much storage or compute capacity will be needed to make sense of that data
- Which services are required to exploit the data to realize and deliver value throughout the business

However, analytics projects provide the radical growth and business opportunities to enable your organization grow and get a competitive edge. Customers who have realized business benefit include the following companies.

- Siemens mobility, a leader in transport solutions, cut their energy costs by 10-15% with analytics projects.
- Kumon, an online talent market for freelancers, managed to increase purchasing conversion by 30% and decrease churn by 40%.
- 3Victors, a leader in travel data analytics, built their business by collecting billions of shopper requests for flights and analyzing them on AWS. This provided actionable real-time insights to travel marketers and revenue managers. Those insights, in turn, enabled 3Victor to improve the customer conversion rate and improve their return on advertising spend.

To experience the business values of analytics project, you must first get started. An on-premises data lake requires a large upfront expenditure, which can be hard to justify when you are unclear on the return on investment.

However, with the AWS Cloud, you can deploy AWS services in minutes, run your experiments, and then shut down the infrastructure, paying only for what you use. With no capacity constraints and a broad array of analytics services, you can run many exploratory experiments concurrently to find the right solution. This, coupled with the ability to turn off your experiments, lowers the cost of experimentation while increasing speed.

Measuring business value

The first project on your data lake journey starts by stating what your business goals are. These are often extracted from your business strategy. Business goals are high-level aspirations that support the business strategy, such as “improve customer intimacy”.

Once these business goals are identified, together with your team write a set of business outcomes that would have a positive impact against these goals. Outcomes are targeted and measurable, such as *reduce cost of customer acquisition*.

Finally, we identify a number of metrics that can be measured to validate the success of the experiments. This helps ensure that the right business value is being achieved.

A good example of this is Hyatt Hotels, a leading global hospitality company. Hyatt wanted to improve customer loyalty, improve the success rate of upsells and add-ons, and better guide the users with accommodation recommendations. This fed directly into their business goal to improve their American Customer Satisfaction Index (ACSI). To achieve this, the team at Hyatt identified a requirement to build personalized connections with their customers.

Some example business outcomes could be:

- Number of return visits per 1000 customers per month
- Number of upsells per 100 customers per month
- Number of add-ons per 100 customers per month
- Number of bookings made per day
- Number of negative customer reviews per weeks

This is only an example of one piece of work a data lake could support. After the initial piece of work is delivered, the team can then iterate. For example, as a by-product of delivering the previous feature, they could have identified important information. For example, perhaps Hyatt discovered that commonly searched for add-on purchases (for example, specific spa treatments) were not offered in the chosen location. Or, they might have discovered that customers were searching for accommodation in areas that Hyatt doesn’t yet have a footprint in.

The team could go on to develop new features and service offerings that would help them deliver a better customer experience or help them make decisions that would

improve their chances of choosing the right location to scale their footprint globally to help them deliver against their business goal.

Establishing an agile delivery process

Once the measurable metrics are captured, teams can start running a number of experiments to assess technology, methodologies, and, most importantly, explore the data to indicate whether a solution to deliver the business goals is possible and, if so, a possible design of that solution.

The breadth of AWS services helps remove the complexity and burden of designing and operationalizing the solution. This enables organizations to rapidly deploy solutions that typically would take months. It also allows organizations to focus on solving the business needs through deriving value from their data.

The AWS serverless services (such as, Amazon Athena, Amazon Kinesis, and AWS Glue) allow for data manipulation and exploration without the need to deploy anything. These services can be consumed immediately in an on-demand basis.

You can also deploy capacity provisioning services, where clusters can be provisioned in a matter of minutes. These clusters are then ready to process customer data for supporting analytic services such as Amazon EMR (Hadoop), Amazon Elasticsearch, Amazon Managed Streaming for Apache Kafka (Amazon MSK), and Amazon Redshift. If the experiment fails, you can shut down the services or just stop using the services to prevent incurring ongoing costs.

This allows you to experiment rapidly. This was the case with Xylem, a leading water technology company. Xylem used AWS to increase innovation, allowing them to support their customer by creating smart technologies that meet the world's water, wastewater, and energy needs.

Building data lakes

Because the aim is to get started on your data lake project, let's break down your experiments into the phases that are typical in data analytics projects:

- Data ingestion
- Processing and transformation

- Analytics
- Visualization, data access, and machine learning

By breaking down the problem into these phases, you reduce the complexity of the overall challenge. This makes the number of variables in each experiment lower, enabling you to get model costs more quickly and accurately.

We recommend that you start your analytics project by implementing the foundation of a data lake. This gives you a good structure to tackle analytics challenges and allow great flexibility for the evolution of the platform.

A data lake is a single store of enterprise data that includes raw copies from various source systems and processed data that is consumed for various analytics and machine learning activities that provide business value.

Choosing the right storage to support a data lake is critical to its success. Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. This means customers of all sizes and industries can use it to store and protect any amount of data.

Amazon S3 provides easy-to-use management features so you can organize your data and configure finely tuned access controls to meet your specific business, organizational, and compliance requirements. Amazon S3 is designed for 99.999999999% (11 9's) of durability, and stores data for millions of applications for companies all around the world.

Data lakes generally support two types of processing: batch and real time. It is common for more advanced users to handle both types of processing within their data lake. However, they often use different tooling to deliver these capabilities. We will explore common architectures for both patterns and discuss how to estimates costs with both.

Batch processing

Batch processing is an efficient way to process large volumes of data. The data being processed is typically not time-critical and is usually processed over minutes, hours, and in some cases, days. Generally, batch processing systems automate the steps of gathering the data, extracting it, processing it, enriching it, and formatting it in a way that can be used by business applications, machine learning applications, or business intelligence reports.

Before we get started, let's look at a common set of services that customers use to build data lakes for processing batch data.

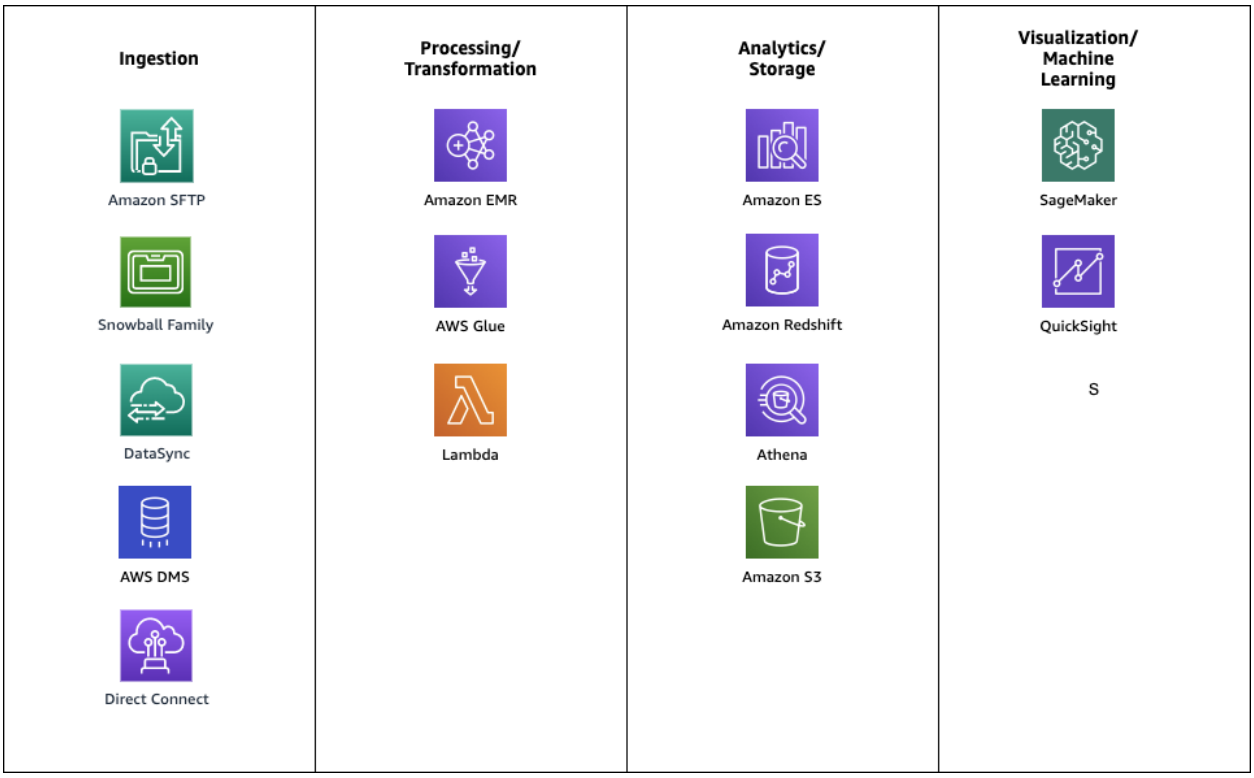


Figure 1 – Common services used to build data lakes for batch data

The following example architecture is relatively common. It uses AWS Glue, Amazon Athena, Amazon S3, and Amazon QuickSight.

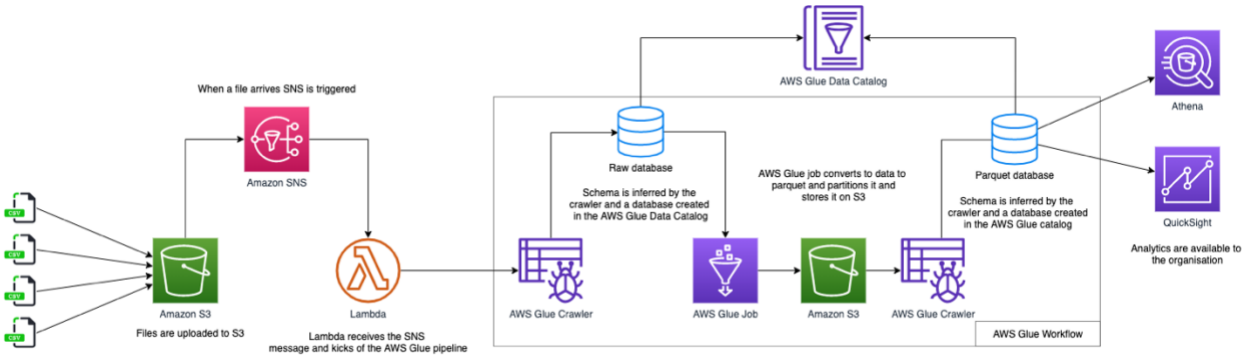


Figure 2 – Example architecture for batch processing

The preceding example shows a typical pipeline to ingest raw data from CSV files. AWS Glue automatically infers a schema to allow the data to be queried. AWS Glue jobs are used to extract, clean, curate, and rewrite the data in an optimized format (Parquet)

before exposing visualizations to end users. This is all achieved using serverless technologies that reduce the operational burden to the analytics team.

We are going to explore each of these steps in more detail, in addition to the things you need to consider along the way. But, before we do, let's take a quick look at the other form of processing.

Real-time processing

Real-time processing is a way of processing an unbounded stream of data in order to generate real-time (or nearly real-time) alerts or business decisions. The response time for real-time processing can vary from milliseconds to minutes.

Real-time processing has its own ingestion components and has a streaming layer to stream data for further processing. Examples of real-time processing are:

- Processing IoT sensor data to generate alerts for predictive maintenance
- Trading data for financial analytics
- Identifying sentiment using a real-time Twitter feed

Before we get started, let's look at a common set of services that customers use to build data lakes for processing real-time data.

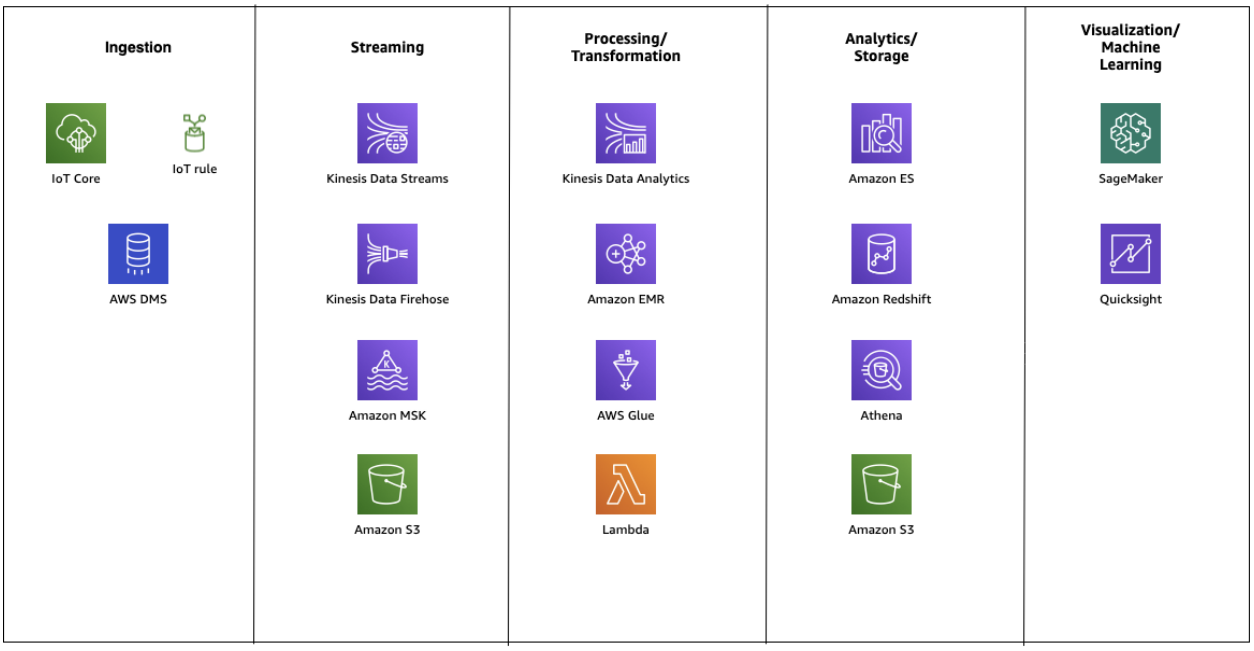


Figure 3 – Common services used to build data lakes for real-time data

Our example architecture is relatively simple and uses the following services: Amazon Kinesis, AWS Lambda, AWS Glue, Amazon Athena, Amazon S3 and Amazon QuickSight.

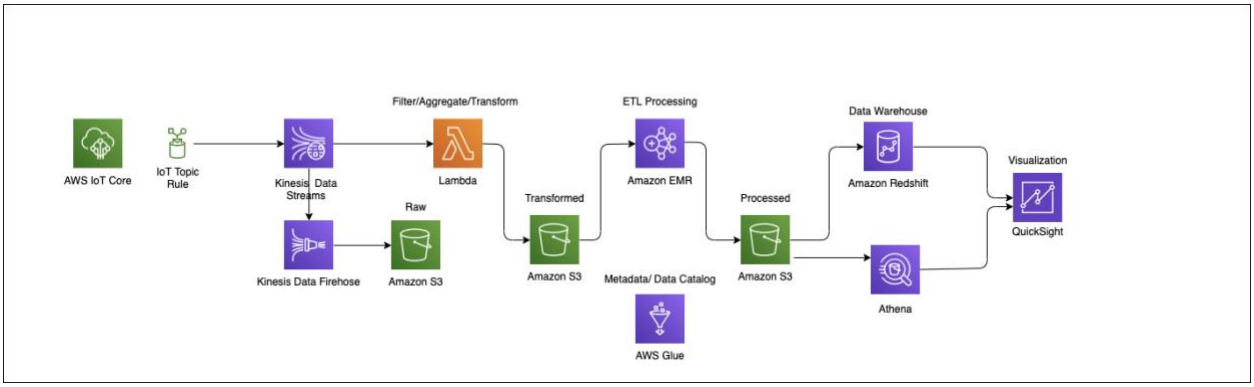


Figure 4 – Example architecture for real-time processing

Figure 4 shows that many IoT devices send their telemetry to AWS IoT Core. AWS IoT allows users to securely manage billions of connected devices and route those messages to other AWS endpoints. In this case, AWS IoT Core passes the messages Amazon Kinesis, which ingests streaming data at any scale. The raw data is split into two streams, one that writes the raw data to Amazon S3 and a second that uses AWS Lambda (a serverless compute service) to filter, aggregate, and transform the data before again storing it on Amazon S3. The manipulated data is then cataloged in AWS Glue and made available to end users to run ad hoc queries using Amazon Athena and create visualizations using Amazon QuickSight.

Understanding what influences data lakes costs

Across both real-time and batch processing, data flows through different stages. For each stage, there is an option to use managed services from AWS or to use compute, storage, or network services from AWS with third-party or open source software installed on top it.

In the case of managed services, AWS provides service features like high availability, backups, and management of underlying infrastructure at additional cost. In some cases, the managed services are on-demand serverless based solutions, where the customer is charged only when the service is used.

In case of third-party or open source software, the user pays AWS for infrastructure components being used and does the management and administration work themselves (in addition to license cost, if applicable).

Ingestion

Services required for ingestion depend on the source of data, the frequency at which the data should be consumed for processing to derive business value from it, and the data transfer rate.

Cost factors

Depending on the services you choose, the primary cost factors are:

- Storage – These are the costs you pay for storing your raw data as it arrives.
- Data transfer – These are the costs you pay for moving the data. Costs can be either bandwidth charges, leased line, or offline transfer. For example, Snowball. It is worth noting the analytics services should be deployed in the same Region to avoid unnecessary inter-Region data transfer. This improves performance and reduces costs.
- Managed service costs – These are the costs you pay (usually per second or per hour) for the service you are using, if you chose a managed service from AWS (for example, AWS IoT or AWS Transfer for SFTP).

Cost optimization factors

We recommend that you consider the following actions to reduce cost:

- Use data compression
- Reduce run frequency
- Choose a columnar format
- Create data lifecycle policies
- Choose serverless services

Streaming

This stage is only applicable for real-time processing. This stage is primarily responsible for ingesting the unbounded stream of data and providing guaranteed delivery for downstream processing.

Cost factors

The primary costs of this stage are

- Data transfer – These are the costs you pay for the rate at which data is consumed by the data streaming service.
- Streaming service costs – These are the costs you pay (usually per second or per hour) for AWS management of Amazon Kinesis or Amazon MSK service that is being used, including instance cost of Amazon MSK.
- Storage cost – This is the cost of storing data in streaming service until data is consumed by its consumers and processed.

Cost optimization factors

We recommend that you consider the following actions to reduce cost:

- Choose the right tool for the job
- Choose serverless services
- Use automatic scaling
- Use Reserved Instances

Processing and transformation

The cost of processing depends on number of factors, such as the size of the data being processed, the complexity of the transformations (which leads to the choice of tool), and how the data is stored (format and compression).

Cost factors

The primary cost at this stage includes:

- Processing unit cost – This is the cost of compute and memory required to process the data. If you use a service like Amazon EMR, the Amazon EC2 instance used to process the data will incur the cost.
- Managed service costs – These are the costs you pay (usually per second or per hour) for the management of the service. For a serverless service like AWS Glue, this will be billed based on the utilization of the service.
- Storage cost – This is the cost of storing the processed data.

Cost optimization factors

We recommend that you consider the following actions to reduce cost:

- Use data compression
- Partition data
- Right size your compute
- Choose serverless services
- Use Spot Instances
- Use Reserved Instances

Analytics/Storage

The business value of data lakes is derived using tools in analytics stage. Cost in this stage correlates directly to the amount of data collected and analytics done on the data to derive the required value.

Cost factors

The primary cost of this stage includes:

- Processing Unit cost – This is the cost of instances used by the analytics tool being used. Data Warehouse solution like Amazon Redshift or analytics solutions on Amazon EMR or operational analytics on Amazon Elasticsearch are all billed based on the instance type and number of nodes used.
- Storage cost – The data stored for analysis either on Amazon S3 or on the nodes of the analytics tool itself contribute to this cost.
- Scanned data cost – This is applicable only for serverless service like Athena and Amazon Redshift Spectrum, where the cost is based on the data scanned by the analytics queries.

Cost optimization factors

We recommend that you consider the following actions to reduce cost:

- Use data compression
- Reduce run frequency
- Partition data
- Choose a columnar format

- Choose serverless services
- Use Spot Instances
- Use Reserved Instances
- Right size your instances
- Use automatic scaling

Visualization/API access

Visualization is used to graphically present the enormous amount of data in data lake in a human consumable format. API access refers to allowing developers to integrate your data in their apps to make it meaningful for consumers. The rate at which data is refreshed or request affects the number of queries being fired (and cost) in the analytics stage.

Cost factors

The primary cost of this stage includes:

- Per user license cost – Most of SaaS visualization tools charge per user license cost.
- Processing unit cost – The cost required to host the visualization software is applicable only for hosted visualization tools.
- In-memory processing cost – This is the cost of the visualization need to load huge amount of data in memory for quick performance. The amount of data required to be loaded also has a direct impact on cost, for example, Super-fast, Parallel, In-memory Calculation Engine (SPICE) in Amazon QuickSight.
- Number of requests through an API – As developers integrate with their apps the number of requests made for data will increase.

Cost optimization factors

We recommend that you consider the following actions to reduce cost:

- Choose the right tool for the job
- Choose serverless services
- Use automatic scaling

- Use Reserved Instances
- Right size your instances

Metadata management, data catalog, and data governance

In addition to the previous data lake stages, you need the following components to make use of the data effectively and securely.

- Metadata management – Responsible for capturing technical metadata (data type, format, and schema), operational metadata (interrelationship, data origin, lineage) and business metadata (business objects and description)
- Data catalog – An extension of metadata that includes features of data management and search capabilities
- Data governance – Tools that manage the ownership and stewardship of data along with access control and management of data usage

Cost factors

The primary cost of this stage includes:

- Processing cost – This is the cost associated with processing required to generate the catalog or metadata of data stored. In governance tools, this cost depends on number of governance rules that is being processed.
- Storage cost – This is the cost associated with amount of metadata and catalog stored.
- License cost – If there is any third party involved in providing these services, it will include the cost of that license.

Cost optimization practices

We recommend that you consider the following actions to reduce cost:

- Choose the right tool for the job
- Choose serverless services
- Reduce run frequency
- Partition data

- Choose a columnar format

Cost optimization

Across stages, the following are general practices to optimize cost of the services used.

Use data compression

There are many different formats for compression. You should be selective about which one you choose to make sure it is supported by the services you are using.

The formats provide different features for example some formats support splitting meaning they support parallelization better, whilst others have better compression but at the cost of performance. Compressing the data, reduces the storage cost and also reduces the cost for scanning data.

To give an example of how compression can help reduce costs we took a CSV dataset and compressed it using bzip2. This reduced the data from 8 GB to 2 GB a 75% reduction. If we extrapolate this compression ratio to a larger dataset, 100 TB the savings are significant:

- Without compression: $2,406.40 per month
- With compression: $614.40 per month

This represents a saving of 75%.

> **Note**: Compression also reduces the amount of data scanned by compute services such as Amazon Athena. An example of this covered in Columnar format section.

Reduce run frequency

In this scenario, let's assume we generate 1 TB of data daily. We could potentially stream this data, because it's available, or choose to batch it and transfer it in one go. Let's assume we have 1 Gbps bandwidth available for the transfer from our source system, for example on-premises systems. A single push of 1 TB daily would take around two hours. In this case we'll allow three hours for some variance in speed.

Also, in this scenario, we will use AWS SFTP to transfer the data. In the first scenario, we need the AWS SFTP service constantly running. In the second scenario, we only need it running for three hours a day to complete our transfer.

- Running constant transfers as the data is available: $259.96 per month
- Running a single batch transfer daily: $68.34 per month

This represents a saving of just over 73%.

Partition data

Partitioning data allows you to reduce the amount of data that is scanned within queries. This helps both reduce cost and improve performance. To demonstrate this, we used the GDELT dataset. It contains around 300 million rows stored in many CSV files. We stored one copy un-partitioned and a second copy partitioned by year, month, day.

```
SELECT count(*) FROM "gdeltv2"."non-partioned|partitioned" WHERE
year = 2019;
```

Unpartitioned:

- Cost of query (102.9 GB scanned): $0.10 per query
- Speed of query: 4 minutes 20 seconds

Partitioned:

- Cost of query (6.49 GB scanned): $0.006 per query
- Speed of query: 11 seconds

This represents a saving of 94% and improved performance by 95%.

Choose a columnar format

Choosing a columnar format such as parquet, ORC or Avro helps reduce disk I/O requirements which can reduce costs and drastically improve performance in data lakes. To demonstrate this, we will again use GDELT. In both examples there is no partitioning but one dataset is stored as CSV and one as parquet.

```
SELECT * FROM "gdeltv2"."csv|parquet" WHERE globaleventid =
778617562
```

CSV:

- Cost of query (102.9 GB scanned): $0.10 per query
- Speed of query: 4 minutes 20 seconds

Partitioned:

- Cost of query (1.04 GB scanned): $0.001 per query
- Speed of query: 12.65 seconds

This represents a saving of 99% and improved performance by 95%.

Create data lifecycle policies

With Amazon S3, you can save significant costs by moving your data between storage tiers. In data lakes we generally keep a copy of the raw in-case we need to reprocess data as we find new questions to ask of our data. However, often this data isn't required during general operations. Once we have processed our raw data for the organization, we can choose to move this to a cheaper tier of storage. In this example we are going to move data from standard Amazon S3 to Amazon S3 Glacier.

For our example, we will model these costs with 100TB of data.

- Standard Amazon S3: **$2,406.40** per month
- Amazon S3 Glacier: **$460.80** per month

This represents a saving of just over 80%.

Right size your instances

AWS services are consumed on demand. This means that you pay only for what you consume. Like many AWS services, when you run the job you define how many resources you want and you also choose the instance size you want. You also have ability to stop the instance when it's not being used and you can spin it up based on specific schedule. For example, transient EMR clusters can be used for scenarios where data has to be processed once a day or once a month.

Monitor instance metrics with analytic workloads and downsize the instances if they are over provisioned.

Use Spot Instances

Amazon EC2 Spot Instances let you take advantage of unused Amazon EC2 capacity in the AWS Cloud. Spot Instances are available at up to a 90% discount compared to on-demand prices. For example, using spot helps the Salesforce save up to 40 percent over Amazon EC2 Reserved Instance pricing and up to 80 percent over on-demand pricing. Spot is a great use case for batch when time to insight is less critical.

In the following example, we have modeled a transient Amazon EMR cluster that takes six hours to complete its job.

- On-Demand Instance: $106.56 per month
- Spot Instance (estimated at a conservative 70%): $31.96 per month

This represents a saving of just over 70%.

Use Reserved Instances

Amazon EC2 Reserved Instances (RI) provide a significant discount (up to 72%) compared to On-Demand pricing and provide a capacity reservation when used in a specific Availability Zone. This can be useful if some services used in the data lake will be constantly utilized.

A good example of this can be Amazon Elasticsearch. In this scenario we'll model a 10 node Amazon Elasticsearch cluster (3 master + 7 data) with both on-demand and reserved pricing. In this model, we'll use c5.4xlarge instances with a total storage capacity of 1 TB.

- On-demand: $ 7,420.40 per month
- RI (three years, all up front): $ 3,649.44 per month

This represents a saving of just over 50%.

Choose the right tool for the job

There are many different ways to interact with AWS services. Picking the right tool for the job, helps reduce cost.

If you are using Kinesis Data Streams and pushing data into it, you can choose between the AWS SDK, the Kinesis Producer Library (KPL), or the Kinesis Agent. Sometimes the option is based on the source of the data and sometimes the developer can choose.

Using the latest KPL enables you to use a native capability to aggregate multiple messages/events into a single PUT unit (Kinesis record aggregation). Kinesis data streams shards support up to 1000 records per second or 1-MB throughput. Record aggregation by KPL enables customers to combine multiple records into a single Kinesis Data Streams record. This allows customers to improve their per shard throughput.

For example, if you have a scenario of pushing 100,000 messages/sec with 150 byes in each message into a Kinesis data stream, you can use KPL to aggregate them into 15 Kinesis data stream records. The following is the difference in cost between using (latest version) KPL and not.

- Without KPL (latest version): $4,787.28 per month
- With KPL (latest version): $201.60 per month

This represents a saving of 95%.

Use automatic scaling

Automatic scaling is the ability to spin resources up and down based on the need. Building application elasticity enables you to incur cost only for your fully utilized resources.

Building EC2 instances using an Auto Scaling group can provide elasticity for Amazon EC2 based services where applicable. Even for serverless services like Kinesis where shards are defined per stream during provisioning, AWS Application Auto Scaling provides ability to automatically add or remove shards based on utilization.

For example, if you are using Kinesis streams to capture user activity for an application hosted in specific Region the streaming information might vary between day and night. During day times, when the user activity is higher you might need more shards compared to night times when the user activity would be very low. Being able to configure AWS Application Auto Scaling based on your utilization enables you to optimize your cost for Kinesis.

Data flowing at rate of 50,000 records/sec with each record having 2 K bytes, will require 96 shards. If the data flow reduces to 1000 records/sec for eight hours during night, it would only need two shards.

- Without AWS Application Auto Scaling: $1068.72 per month
- With AWS Application Auto Scaling (downsizing): $725.26 per month

This represents a saving of 32%.

Choose serverless services

Serverless services are fully managed services that incur costs only when you use them. By choosing a serverless service, you are eliminating the operational cost of managing the service.

For example, in scenarios where you want to run a job to process your data once a day, using serverless service incurs a cost only when the job is run. In comparison, using a self-managed service incurs a cost for the provisioned instance that is hosting the service.

Running a Spark job in AWS Glue to process a CSV file stored in Amazon S3 requires 10 minutes of 6 DPU's and cost $0.44. To execute the same job on Amazon EMR, you need at least three m5.xlarge instances (for high availability) running at a rate of $0.240 per hour.

- Using a serverless service: $13.42 price per month
- Not using a serverless service: $527.04 price per month

This represents a saving of 97%.

Similar savings are applicable between running Amazon Kinesis vs Amazon MSK and between Amazon Athena vs Amazon EMR.

While we are not covering cost optimization in any detail, we have indicated the macro factors that you should consider during each stage to keep costs down. To understand cost optimization in detail, see the Analytics Lens for the AWS Well-Architected Framework.

Monitoring data lakes costs

Once the data lake is built to provide its intended features, we recommend that you measure the cost and tie it back to business value it provides. This enables you to perform a return-on-investment analysis on your analytics portfolio. It also enables you to iterate and tune the resources for optimal cost based on cost optimization techniques discussed earlier in this whitepaper.

To track the cost utilization for your analytic workloads, you need to define your cost allocation strategy. Cost-allocation tagging ensures that you tag your AWS resources

with metadata key-value pairs that reflect the business unit the data lake pipeline was built for.

Tags enable you to generate billing reports for the resources associated with a particular tag. This lets you to either do charge-back or return-on-investment analysis.

Another strategy to track your costs is to use multiple AWS accounts and manage them using AWS Organizations. In this approach, every business unit owns their AWS account and provisions and manages their own resources. This lets them track all cost associated with that account for their data lake needs.

By tracking costs and tying it back to your business value, you can complete your cost modeling for your first analytics workload. This process also lets you iterate and repeat the process again of deciding business use cases, defining KPI and building data lake features on top of your already built data lake foundation while monitoring the cost associated with it.

Cost modeling your first experiments

To help give confidence to cost model your first analytics project, we are going to do the following:

- Generate a couple of fictitious customer scenarios for analytics experiments
- Walk through our very lightweight cost model
- Build the actual solution and demonstrate how close we got to the actual costs

AWS offers a free tier for most of its services. This lets you experiment with no cost. For the cost modeling that follows, we discuss both free tier and the actual costs.

It is also important to know that AWS pricing for every product is transparent and the details on how the costs can be calculated are available on every product page. In this scenario we will use the following pricing pages.

- AWS Glue pricing
- Amazon SageMaker Pricing
- Amazon VPC pricing
- Amazon S3 pricing

- Amazon Athena pricing
- Amazon QuickSight Pricing

Read through these pages before you start your experiments. They can help you think about the variables you need to consider when you are gathering details about the data you will be using.

Scenario 1: Improving healthcare KPIs

In this scenario, we are a health trust company with a goal to improve the health of the community we are serving. The following are a few key performance indicators (KPIs) that we want to achieve in near future.

- Reduce health-related crime rate by 5% over 12 months
- Reduce health-related crime rate by 15% over 36 months

To achieve these KPIs, we decide to analyze:

1. Drugs that have a strong correlation to criminal activity
2. Health conditions that have a strong correlation to criminal activity
3. Unique citizens who are using the drugs that have top five correlations to criminal activity
4. Unique citizens who have health conditions that have top five correlations to criminal activity

Our plan is to use these and work with identified citizens and provide them alternate drugs or provide them counseling or any other treatments as applicable, to prevent them from committing any crime. We believe doing this will improve the overall mental health of our community, in addition to other activities we have been working on.

Build a data lake

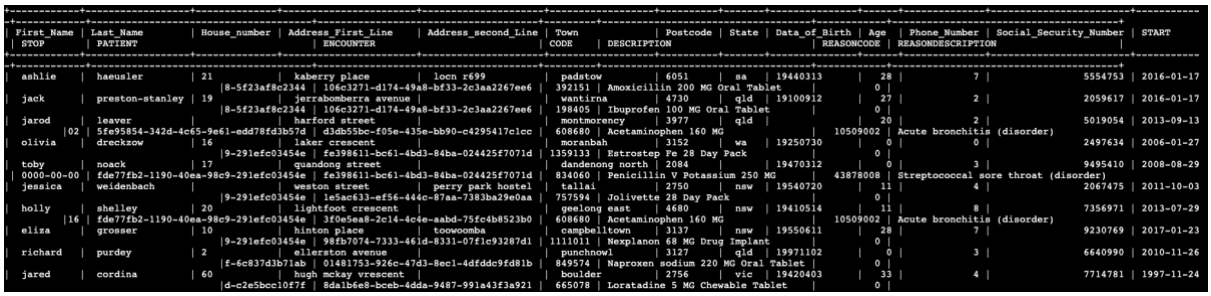
Our health trust company has come to a decision that to conduct the required analysis, they must build a data lake.

We will need the healthcare data which we already own and we would have to get data from police department to identify crime trends. We would like to automate the setup so we can ingest and process new data and make it available to teams around our organizations.

The following data has been generated or obtained from the following sources:

- Fake citizen data with duplicates
- Fake healthcare data
- Open police data

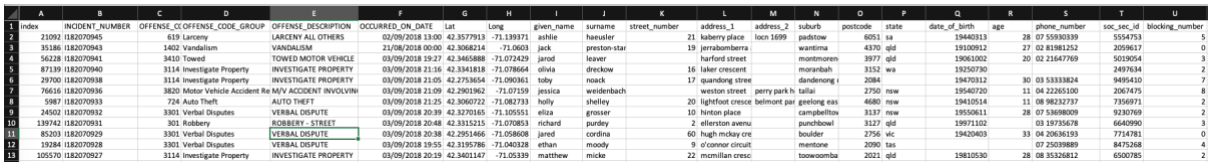
The following is an example of the synthetic data created for each provider.



First_Name	Last_Name	House_number	Address_First_Line	Address_second_Line	Town	Postcode	State	Data_of_Birth	Age	Phone_Number	Social_Security_Number	START	STOP	PATIENT	ENCOUNTER	CODE	DESCRIPTION	REASONCODE	REASONDESCRIPTION
ashlie	haeusler	21	kaberry place	locn r699	padstow	6051	sa	19440313	28	7	5554753	2016-01-17		8-5f23af8c2344	106c3271-d174-49a8-bf33-2c3aa2267ee6	392151	Amoxicillin 200 MG Oral Tablet		
jack	preston-stanley	19	jerrabomberra avenue		wantirna	4730	qld	19100912	27	2	2059617	2016-01-17		8-5f23af8c2344	106c3271-d174-49a8-bf33-2c3aa2267ee6	198405	Ibuprofen 100 MG Oral Tablet		
jarod	leaver		harford street		montmorency	3977	qld		20	2	5019054	2013-09-13	02	5fe95854-342d-4c65-9e61-edd78fd3b57d	d3db55bc-f05e-435e-bb90-c4295417c1cc	608680	Acetaminophen 160 MG	10509002	Acute bronchitis (disorder)
olivia	dreckzow	16	laker crescent		moranbah	3152	wa	19250730	0	0	2497634	2006-01-27		9-291efc03454e	fe398611-bc61-4bd3-84ba-024425f7071d	1359133	Bstrostep Fe 28 Day Pack		
toby	noack	17	quandong street		dandenong north	2084		19470312	0	3	9495410	2008-08-29	0000-00-00	fde77fb2-1190-40ea-98c9-291efc03454e	fe398611-bc61-4bd3-84ba-024425f7071d	834060	Penicillin V Potassium 250 MG	43878008	Streptococcal sore throat (disorder)
jessica	weidenbach		weston street	perry park hostel	tallai	2750	nsw	19540720	11	4	2067475	2011-10-03		9-291efc03454e	1e5ac633-ef56-444c-87aa-7383ba29e0aa	757594	Jolivette 28 Day Pack		
holly	shelley	20	lightfoot crescent		geelong east	4680	nsw	19410514	11	8	7356971	2013-07-29	16	fde77fb2-1190-40ea-98c9-291efc03454e	3f0e5ea8-2c14-4c4e-aabd-75fc4b8523b0	608680	Acetaminophen 160 MG	10509002	Acute bronchitis (disorder)
eliza	grosser	10	hinton place	toowoomba	campbelltown	3137	nsw	19550611	28	7	9230769	2017-01-23		9-291efc03454e	98fb7074-7333-461d-8331-07f1c93287d1	1111011	Nexplanon 68 MG Drug Implant		
richard	purdey	2	ellerston avenue		punchbowl	3127	qld	19971102	0	3	6640990	2010-11-26		f-6c837d3b71ab	01481753-926c-47d3-8ec1-4dfddc9fd81b	849574	Naproxen sodium 220 MG Oral Tablet		
jared	cordina	60	hugh mckay vrescent		boulder	2756	vic	19420403	33	4	7714781	1997-11-24		d-c2e5bcc10f7f	8da1b6e8-bceb-4dda-9487-991a43f3a921	665078	Loratadine 5 MG Chewable Tablet		

Figure 5 – Healthcare data

The following is an example of the open police data we used.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	index	INCIDENT_NUMBER	OFFENSE_CC	OFFENSE_CODE_GROUP	OFFENSE_DESCRIPTION	OCCURRED_ON_DATE	Lat	Long	given_name	surname	street_number	address_1	address_2	suburb	postcode	state	date_of_birth	age	phone_number	soc_sec_id	blocking_number
2	21092	I182070945	619	Larceny	LARCENY ALL OTHERS	02/09/2018 13:00	42.3577913	-71.139371	ashlie	haeusler	21	kaberry place	locn 1699	padstow	6051	sa	19440313	28	07 55930339	5554753	5
3	35186	I182070943	1402	Vandalism	VANDALISM	21/08/2018 00:00	42.3068214	-71.0603	jack	preston-star	19	jerrabomberra		wantirna	4370	qld	19100912	27	02 81981252	2059617	0
4	56228	I182070941	3410	Towed	TOWED MOTOR VEHICLE	03/09/2018 19:27	42.3465888	-71.072429	jarod	leaver		harford street		montmoren	3977	qld	19061002	20	02 21647769	5019054	3
5	87139	I182070940	3114	Investigate Property	INVESTIGATE PROPERTY	03/09/2018 21:16	42.3341818	-71.078664	olivia	dreckow	16	laker crescent		moranbah	3152	wa	19250730		19250730	2497634	2
6	29700	I182070938	3114	Investigate Property	INVESTIGATE PROPERTY	03/09/2018 21:05	42.2753654	-71.090361	toby	noack	17	quandong stree		dandenong	2084		19470312	30	03 53333824	9495410	7
7	76616	I182070936	3820	Motor Vehicle Accident Re	M/V ACCIDENT INVOLVIN	03/09/2018 21:09	42.2901962	-71.07159	jessica	weidenbach		weston street	perry park h	tallai	2750	nsw	19540720	11	04 22265100	2067475	8
8	5987	I182070933	724	Auto Theft	AUTO THEFT	03/09/2018 21:25	42.3060722	-71.082733	holly	shelley	20	lightfoot cresce	belmont par	geelong eas	4680	nsw	19410514	11	08 98232737	7356971	2
9	24502	I182070932	3301	Verbal Disputes	VERBAL DISPUTE	03/09/2018 20:39	42.3270165	-71.105551	eliza	grosser	10	hinton place		campbelltos	3137	nsw	19550611	28	07 53698009	9230769	2
10	139742	I182070931	301	Robbery	ROBBERY - STREET	03/09/2018 20:48	42.3315215	-71.070853	richard	purdey	2	ellerston avenu		punchbowl	3127	qld	19971102		03 19735678	6640990	3
11	85203	I182070929	3301	Verbal Disputes	VERBAL DISPUTE	03/09/2018 20:38	42.2951466	-71.058608	jared	cordina	60	hugh mckay cre		boulder	2756	vic	19420403	33	04 20636193	7714781	0
12	19284	I182070928	3301	Verbal Disputes	VERBAL DISPUTE	03/09/2018 19:55	42.3195786	-71.040328	ethan	moody	9	o'connor circuit		mentone	2090	tas			07 25039889	8475268	4
13	105570	I182070927	3114	Investigate Property	INVESTIGATE PROPERTY	03/09/2018 20:19	42.3401147	-71.05339	matthew	micke	22	mcmillan cresc		toowoomba	2021	qld	19810530	28	08 35326812	6500785	2

Figure 6 – Police data

The files contain separate data but there is some commonality in the details being recorded. For example they each use first name, last name, and address. We also have a unique key present in both systems that will allow us to correlate the data and draw our conclusions. This key is the Social Security number.

> Note: This is fictitious data we have created to help demonstrate cost modeling. No actual conclusion can be drawn from the data. It is important that organizations use data responsibly. There is a good series of blog posts that discuss good practices for ethical sharing of data to help organizations adopt good practices.

Process

To start the project, let's consider the phases we highlighted earlier:

- For **data ingestion**, we need to identify our source data, understand some high-level characteristics (for example, what protocols can be used to transfer the data), what format the data is in, and the size of the data.
- For **processing and transformation**, we need to think about how we need to manipulate the data to make it ready for our analytics. We need to think how we catalog it and make it available for our next phase.
- For **analytics**, we need to think about how we might explore our data and what analytics we may wish to perform to get the data ready to be queried by our business users.
- For **visualization, data access and machine learning**, we need to understand how we will expose our data so we can ask questions of it and drive value.

Based on our requirements, we will be using the following architecture:

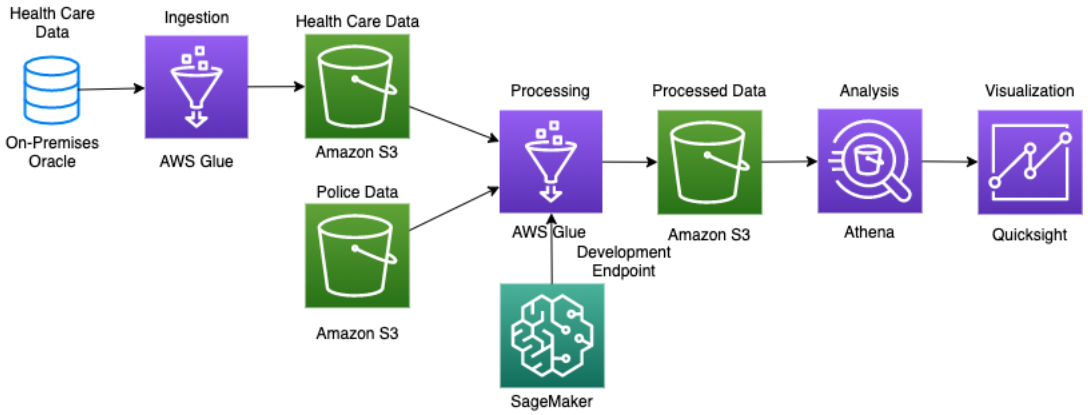


Figure 7 – Example architecture for Scenario 1

Data ingestion

After an initial assessment, we identify a health dataset in our internal system in a relational database that will help us. After performing a count, we discover there are 70,000 citizens tracked in the system. Because we are collaborating with the police, we cannot have direct access to the system and agree that the police department will provide a CSV file with 25,000 citizens in it.

We assess our options and decide that we will use AWS Glue to extract the data from our relational database and use Amazon S3 to allow the police department to upload

the data directly. Both of these methods will allow us to automate the ongoing capture and processing of the data.

Amazon S3 cost modeling

To estimate the cost, we obtain a copy of the CSV file from the police department. The file is only 5 MB. We use the AWS Pricing Calculator to cost our solution.

Amazon S3 offers 5 GB of free storage per month, 20,000 GET requests, and 2,000 PUT requests. This comfortably allows us to experiment with uploading and storing the police department data. Had that not been the case, the cost would be under one cent anyway.

AWS Glue cost modeling

To get the data from the relational database we decide to use three features in AWS Glue:

- Job – to read the data from the database and convert it to parquet stored on Amazon S3
- Crawler – to return a schema from the relational database and store it in our central catalog
- Connection – to connect to our database

To understand how the service is charged, we refer to the AWS Glue pricing page.

In order for the AWS Glue job to extract and transform the data to Amazon S3, we estimate that we'll use two DPUs. We also estimate that we can extract and convert the records in 10 minutes. An AWS Glue job costs $0.44 per hour. So, our calculation is:

(($0.44 * 2)/60 minutes) *10 minutes = $0.15

Crawlers are also charged on the same basis as Glue jobs. We estimate that we will crawl our data in 10 minutes, but this time with a single DPU.

$0.44 /60 minutes) *10 minutes = $0.07

To connect to our database, we also need to provision some network components. We will use a NAT gateway to allow outbound connectivity through to our environment. We estimate we will complete our experiment in two days.

A NAT gateway costs $36.55 per month with a 1 GB data transfer.

($36.55/730 hours in a month) * 48 hours = $2.40

Finally, for the storage we will again be able to use the AWS Free Tier.

Service	Cost
AWS Glue:	~$0.22
Amazon VPC:	~$2.4
Storage:	$0
Total	~$2.62

Processing and transformation

For our processing we need to do the following tasks.

- Explore our data, running ad hoc queries to understand in more detail what we are working with
- Develop code to merge the data so we can interrogate it
- Run queries against our data to prove our hypothesis

To achieve this, we will use AWS Glue developer endpoints and SageMaker. SageMaker will provide us a notebook development environment to explore our data and AWS Glue will generate our labeled, train and run our model in addition to catalog and store the data so we can make the output available to our users around the organization.

Because this is data exploration, we don't have an exact figure for how long we will require the SageMaker development environment. However, we will initially estimate needing the notebook for two days. Because we can stop the notebook overnight while we are not working, we will estimate 16 hours use. At this point, we know that our datasets are relatively small. So we will assume a small notebook instance, ml.t2.medium.

$0.0520 * 16 hours = $0.83

Because we don't know how long it actually takes, we must estimate how long we will need an AWS Glue development endpoint available. Because this is the first time we

are using AWS Glue, we will estimate requiring a developer endpoint for eight hours. This should give us plenty of time to develop our AWS Glue job to transform the data.

The AWS Glue pricing page shows that a development endpoint is provisioned with five DPU's. A DPU is billed at $0.44 per DPU hour. So eight hours our costs would be $17.60.

Service	Dimensions	Cost
AWS Glue	($0.44 * 5 DPUs) * 8 hours	~$17.60
SageMaker		~$0.83
Total		~$18.43

Analytics

We are getting close to be able to prove whether the experiment is driving business value. In this phase, we need to start asking questions of our data. For this we will assume a number of queries being run over our dataset. B we know the dataset is approximately 70,000 records and likely to be less than 1 GB. To explore and generate the required views for our business users, we will estimate needing to run 500 queries over the data.

Athena is charged based on the amount of data scanned.

Service	Dimensions	Cost
Athena	1 GB * 500 queries = 1TB	~$2.50
Total		~$2.50

Visualization, data access and machine learning

Our final stage is to create a dashboard to present the data back to the business units so they can make decisions based on the data. Because we are only experimenting to see if the data is valuable for users, we're going to assume we only have two people creating dashboards and demonstrating the capability to the various business units.

Service	Dimensions	Cost
Amazon QuickSight	$24 * 2	~$48/month
Athena	(50 queries per day for demos * 1 GB) * 25 days of demos over the month	~$1.22/month
Total		~$49.22

Total cost

We estimate the setup of the solution to take around two days. This accounts for preparing the data and running queries across our processed data. We are using serverless services so we can just start consuming them with no setup.

Once we have completed our data preparation in the first three stages, we have scheduled a month of demos where we interface with the business units to demonstrate the value of the project. To help make this easier to understand, we have split the project into two parts. The first are the cost to ingest, prepare, process, and run analytics across the data. The second are costs to demonstrate the outcomes back to the rest of the business.

Part	Cost
Preparation, processing, and analytics to prove outcome	$23.55
One month of demos to the rest of the business	$49.22

We made this split because we won't actually fake taking demos to business units over a month. Rather, we will focus on validating the first part which is proving the outcome.

While this is based on a number of assumptions, this should give us confidence the costs are minuscule in terms of the value we can generate for the organization. This should put us in a position to start the project.

Once we have completed this project, we will also have accurate figures from each stage that will help us make more informed decisions for cost modeling our next round of experiments, such as automating the ingesting and processing of data for our users to keep it fresh, or ingesting more sources of data to correlate across.

The actual cost

In this final section, we went through the process of setting up a new AWS account and running the experiments in each phase as described earlier. We then used Cost Explorer to get a detailed breakdown of the charges involved to see how accurate our modeling was.

The initial three phases to ingest, process, and analyze the data to prove a valuable outcome was $23.55. We estimated the initial piece of work to take around two days, which we managed to stick to.

The following is a breakdown of the costs by service.

Service	Jun 2020	Service Total
AWS Glue	$20.61	$20.61
EC2 – Other	$0.77	$0.77
Amazon S3	$0.08	$0.08
SageMaker	$0.01	$0.01
Athena	$0.00	$0.00
CloudTrail	$0.00	$0.00
Total		$21.47

The actual costs to make the data available were $21.47. This is approximately 9% less than we estimated.

As discussed, we have not gone through a fake month of demos with the dashboards.

Scenario 2: Improving manufacture turnaround time

In this second scenario, we are a manufacturing company with a goal to improve the timing of our delivery. The purchasing team of this manufacturing company is looking to improve the turnaround time of the parts required for their orders by 20%

To achieve this KPI, we decided to analyze:

- Customer orders and line items, to identify parts that are frequently ordered
- Suppliers data to identify suppliers that kept orders waiting
- Customer data to identify large volume customers
- Supplier shipping modes and order priority
- Correlation of any returned items with supplier

Our plan is to identify the suppliers with good track record and use them for parts requested by large volume customers. This should bring down the turnaround time for majority of the parts requested overall.

Build a data lake

Our manufacturing company have come to a decision that to conduct the required analysis, we will have to build a data lake. The following is a schema of dataset available for processing. The dataset used is from TPC-H benchmark data.

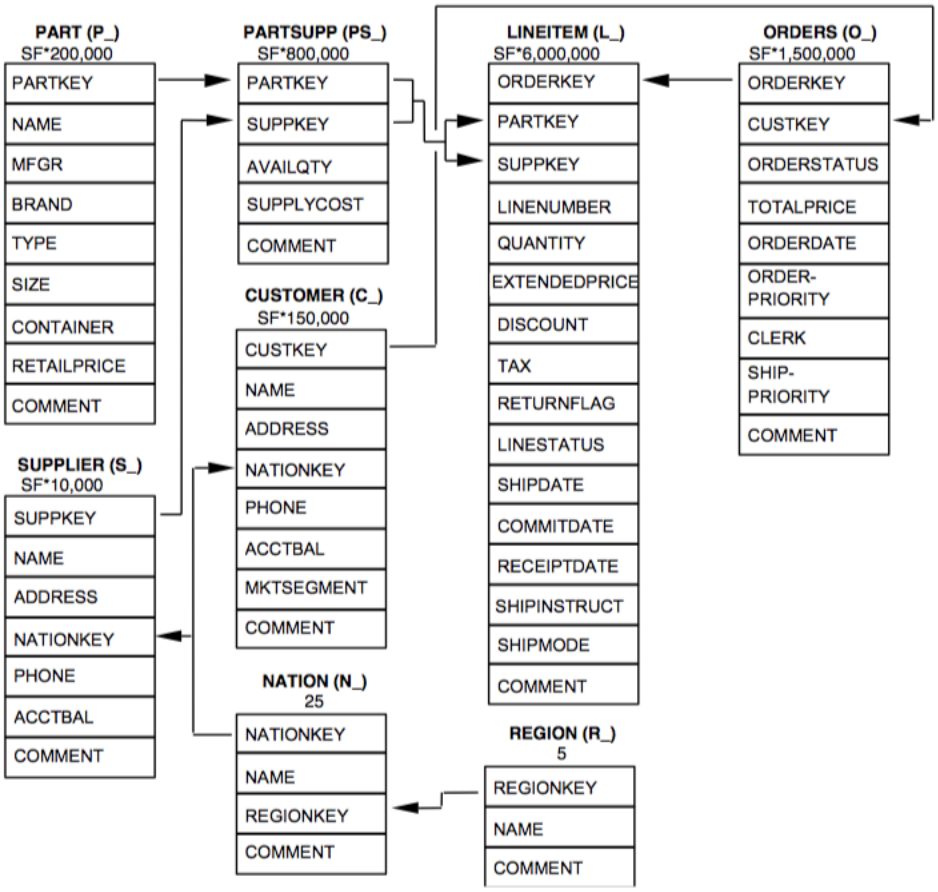


Figure 8 – Schema for a dataset from TPC-H benchmark data

> **Note**: This is benchmark data. Though it can help demonstrate how cost modeling works, no actual conclusion can be drawn from the data. It is important that organizations use data responsibly. There is a good series of blog posts that discuss good practices for ethical sharing of data to help organizations adopt good practices.

Process

To start our project let's consider the phases we highlighted earlier.

- Data ingestion
- Processing and transformation
- Analytics
- Visualization, data access, and machine learning

For **data ingestion**, we are going to have these datasets delivered once a day into AWS SFTP in a specific folder.

For **processing and transformation,** we need to process the data and load into a data-warehouse solution

For **analytics** we need a data-warehouse solution that will be available to fire different analytical queries and generate reports

For **visualization, data access and machine learning**, we need to understand how we will expose our data so we can ask questions of it and drive value.

Based on our requirements, we will be using the following architecture:

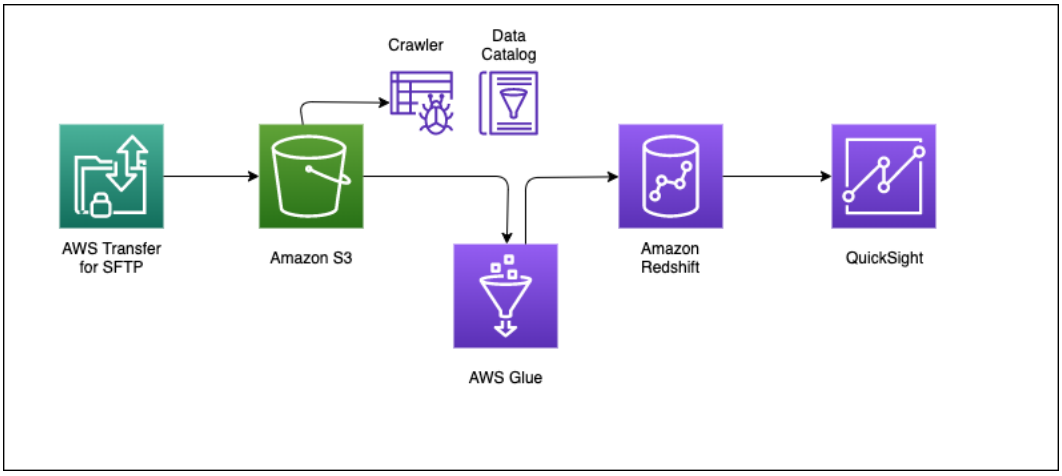


Figure 9 – Example architecture for Scenario 2

Data ingestion

The order and supplier systems in the manufacturing company agreed to push all the data on a daily basis to and SFTP location.

AWS Transfer for SFTP cost modeling

To estimate the cost, we obtain the dataset from TPC-H data. The total dataset is 3 TB. We use the AWS Pricing Calculator to cost our solution.

AWS Transfer for SFTP is priced based on data upload and download cost, along with endpoint cost charged at a per hour basis. To collect all these data files from different systems, the endpoint will be up for four hours per day.

Dimensions	Cost
SFTP Endpoint Cost	$1.2
Upload per day (1 TB new data)	$40
Download per day (1 TB new data)	$40
Total	~$81.2
Monthly total (21 business days)	~$1705.2

Amazon S3 cost modeling

Raw data received from source systems will be stored in Amazon S3, which will eventually then be processed by AWS Glue and loaded into Amazon Redshift. Amazon S3 will incur storage cost and costs associated with PUT and GET requests. The total dataset storage size is 3.84 TB

Dimensions	Cost
Amazon S3 Storage (3 TB)	$69
PUT requests (300000)	$1.5
GET requests(3000000)	$1.2
Total	~$71.7
Monthly Total	~$2126.7

Processing and transformation

In this use case we are going to use Glue for processing, transforming the data and loading into Amazon Redshift. In addition, Glue is also used to crawl and create a catalog of the data. The data files in this use case are CSV and JSON.

Glue cost modeling

- Glue job – to read the data from the database and convert it to parquet stored on Amazon S3
- Glue crawler – to return a schema from the relational database and store it in our central catalog
- Glue connection – to connect to our database

We view the AWS Glue pricing page to understand how the service is charged.

We estimate the AWS Glue job to extract and transform the data from Amazon S3 and load to Amazon Redshift. We estimate following time for each of the dataset:

Dataset	Dimensions	Cost
Customers	30 min	
Orders	20 min	
Lineitem	300 min	
Part	15 min	
PartSupp	60 min	
Supplier	20 min	
Total	(0.44/60 minutes) *445 minutes	$3.26

Crawlers are also charged on the same basis as AWS Glue jobs. Again, we estimate that we will crawl our data in two minutes, but this time with a single DPU.

$0.44 /60 minutes) *2 minutes = $0.014

Finally, in order to connect to our database, we also need to provision some network components. Again, we use the AWS Pricing Calculator to help us consider the costs for our networking.

We will use a NAT gateway to allow outbound connectivity through to our environment. We estimate we will complete our experiment in two days.

A NAT gateway costs $36.55 per month with a 1 GB data transfer.

($36.55/730 hours in a month) * 48 hours = $2.40

Finally, for the storage we will again be able to use the AWS Free Tier.

Service	Cost
AWS Glue	~$3.26
Amazon VPC	~$2.40
Storage	$0
Total	~$5.66
Monthly Total	~118.86

Analytics

This use case needs a data warehouse to be able to run multiple analytic queries. For storing 3 TB data storage per day and for a need to process month data of size 30TB, we need at least two nodes of ra3.4xlarge which give 12 virtual CPUs and 96 TB memory. Cost per day would be as follows:

Resource	Cost
Node	~$156.48
Total	~$156.48
Monthly Total	~$4694.4

Visualization, data access, and machine learning

Our final stage is to create a dashboard to present the data back to the business units so they can make decisions based on the data. Because we are only experimenting to see if the data is valuable for users, we're going to assume we only have two people creating dashboards and demonstrating the capability to the various business units.

Service	Dimensions	Cost
Amazon QuickSight	$24 * 2 = $48 for the month	~$48
Athena	(50 queries per day for demos * 1GB) * 25 days of demos over the month	~1.22
Total		~$49.22

Total cost

We estimate the setup of the solution to take around two days to prepare the data and run queries across our processed data. This requires setup for AWS Transfer for SFTP and Amazon Redshift.

Once the initial setup is complete, the other services are serverless services that do not require any additional setup. Once we complete our data preparation in the first three stages, we have scheduled a month of demos where we interface with the business units to demonstrate the value of the project. To help make this easier to understand we have split the project into two parts. The first part includes the cost to ingest, prepare,

process, and run analytics across the data. The second part includes costs to demonstrate the outcomes back to the rest of the business.

Phases	Cost
Data Ingestion, processing and transformation, and analytics to prove outcome (per month)	$8645.14
One month of demos to the rest of the business (per month)	$49.22

Processing cost per day will end up to be around $412.32. While this is based on a number of assumptions, this should give us confidence the costs are minuscule in terms of the value we may generate for the organization. This should put us in a position to start the project.

Once we have completed this project, we will also have accurate figures from each stage which will help us make more informed decisions for cost model our next round of experiments. For example, automating the ingesting and processing of data for our users to keep it fresh or ingesting more sources of data to correlate across.

The actual cost

In this final section we went through the process of setting up a new AWS account and running the experiments in each phase as described previously. We then used Cost Explorer to get a detailed breakdown of the charges involved to see how accurate our modeling was.

The initial three phases to ingest, process, and analyze the data to prove a valuable outcome was $631.94 for two days ($315.97), not including SFTP. Including SFTP adds $81.20 per day.

The following is a breakdown of the costs by service.

Service	Aug 2020	Service Total
Amazon Redshift	$346.27	$346.27
AWS Glue	$154.57	$154.57

Service	Aug 2020	Service Total
EC2 – Other	$114.79	$114.79
Amazon S3	$13.68	$13.68
SageMaker	$2.45	$2.45
GuardDuty	$0.17	$0.17
Total	$631.94	$631.94

The actual costs for data processing per day is $397.17. This is approximately 3% less than we estimated.

Conclusion

Customers struggle with starting their analytics projects because it is difficult to estimate costs when you have no knowledge or foresight of their unique requirements as an organization. Without a cost estimate, projects fail to get funding and organizations miss the enormous value making data driven decisions can offer.

This whitepaper offers a way forward. We have shown how you can tackle this challenge. This involves breaking down the problem into smaller components that can more easily be sized, costed, and implemented, while delivering measurable business value early on in your project.

You can use the scenarios as templates to build out a picture of what your analytics experiment will look like. The scenarios can help your organization start a journey towards making data-based decisions and drive business value, offering benefits for your organization and its customers.

Contributors

Contributors to this document include:

- Charlie Llewellyn, Solutions Architecture (Public Sector), Amazon Web Services
- Anusha Dharmalingam, Enterprise Solutions Architecture, Amazon Web Services

Further Reading

The following resources can help you get started in running big data analytics on AWS:

- Well Architected for Analytics – This document helps you understand AWS best practices and strategies to use when designing architectures for analytics applications.
- Big Data on AWS – View the comprehensive portfolio of big data services in addition to links to other resources such AWS big data partners, tutorials, articles, and AWS Marketplace offerings on big data solutions. Contact us if you need any help.
- AWS Big Data Blog – The blog features real life examples and ideas updated regularly to help you collect, store, clean, process, and visualize big data.
- Big Data Test Drives – Explore the rich ecosystem of products designed to address big data challenges using AWS. Test Drives are developed by AWS Partner Network (APN) Consulting and Technology partners and are provided free of charge for education, demonstration, and evaluation purposes.
- Big Data on AWS – The Big Data on AWS course introduces you to cloud-based big data solutions and Amazon EMR. We show you how to use Amazon EMR to process data using the broad ecosystem of Hadoop tools like Pig and Hive. We also teach you how to create big data environments, work with DynamoDB and Amazon Redshift, understand the benefits of Amazon Kinesis Streams, and leverage best practices to design big data environments for security and cost-effectiveness. Amazon Web Services.
- Big Data Customer Case Studies – Learn from the experience of other customers who have built powerful and successful big data platforms on the AWS Cloud.

Document Revisions

Date	Description
November 2020	First publication