



THE GEORGE
WASHINGTON
UNIVERSITY

WASHINGTON, DC

1

INTRODUCTION TO BIG DATA AND ANALYTICS
CSCI 6444
CLOUD COMPUTING AND BIG DATA
PLATFORMS

Prof. Roozbeh Haghazadeh

Slides Credit:

Prof. Roozbeh Haghazadeh

OUTLINE – WEEKS 4

- Introduction
- On-premises Architecture
- Challenges
- Cloud Solutions
- AWS & GCP Services

INTRODUCTION

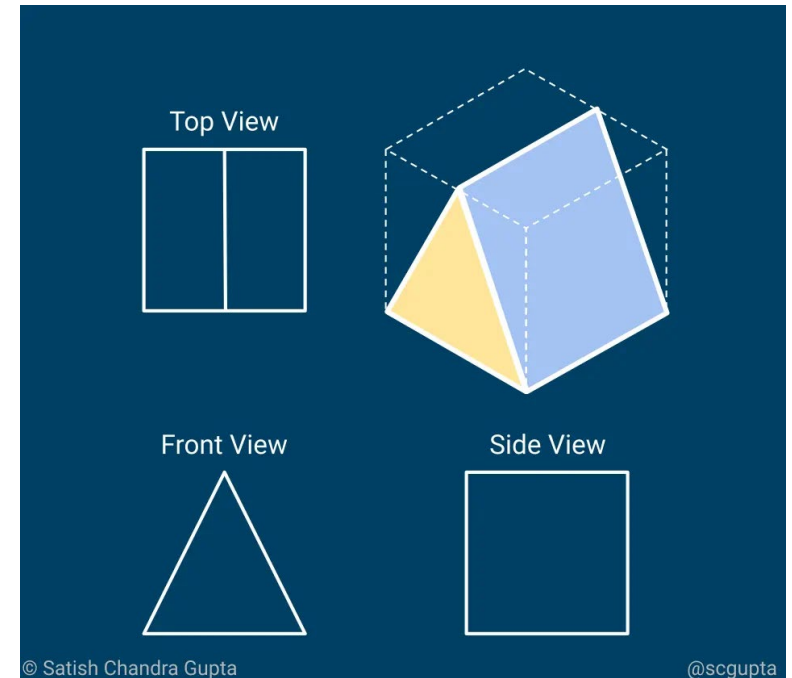
INTRODUCTION TO DATA PIPELINES

Definition:

- A data pipeline is a set of data processing elements connected in a series, where the output of one element is the input of the next. Just as a physical pipeline transports water or oil, a data pipeline transports data from one system to another.
- **Purpose:** The primary purpose of a data pipeline is to ensure a smooth and organized flow of data from its source to its endpoint (e.g., from a data collection to a data warehouse or from logs to an analytics system).
- **Automation:** Pipelines are often automated, moving data without manual intervention, ensuring efficiency and consistency.
- **Transformation:** As data travels through the pipeline, it can be cleansed, transformed, enriched, and aggregated, making it more suitable for analysis.

PERSPECTIVE

- There are three stakeholders involved in building data analytics or machine learning applications:
 - Data scientists: **find** the *most robust* and computationally *least expensive* model for a given problem using available data
 - Engineers: **build** things that *others can depend on*; to innovate either by building *new things* or finding *better* ways to build existing things that function *24x7* without much human intervention.
 - Business managers: **deliver** *value* to customers; science and engineering are means to that end.



ENGINEERING PERSPECTIVE

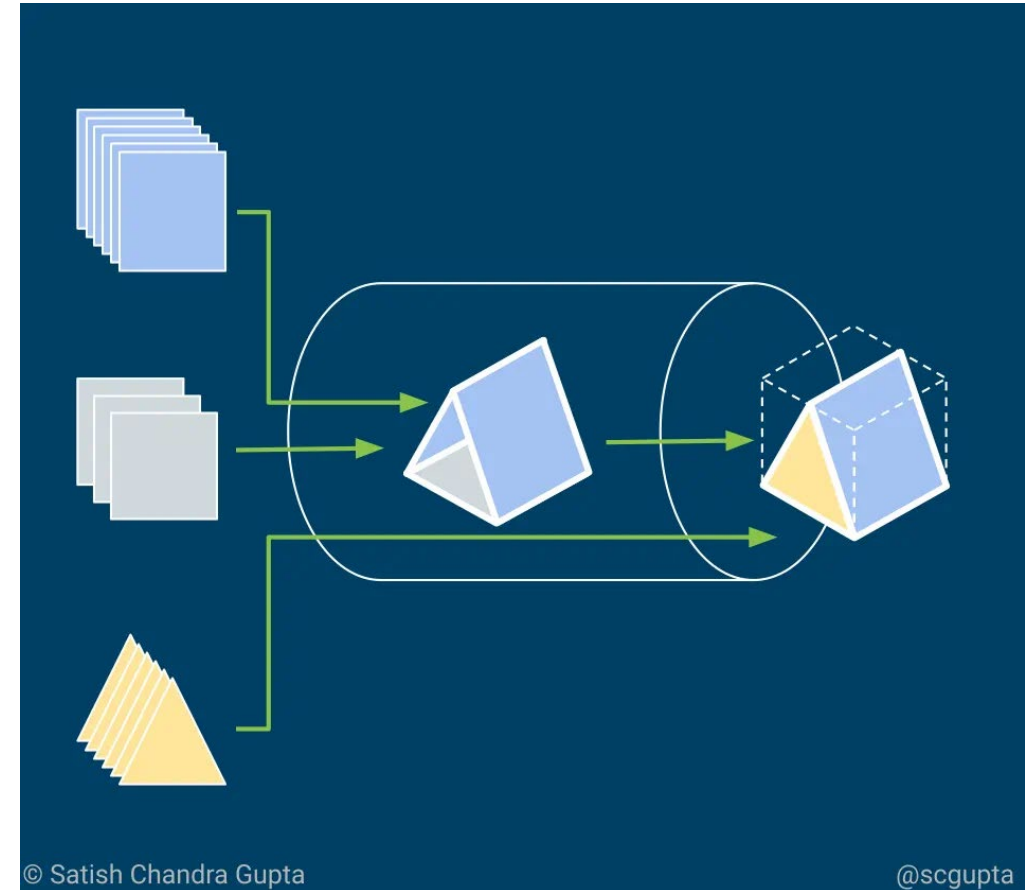
- **Accessibility:** data being easily accessible to data scientists for hypothesis evaluation and model experimentation, preferably through a query language.
- **Scalability:** the ability to scale as the amount of ingested data increases, while keeping the cost low.
- **Efficiency:** data and machine learning results being ready within the specified latency to meet the business objectives.
- **Monitoring:** automatic alerts about the health of the data and the pipeline, needed for proactive response to potential business risks.

PIPELINE

- A data pipeline stitches together the end-to-end operation consisting of capturing the data, transforming it into insights, training a model, delivering insights, applying the model whenever and wherever the action needs to be taken to achieve the business goal.

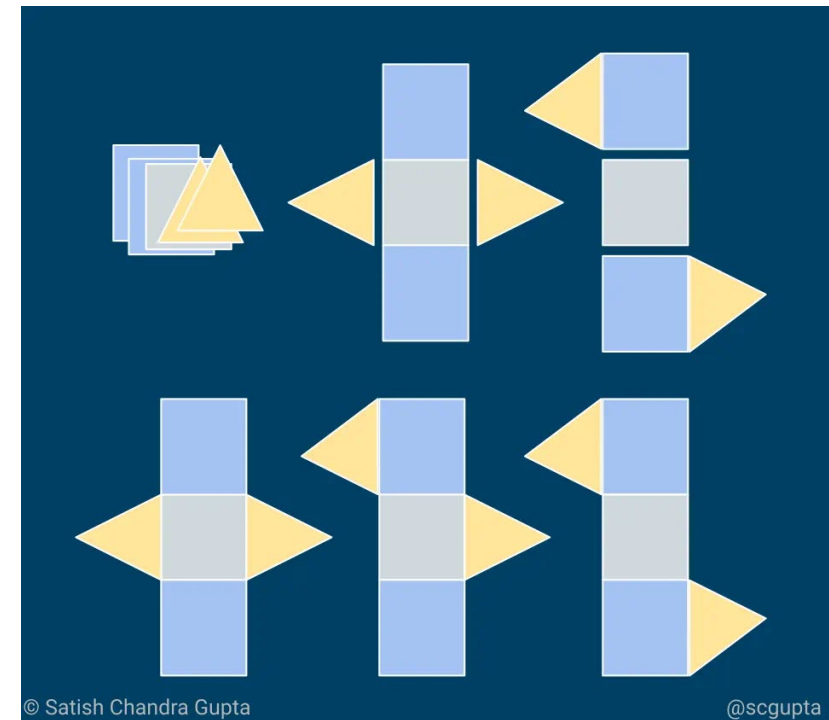
Data is the new oil. It's valuable, but if unrefined it cannot really be used. It has to be changed into gas, plastic, chemicals, etc. to create a valuable entity that drives profitable activity; so must data be broken down, analyzed for it to have value.

— Clive Humby, UK Mathematician and architect of Tesco's Clubcard



POSSIBILITIES

- Architecture is a trade-off between performance and cost. There are six options shown in this image to make a triangular tent shape.

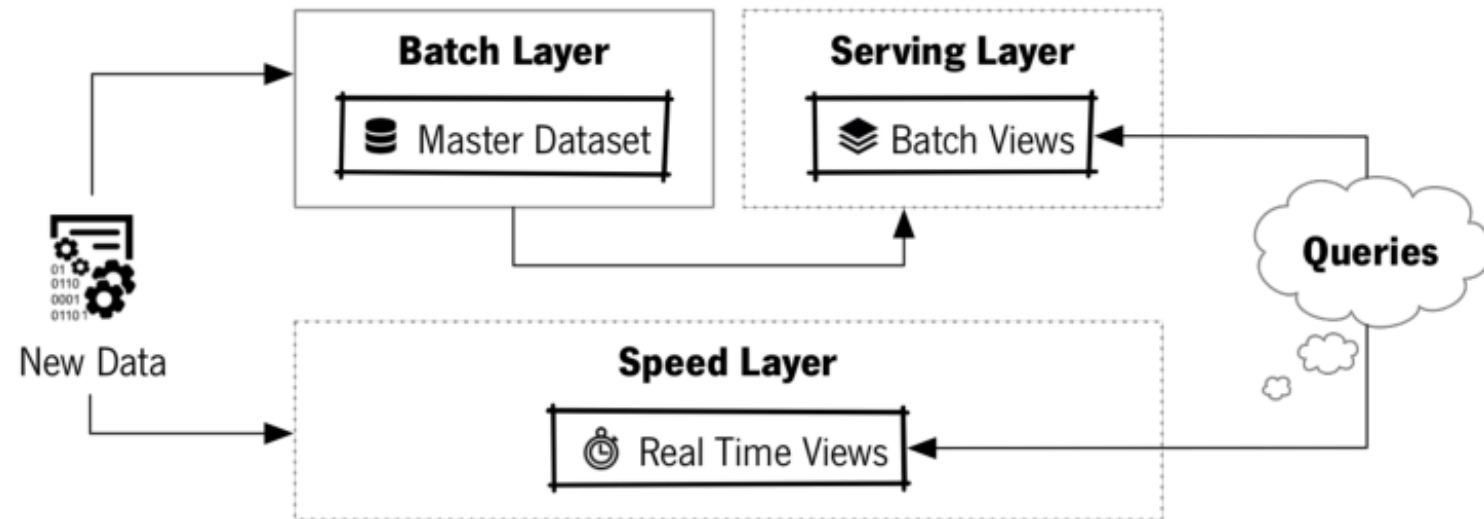


BIG DATA PIPELINE ARCHITECTURE

- the technically best option may not necessarily be the most suitable solution in production. You must carefully examine your requirements:
 - Do you need real-time insights or model updates?
 - What is the staleness tolerance of your application?
 - What are the cost constraints?
- Based on the answers to these questions, you have to balance the batch and the stream processing in the Lambda architecture to match your requirements of throughput and latency. Lambda architecture consists of three layers:
 - **Batch Layer:** offers high throughput, comprehensive, economical map-reduce batch processing, but higher latency.
 - **Speed Layer:** offers low latency real-time stream processing, but is costlier and may overshoot memory limit when data volume is high.
 - **Serving Layer:** The output from high throughput batch processing, when ready, is merged with the output of the stream processing to provide comprehensive results in the form of pre-computed views or ad-hoc queries.

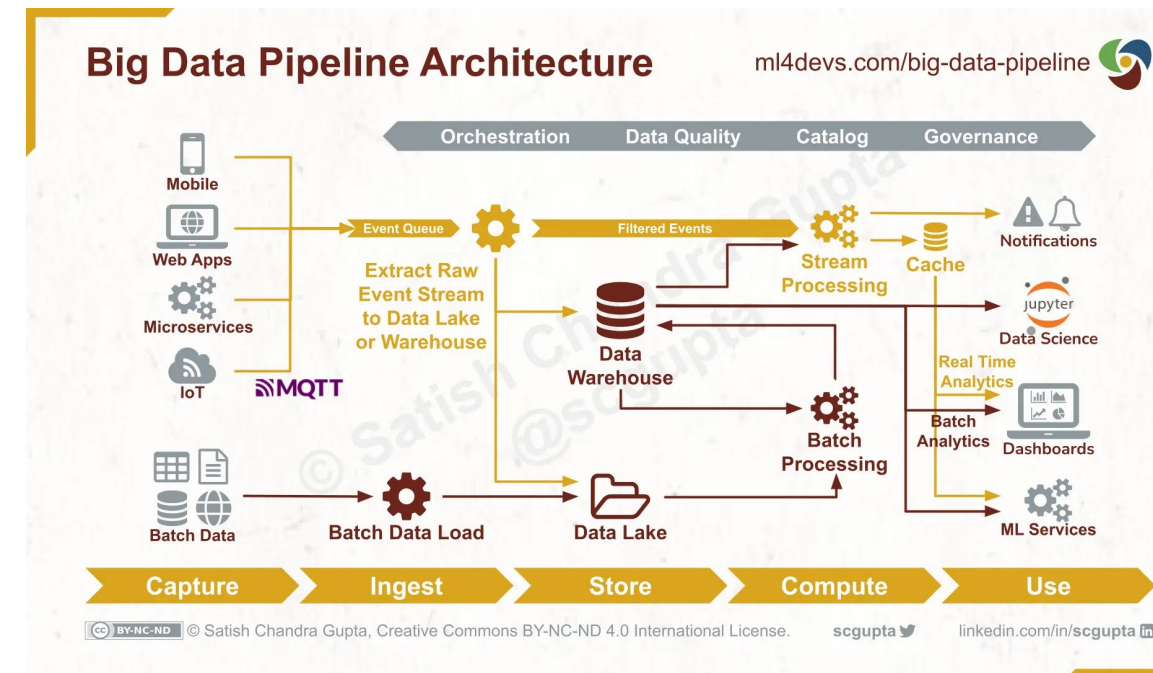
LAMBDA ARCHITECTURE

- Lambda architecture is a data deployment model for processing that consists of a traditional batch data pipeline and a fast streaming data pipeline for handling real-time data. In addition to the batch layer and speed layers, Lambda architecture also includes a data serving layer for responding to user queries.



LAMBDA ARCHITECTURE

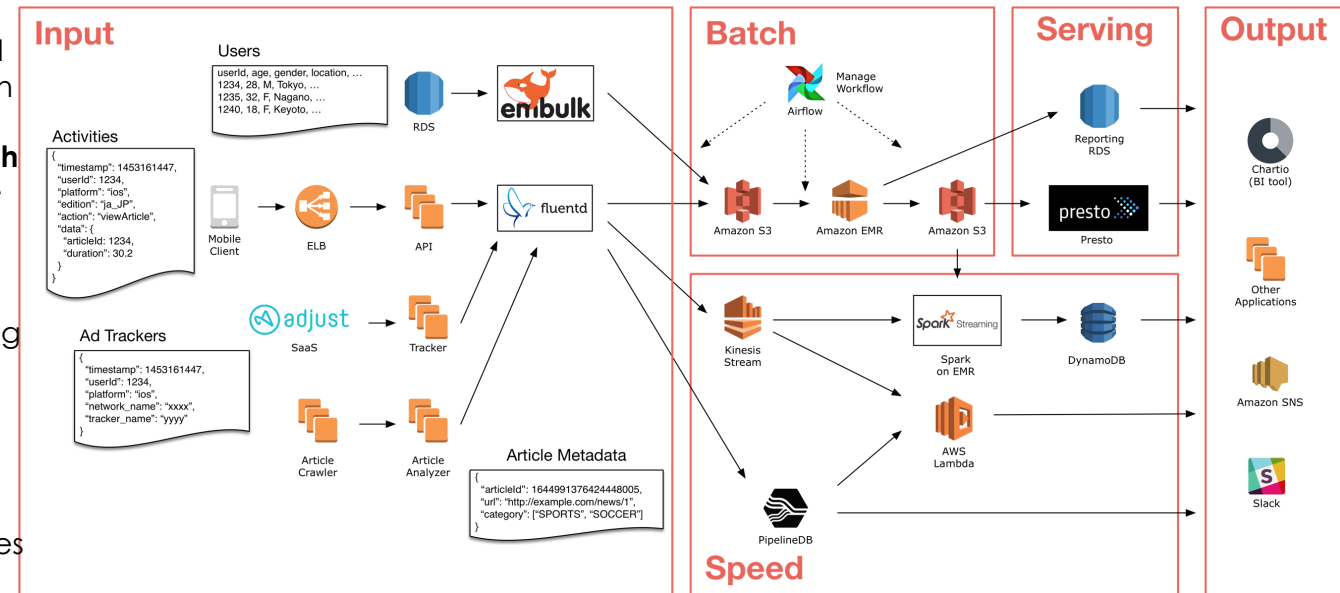
- **Orchestration:** Data pipelines are complex and have several parts forming a Directed Acyclic Graph (DAG). Pipeline Orchestration is to ensure that these parts are run in right order. All required inputs for a part must have already been computed before running apart.
- **Data Quality:** Checking the statistical distribution, outliers, anomalies, or any other tests required at each part of the data pipeline.
- **Catalog:** Data Catalog provides context for various data assets (e.g. tables in data warehouse, events in data lake, topics in message queue). It creates and manages metadata and schema of the data assets so that data engineers and data scientists can understand it better.
- **Data Governance:** Policies and processes to follow throughout the lifecycle of the data for ensuring that data is secure, anonymised, accurate, and available.



HOW SMARTNEWS BUILT A LAMBDA ARCHITECTURE ON AWS TO ANALYZE CUSTOMER BEHAVIOR AND RECOMMEND CONTENT¹²

- “SmartNews is a machine learning-based news discovery app that delivers the very best stories on the Web for more than 18 million users worldwide.”

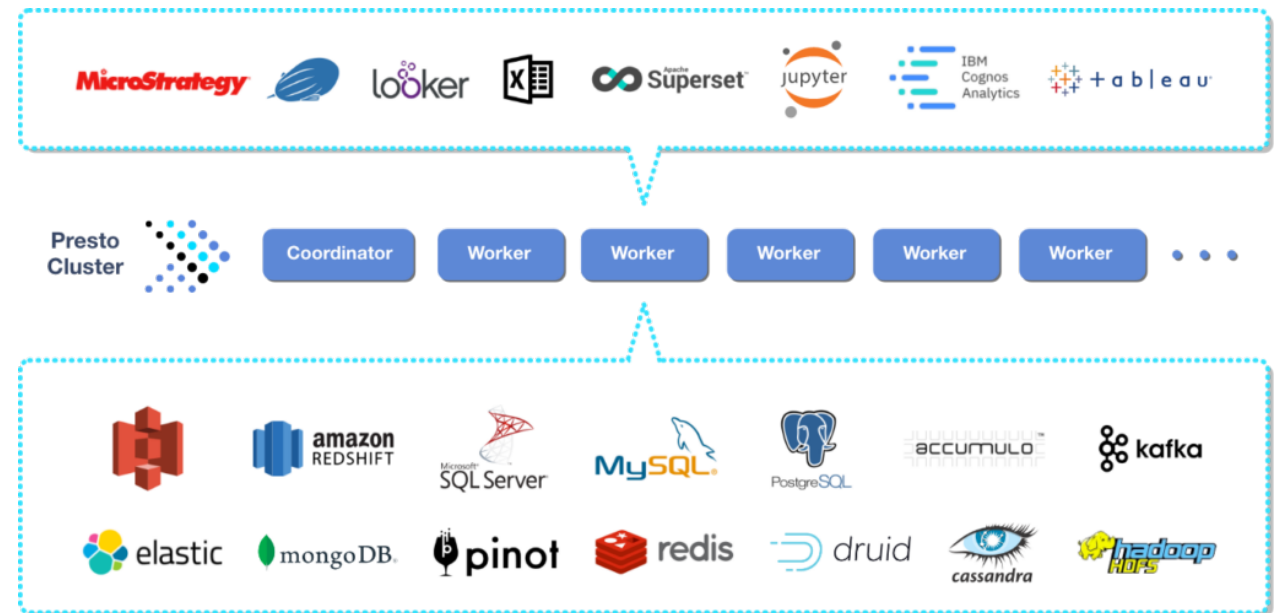
- The news team at SmartNews uses data as input to their machine learning algorithm for delivering the very best stories on the Web.
- The product team relies on data to run various A/B tests, to learn about how our customers consume news articles, and to make product decisions.
- Storage/compute split:** SmartNews decoupled storage (Amazon S3) and streaming storage (Amazon Kinesis) from compute so multiple teams can spin up heterogeneous Spark/Hive clusters independently.
- Dual paths (Lambda architecture):** Raw events are fanned out to a **batch layer** (EMR + Hive/Spark for ETL, features, precomputed views, heavy use of Spot Instances) and a **speed layer** (Spark Streaming over Kinesis) for low-latency signals.
- Ingestion tooling:** Mobile logs go through **Fluentd** to S3 and Kinesis; relational data in **Amazon RDS** is exported with **Embulk** (with field masking support).
- Ops & monitoring:** They instrument Fluentd with Datadog (flow counters, DogStatsD) and to avoid data loss on scale-in. **Airflow** manages ETL dependencies instead of cron.
- Near-real-time ML join:** The speed layer **joins Kinesis user activity** with **offline user clusters** (from batch ML) inside Spark Streaming to rank articles within minutes. A shared Hive metastore makes batch tables queryable by streaming jobs.
- Consumption:** BI is served via **Charlio** (supports RDS, Presto, Redshift, OpenSearch), enabling non-engineers to explore dashboards quickly.



<https://aws.amazon.com/blogs/big-data/how-smartnews-built-a-lambda-architecture-on-aws-to-analyze-customer-behavior-and-recommend-content/>

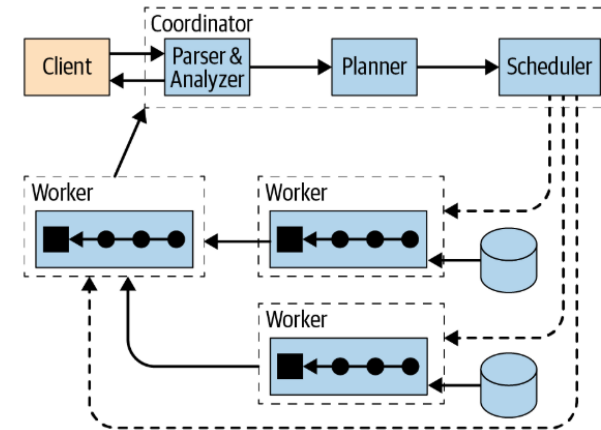
WHAT IS PRESTO?

- Presto is a distributed SQL **query engine** (not a database). It lets you run ANSI SQL across many data sources without moving the data.
- What it is:** an open-source, distributed SQL engine built for fast, interactive analytics on large data (originated at Facebook).
- How it works:** MPP architecture (coordinator + workers) that executes queries over data where it lives; no storage layer of its own.
- Federation via connectors:** query lakes (S3/HDFS/Parquet/ORC) and many systems (MySQL, Postgres, Kafka, MongoDB, etc.) in a single SQL query.
- Typical use:** ad-hoc BI/analytics, “SQL over the lake,” and cross-source joins without heavy ETL.
- Ecosystem note:** Trino is a 2019 fork of Presto that many services now use; both remain distributed SQL query engines with similar goals.

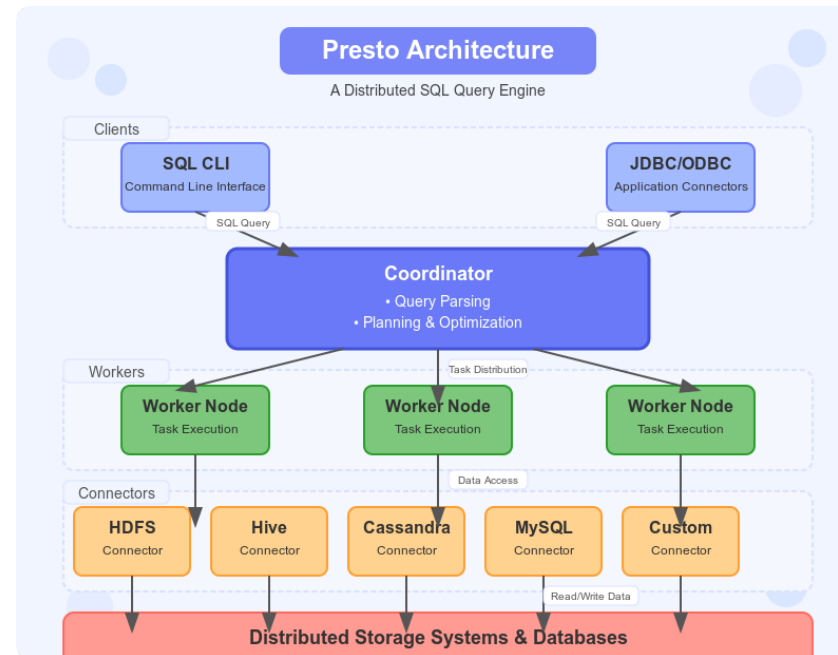


DISTRIBUTED DATA ENGINES

- **What they are:** Presto & Trino are open-source, distributed ANSI-SQL **query engines** that run without their own storage—built for interactive analytics and federation across many sources.
- **Why they matter:** Query data where it lives (S3/HDFS/object stores, RDBMS, Kafka, etc.) and **join across heterogeneous sources** with connectors, perfect for “SQL over the lake/mesh” without heavy ETL.

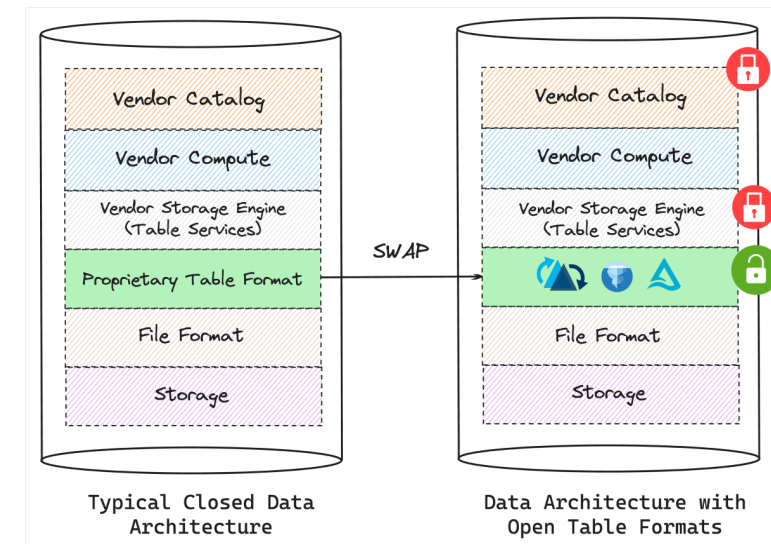
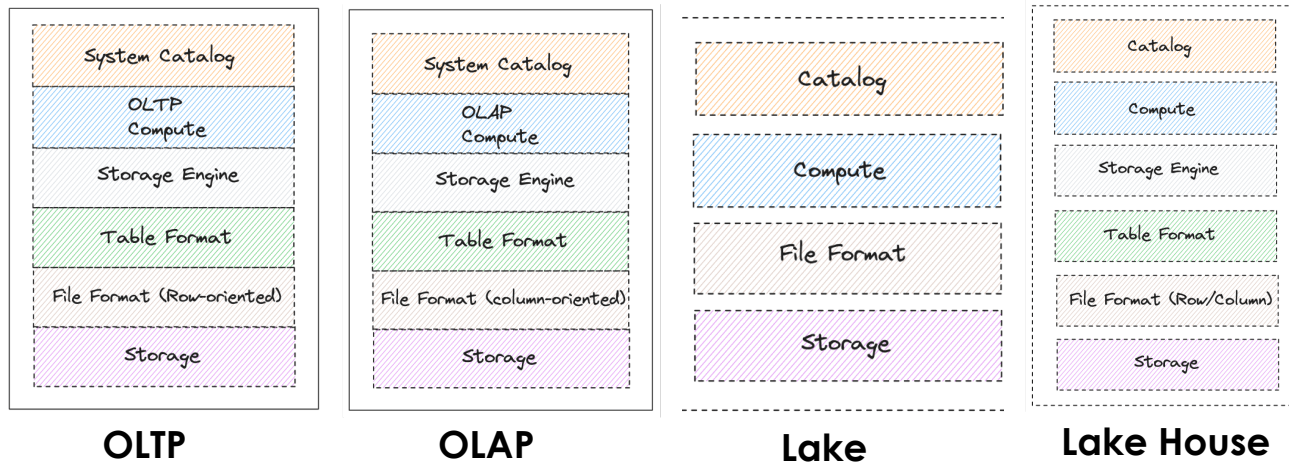
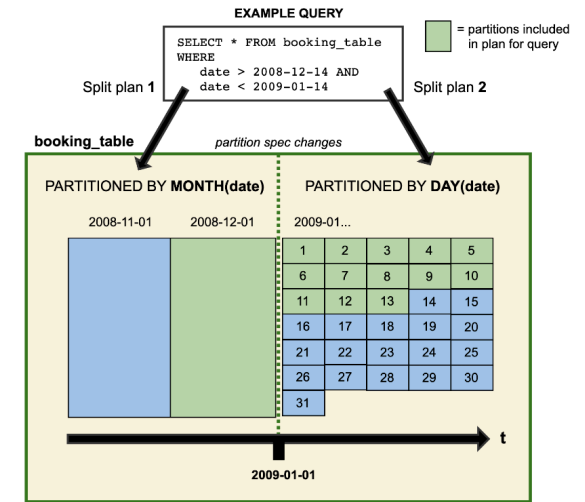


- **Examples of distributed query engines include:**
- **Apache Hive:** Initially built on top of Hadoop for batch processing, Hive provides a SQL-like interface to query data stored in various databases and file systems that integrate with Hadoop.
- **Presto:** An open-source distributed SQL query engine designed for interactive analytic queries against data sources of all sizes ranging from gigabytes to petabytes.
- **Apache Spark SQL:** Part of the Apache Spark big data framework, Spark SQL provides a distributed query engine capable of processing structured data in Spark programs.
- **Google BigQuery:** A fully-managed, serverless data warehouse that enables scalable, cost-efficient, and fast analysis of big data with SQL and is tightly integrated with Google Cloud Storage.
- **Amazon Athena** — managed service whose engine aligns closely with Trino/Presto (Engine v3 functions & integration path).

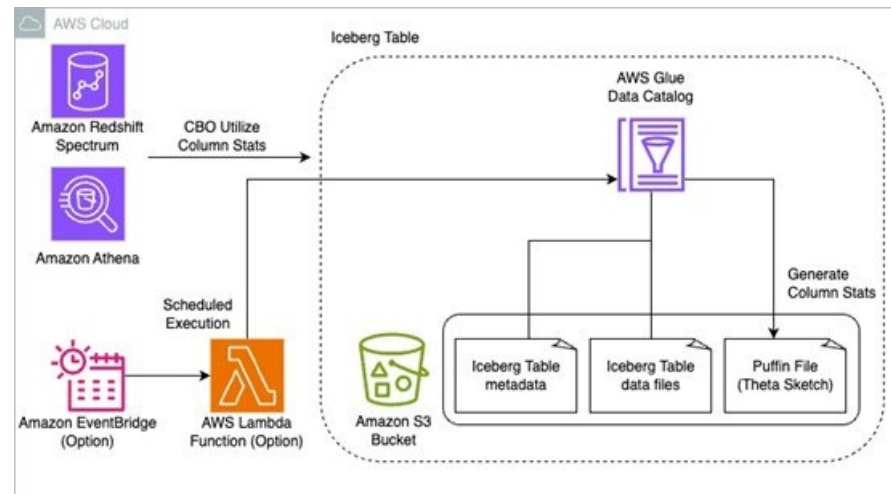
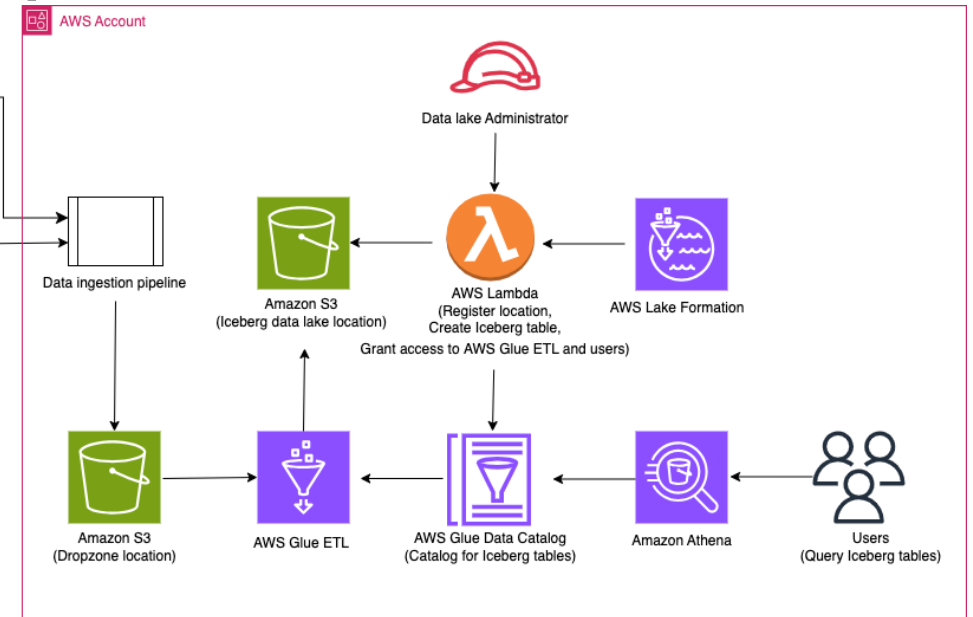
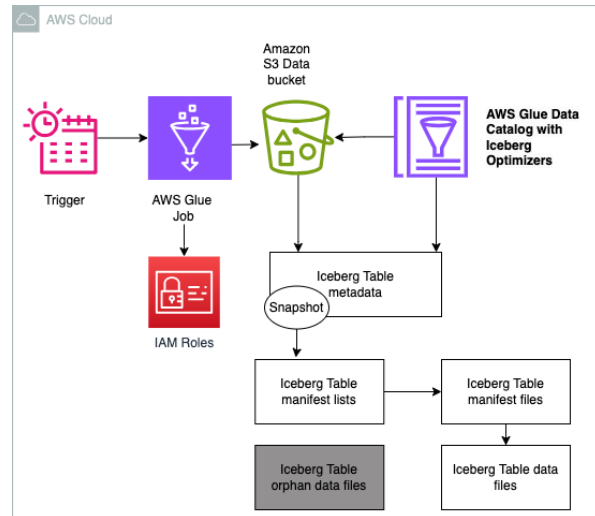


OPEN TABLE FORMATS

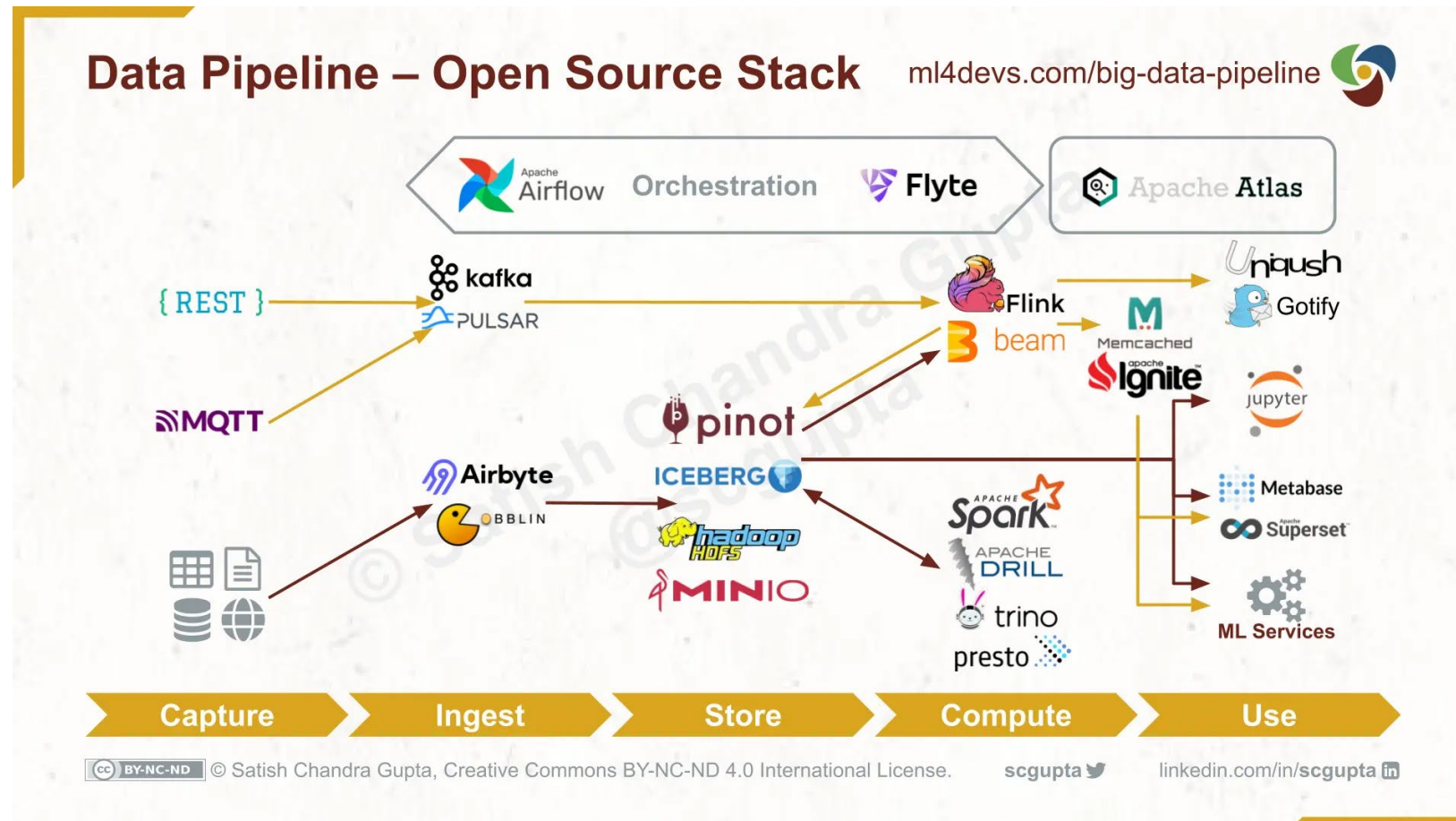
- **What they are:** define how data files within a data lake are organized into tables, providing a layer of abstraction and management over raw files (like Parquet or ORC).
- **Why they matter:** They manage metadata (schema, partitions, file locations), enable features like ACID transactions, schema evolution, time travel, and efficient data skipping.
- **Role:** They provide a structured, reliable, and performant way to interact with data stored in a data lake, making it behave more like a traditional database table. They are vendor-agnostic and allow different engines to access the same data consistently.



STAND UP ICEBERG ON S3 (GLUE CATALOG + ATHENA)



BUILDING DATA PIPELINES WITH OPEN SOURCE STACK



BUILDING DATA PIPELINES WITH OPEN SOURCE STACK

- **HTTP / MQTT Endpoints** for ingesting data, and also for serving the results. There are several frameworks and technologies for this.
- **Pub/Sub Message Queue** for ingesting high-volume streaming data. [Kafka](#) is currently the de-facto choice. It is battle-proven to scale to a high event ingestion rate.
- **Low-Cost High-Volume Data Store** for data lake (and data warehouse), [Hadoop HDFS](#) or cloud blob storage like [AWS S3](#).
- **Query and Catalog Infrastructure** for converting a data lake into a data warehouse, Apache [Hive](#) is a popular query language choice.
- **Map-Reduce Batch Compute** engine for high throughput processing, e.g. [Hadoop Map-Reduce](#), Apache [Spark](#).
- **Stream Compute** for latency-sensitive processing, e.g. Apache [Storm](#), Apache [Flink](#). Apache [Beam](#) is emerging as *the* choice for writing the data-flow computation. It can be deployed on a Spark batch runner or Flink stream runner.
- **Machine Learning Frameworks** for data science and ML. [Scikit-Learn](#), [TensorFlow](#), and [PyTorch](#) are popular choices for implementing machine learning.
- **Low-Latency Data Stores** for storing the results. There are many well-established [SQL vs. NoSQL](#) choices of data stores depending on data type and use case.
- **Deployment** orchestration options are [Hadoop YARN](#), [Kubernetes](#) / [Kubeflow](#).

ASSIGNMENT 1

ASSIGNMENT 1

Setting up an open-source modern data stack

- Pick a data set
- Create 2 pipelines up to a database and data warehouse with different technologies
- You can download VM-Ware and setup VMs on the team members' computers to build the pipeline
- Please check the following links as an example:
 - <https://medium.com/@Cartelis/setting-up-an-open-source-modern-data-stack-6dd41094cd2f>
 - <https://towardsdatascience.com/building-an-end-to-end-open-source-modern-data-platform-c906be2f31bd>
- Or even you can use AWS based services.
- Each team needs to submit a technical report to compare the pipelines
 - Explain the data size
 - Compare the performance
 - Cons and Pros of each tech stack
- Each team need to create a video to demo the pipeline from data source to loaded data into the DB

CHALLENGES

CHALLENGES

Based on the explained pipeline what are the challenges?

SCALABILITY ISSUES

- Scalability refers to the system's capability to handle an increase in workload or demand by adding resources. On-premises systems often have a maximum limit in terms of how much they can be scaled up due to physical, logistical, and financial constraints.
- **Physical Limitations:** There's a physical limit to how many servers or storage devices you can fit into a data center. Once you reach this limit, you'd need to consider building or renting another facility, which is a significant undertaking.
- **Hardware Availability:** Even if space is available, the exact hardware models might become hard to procure, especially if they get outdated. This inconsistency in hardware can introduce compatibility and performance issues.
- **Infrastructure Dependencies:** On-premises setups might have complex interdependencies. Scaling one component might necessitate changes or upgrades to other interconnected parts, making the scaling process intricate.
- **Upfront Costs:** Scaling on-premises infrastructure requires significant capital expenditure. This includes the cost of hardware, software licenses, and the infrastructure needed to support and cool new equipment.
- **Operational Delays:** The time from deciding to scale to having a fully operational system can be lengthy. It involves procurement, shipping, setup, and testing. During this time, the business might suffer if existing systems are overburdened.
- **Manpower:** Scaling isn't just about adding new servers. It requires skilled personnel to set up, configure, and maintain these systems. Hiring or training such individuals can be time-consuming and costly.
- **Unpredictable Workloads:** For businesses with fluctuating demands, predicting the exact scaling requirements can be challenging. Overprovisioning leads to wasted resources, while underprovisioning can affect performance and user experience.

MAINTENANCE OVERHEAD

- **Software Updates:**
 - **Frequency:** Software, whether it's the operating system, databases, or applications, often requires frequent updates. These updates can bring new features, performance improvements, or security patches.
 - **Downtime:** Some updates might necessitate system restarts or temporarily halt services. This can disrupt business operations, especially if not well-planned.
 - **Incompatibilities:** Updates might sometimes introduce incompatibilities with other software or systems, potentially leading to issues or service interruptions.
 - **Rollbacks:** If an update causes issues, there might be a need to revert to a previous version, which can be time-consuming and challenging.
- **Hardware Repairs & Replacement:**
 - **Wear & Tear:** Physical hardware components have a lifespan. Over time, parts like hard drives, power supplies, or fans can wear out and fail.
 - **Replacement Time:** Identifying a failing component, procuring a replacement, and then swapping it out can lead to prolonged system downtimes, especially if backups or redundancies aren't in place.
 - **Costs:** Regular replacements and repairs come with direct costs for parts and potentially service fees if external technicians are involved.
 - **Inventory Management:** Maintaining a stock of spare parts for different hardware components requires careful inventory management and can tie up capital.
- **Physical Environment:**
 - **Cooling:** Data centers need efficient cooling to ensure hardware doesn't overheat. Cooling systems themselves can break down, leading to potential hardware risks.
 - **Power:** Fluctuations or outages can harm equipment or lead to data loss. Redundant power supplies or generators add to the complexity and cost.
 - **Space:** As the infrastructure grows, more physical space is required. This can lead to increased costs and logistical challenges.
- **Skilled Personnel:**
 - **Diverse Skills Needed:** Different systems and hardware components require different expertise. Maintaining a diverse IT team that can address all potential issues is expensive.
 - **Training:** As technology evolves, IT personnel need ongoing training to stay current, adding to costs and time commitments.

HIGH CAPITAL EXPENDITURE

- **Infrastructure Costs:** The capital required for initial setup
 - **Server Costs:** Investing in servers, especially high-performance ones, can be quite expensive. This includes the primary servers for applications and databases, backup servers, and redundant systems for high availability.
 - **Networking Equipment:** Ensuring fast and reliable connectivity requires investments in switches, routers, firewalls, and other networking gear.
 - **Facility Costs:** Constructing or leasing a data center or server room with the necessary security and environmental controls is a significant cost component.
 - **Licensing:** Software licenses, especially for enterprise-grade solutions, can be a major part of the initial expenditure. This includes OS licenses, database management systems, and other specialized software.
- **Power and Cooling:** Additional overheads in keeping servers operational
 - **Electricity Costs:** Servers and other equipment consume a lot of power, especially when operational 24/7. These costs recur monthly and can be substantial for a large data center.
 - **Cooling Systems:** Servers generate heat, necessitating cooling solutions. Investing in and operating cooling systems like HVAC units or specialized data center cooling solutions adds to the costs.
 - **Redundancy:** Ensuring uninterrupted operations requires backup power solutions, such as UPS units and generators. These not only have initial costs but also maintenance and fuel costs.
- **Hardware Replacements:** Recurring costs associated with wear and tear
 - **Lifecycle:** Most hardware components have a lifecycle after which their performance degrades, or they become obsolete. Replacing these components regularly is essential for maintaining operational efficiency.
 - **Compatibility:** Upgrading one component might necessitate changes to others due to compatibility issues. For instance, a new server might require newer networking gear to fully utilize its capabilities.
 - **Inventory:** Keeping a stock of replacement parts or entire servers can be a wise strategy to minimize downtime. However, this means locking capital in inventory that might or might not be used.

DISASTER RECOVERY

- **Recovery Point Objective (RPO):**
 - **Definition:** RPO refers to the maximum acceptable amount of data loss measured in time. It answers: "How much data can I afford to lose before it severely impacts my business?"
 - **Challenge:** In on-premises setups, achieving a low RPO might require more frequent backups, leading to increased storage costs and potential performance impacts.
- **Recovery Time Objective (RTO):**
 - **Definition:** RTO is the duration of time within which data must be restored to ensure business continuity. It answers: "How quickly must I recover my data to keep my business running?"
 - **Challenge:** Faster recovery times necessitate more sophisticated (and often more expensive) backup and recovery solutions.
- **Infrastructure Damage:**
 - **Challenge:** If the physical infrastructure, such as servers or data centers, is damaged due to incidents like fires, floods, or other disasters, restoring data becomes even more complex. It's not just about data recovery but also about quickly procuring and setting up new hardware.
- **Data Corruption:**
 - **Challenge:** Sometimes, the issue isn't just data loss but data corruption. Identifying the cause of corruption and restoring to a clean state without reintroducing the problem can be intricate.
- **Testing:**
 - **Challenge:** Regularly testing disaster recovery plans is essential to ensure they work when needed. However, testing can be resource-intensive and might disrupt regular operations.

ADAPTABILITY CHALLENGES

- **Inflexibility of Legacy Systems:**
 - **Outdated Technology:** Legacy systems are often built on older technologies that might not support newer functionalities or integration with modern software and platforms. This can hamper innovation and slow down the adoption of newer, more efficient tools.
 - **Integration Hurdles:** As new tools and systems emerge, integrating them with older legacy systems can be a challenge due to compatibility issues, different data formats, or communication protocols.
 - **Customized Solutions:** Over time, many legacy systems have been highly customized to cater to specific business needs. While this might have provided a short-term solution, it can create long-term challenges, as these custom modifications make upgrades or migrations more complex.
- **Challenges in Technology Adoption:**
 - **Delayed Implementation:** Adopting new technologies in an on-premises environment often requires lengthy planning, procurement, setup, and testing phases. This slows down the ability to harness new tech trends quickly.
 - **High Upfront Costs:** While cloud solutions often follow a pay-as-you-go model, on-premises solutions require significant capital expenditure upfront, making it a substantial financial commitment to adopt new technologies.
 - **Resource Allocation:** New technology adoption demands skilled personnel for setup, configuration, and maintenance. Finding or training such resources can take time and pull them away from other critical tasks.
- **Difficulties in Downscaling:**
 - **Fixed Investments:** With on-premises setups, organizations have made fixed investments in infrastructure. Unlike cloud environments where you can scale down and reduce costs, in on-premises scenarios, the investment remains sunk even if the actual demand decreases.
 - **Operational Costs:** Even if an organization wants to reduce its server usage, the operational costs, such as maintenance, power, and cooling, do not decrease proportionally.
 - **Redundancies:** As business needs evolve, certain legacy systems might become redundant. However, phasing them out can be challenging due to dependencies or the potential disruption they might cause.
- **Global Availability**

SECURITY AND COMPLIANCE CHALLENGES IN ON-PREMISES NETWORKS:

Infrastructure Security:

- **Physical Security:** Unlike cloud providers who maintain data centers with strict physical access controls, on-prem infrastructures are often housed within the company's facilities. This means businesses are responsible for all physical security measures such as surveillance, access controls, and security personnel.
- **Network Vulnerabilities:** Without the vast resources of cloud providers, on-prem setups might lack advanced intrusion prevention and detection systems, making them potentially more susceptible to attacks.
- **Hardware Failures:** On-prem infrastructure is susceptible to hardware failures. Malfunctions can pose direct risks to data integrity and can be exploited by malicious actors in certain scenarios.

Data Security:

- **Backup and Recovery:** Companies are responsible for their own backup solutions. If not done correctly, there's a risk of data loss. Moreover, backup data can be a target for theft or ransomware.
- **Encryption:** Implementing and managing encryption (both in transit and at rest) typically falls entirely on the company, which might not always have the expertise or resources to ensure best practices.

Patch Management:

- **Software Vulnerabilities:** Ensuring software is up-to-date is crucial. On-prem setups often require manual patch management, leading to potential delays in addressing vulnerabilities.
- **Legacy Systems:** Older systems that aren't regularly updated or have reached end-of-life can have unpatched vulnerabilities, posing security risks.

Compliance:

- **Regulatory Landscape:** Complying with a rapidly evolving regulatory landscape is challenging. From GDPR to HIPAA, businesses need to ensure data practices comply with applicable regulations.
- **Audit Preparedness:** Unlike cloud providers who often have compliance certifications and undergo regular audits, on-prem solutions require the business to handle all audits. This can be resource-intensive and may expose gaps in compliance.
- **Data Sovereignty:** For global businesses, data might need to be stored in specific jurisdictions. On-prem solutions might complicate this if the business doesn't have physical locations in the required regions.

USEFUL SERVICES ON AWS TO ADDRESS THE CHALLENGES

DATA INGESTION:

- **Amazon Kinesis:** A platform for streaming data on AWS, making it easy to collect, process, and analyze real-time, streaming data.
 - *Components:* Kinesis Data Streams, Kinesis Data Firehose, Kinesis Video Streams, and Kinesis Data Analytics.
- **AWS Direct Connect:** Provides a dedicated network connection from your on-premises data centers to AWS.
- **AWS Snowball:** A petabyte-scale data transport solution that uses secure appliances to transfer large amounts of data into and out of AWS.
- **AWS DataSync:** An online data transfer service that simplifies, automates, and accelerates copying large amounts of data to and from AWS storage services.

AMAZON KINESIS

- **Definition:** Amazon Kinesis offers real-time data streaming services that can handle large amounts of fast-moving data.
- **Components:**
 - **Kinesis Data Streams:** Capture and process real-time data.
 - **Kinesis Data Firehose:** Load data streams to other AWS services.
 - **Kinesis Data Analytics:** Analyze data streams with SQL or Apache Flink.
 - **Kinesis Video Streams:** Process and analyze video streams.
- **Use Cases:** Real-time analytics, logging, mobile app telemetry, and more.



DATA STORAGE

- **Amazon S3 (Simple Storage Service):** Highly scalable and durable object storage. It can store and retrieve any amount of data.
 - **Durability & Availability:** S3 is designed for 99.999999999% (11 9's) of durability, meaning your data is extremely safe.
 - **Scalability:** No storage limit. Users can store as much data as they need.
 - **Security:** Offers robust encryption features, access controls, and compliance certifications.
 - **Integration:** Easily integrates with other AWS services.
- **Amazon Glacier & Glacier Deep Archive:** For archival storage solutions that are cheaper than regular S3 storage but have longer retrieval times.



DATA PROCESSING

- **Amazon EMR (Elastic MapReduce):** Provides a managed Hadoop framework that makes it easy, fast, and cost-effective to process vast amounts of data. Also supports other popular frameworks like Spark and Presto.
- **AWS Glue:** A fully managed ETL (Extract, Transform, Load) service that prepares and loads your data for analytics. It also provides a data catalog.
- **Amazon Athena:** An interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL.

AMAZON EMR

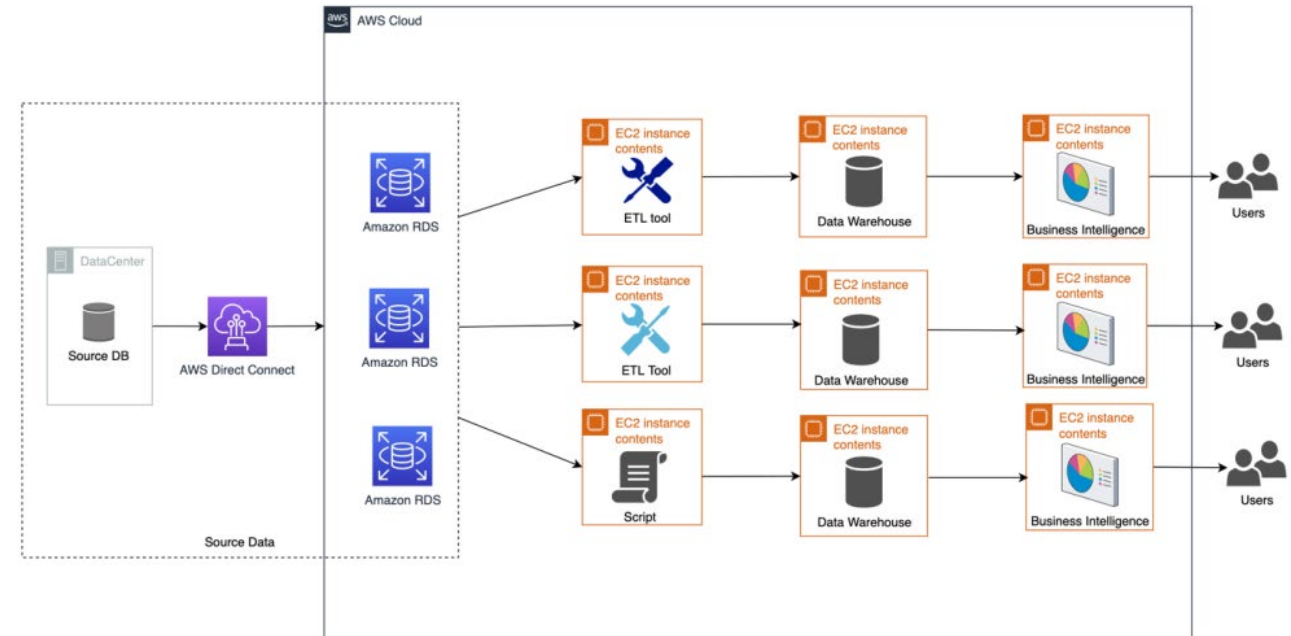
- **Definition:** Amazon EMR is a cloud-native big data platform, supporting processing frameworks like Hadoop, Spark, and others.
- **Key Features:**
 - **Scalability:** Easily resize clusters and adjust compute resources.
 - **Flexibility:** Supports multiple big data frameworks.
 - **Cost-Effective:** Pay for what you use, and leverage spot instances for savings.
 - **Integration:** Integrates well with other AWS services, especially data storage services like S3.



amazon
EMR

AWS RDS (RELATIONAL DATABASE SERVICE)

- AWS RDS is a managed relational database service that provides multiple database product options, ease of management, and scalability for database deployments.
- **Key Features:**
 - **Database Engines:** Supports several relational database engines, including MySQL, PostgreSQL, MariaDB, Oracle, and Microsoft SQL Server.
 - **Management & Maintenance:** Automates time-consuming tasks such as backups, software patching, and hardware provisioning.
 - **Scalability:** With just a few clicks, you can scale the database's compute resources up or down.
 - **Availability & Durability:** Offers Multi-AZ deployments for failover, along with automated backups, database snapshots, and automated or manual replication.
 - **Security:** Integrates with AWS Identity and Access Management (IAM), and provides encryption at rest and in transit.
- **Use Cases:** Web and mobile applications, CMS, CRM, ERP, e-commerce, and any application that requires a relational database backend.



AMAZON AURORA

- Amazon Aurora is a relational database service developed and optimized by Amazon Web Services. It's a part of RDS and is designed to be MySQL- and PostgreSQL-compatible while offering better performance and availability.
- **Key Features:**
 - **Performance:** Aurora delivers up to 3x the performance of standard MySQL databases and 2x the performance of standard PostgreSQL.
 - **Scalability:** Aurora automatically scales up with up to 15 read replicas for read-heavy database workloads and offers Aurora Auto Scaling for adjusting the number of Aurora Replicas.
 - **Distributed & Fault-Tolerant:** Aurora automatically divides your database volume into 10GB segments spread across many disks. Each 10GB chunk of your database volume is replicated six ways, across three Availability Zones.
 - **Continuous Backup:** Aurora continuously backs up data to Amazon S3 and is designed to offer fault-tolerance against the loss of up to two copies of data without affecting write availability and up to three copies without affecting read availability.
 - **Serverless Option:** Aurora Serverless provides on-demand, auto-scaling configurations that automatically shut down during inactivity.
- **Use Cases:** Enterprise applications requiring complex transactions, sophisticated applications that need high performance and scalability, applications migrating away from commercial databases for cost efficiency.

COMPARISON

- While both RDS and Aurora are managed relational database services, Aurora is specifically built by AWS to maximize the capabilities of the AWS infrastructure. It offers performance enhancements over traditional RDS deployments.
- Aurora may be more cost-effective at scale because of its performance improvements over standard RDS configurations.
- RDS's strength lies in its broad support for various third-party database engines, while Aurora focuses on being the best-performing MySQL and PostgreSQL compatible system within AWS.

DATA WAREHOUSING

- Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud.
- **Characteristics:**
 - **Fast:** Uses columnar storage, data compression, and parallel query execution to achieve fast query performance.
 - **Scalable:** You can start with a small dataset and scale to petabytes.
 - **Integration:** Directly integrates with popular business intelligence tools and other AWS services.
 - **Secure:** Offers encryption at rest and in transit, VPCs, and more to ensure data security.

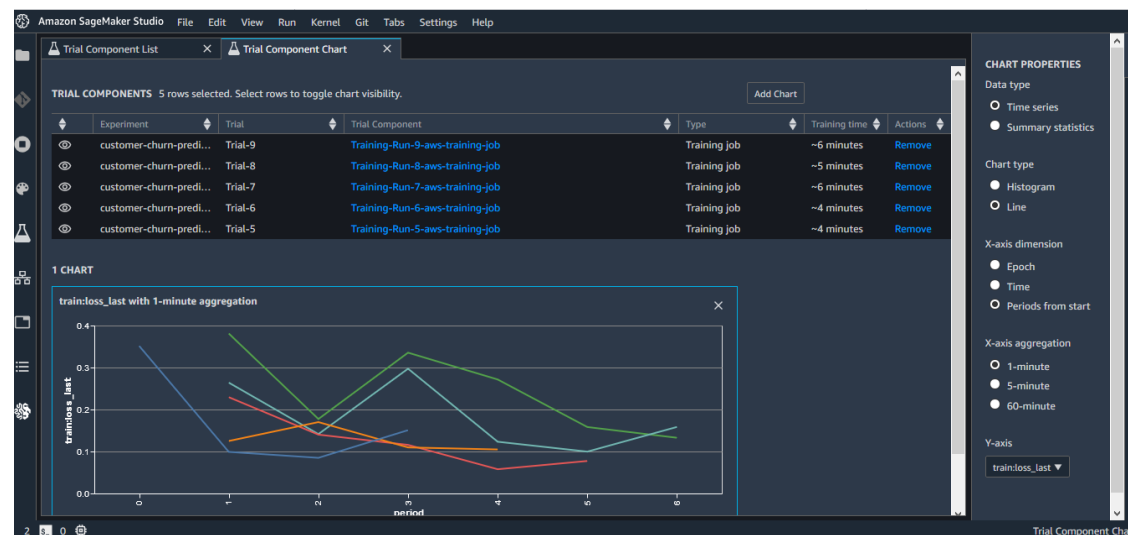
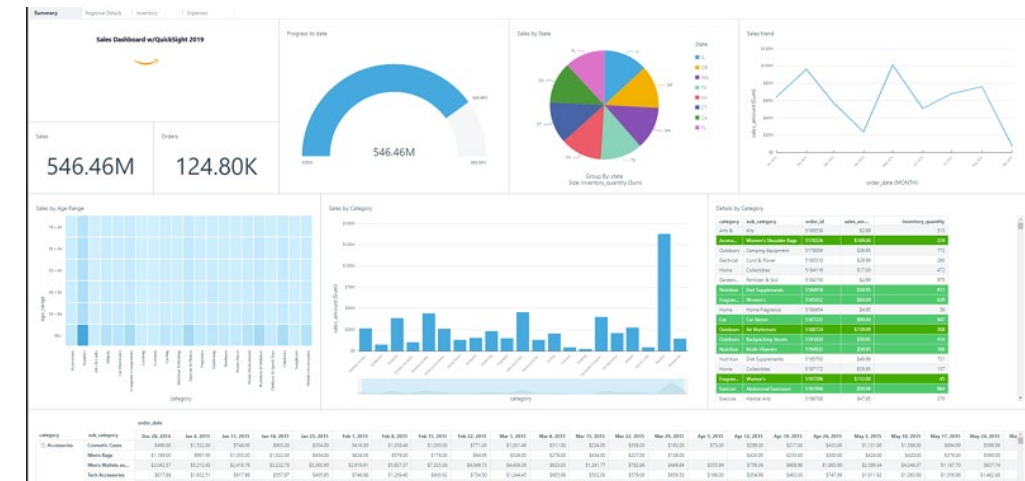


CHOOSE

- <https://aws.amazon.com/getting-started/decision-guides/databases-on-aws-how-to-choose/>

DATA ANALYSIS & VISUALIZATION

- **Amazon QuickSight:** A fully managed business intelligence service that provides insights through rich visualizations and dashboards.
- **Amazon SageMaker:** A fully managed service that allows data scientists and developers to build, train, and deploy machine learning models.



SECURITY & MANAGEMENT

- **AWS Identity and Access Management (IAM):** This lets you manage access to AWS services and resources securely.
- **Amazon Macie:** Uses machine learning to recognize sensitive data, such as personally identifiable information (PII), and provides you with dashboards and alerts.
- **AWS Key Management Service (KMS):** Allows you to create and manage cryptographic keys and control their use across a wide range of AWS services and in your applications.

AMAZON MACIE

- Amazon Macie is a fully managed data security and data privacy service that uses machine learning and pattern matching to discover, monitor, and protect sensitive data in AWS.

Key Features:

- **Sensitive Data Discovery:**
 - **Machine Learning:** Uses machine learning to recognize sensitive data, such as Personally Identifiable Information (PII) or financial data.
 - **Content Classifiers:** Macie provides built-in classifiers and also allows customers to define custom classifiers to identify patterns specific to their data.
- **Risk Assessment:**
 - **Dashboard:** Offers a centralized dashboard to monitor and rectify any potential data exposures or policy violations.
 - **Alerts:** Sends alerts when it detects an anomaly or unauthorized access.
- **Activity Monitoring:**
 - **GuardDuty Integration:** Macie integrates with Amazon GuardDuty to monitor data access activity for unusual or unauthorized behaviors.
 - **Access Patterns:** It can identify and alert on subtle deviations in data access, which could indicate potential threats.
- **Inventory and Reporting:**
 - **Data Visibility:** Provides an inventory of your S3 buckets and helps you understand where sensitive information is stored.
 - **Regulatory Reporting:** Helps in generating reports for regulatory purposes, especially handy for standards like GDPR or HIPAA.
- **Management & Integration:**
 - **AWS Native:** As a native AWS service, it integrates seamlessly with other AWS services.
 - **Policy Management:** Provides a platform to manage data protection policies and audits across AWS resources.
- **Cost Management:**
 - **Predictable Costs:** Macie provides cost estimates before running sensitive data discovery jobs, allowing users to understand and control costs.

AMAZON MACIE

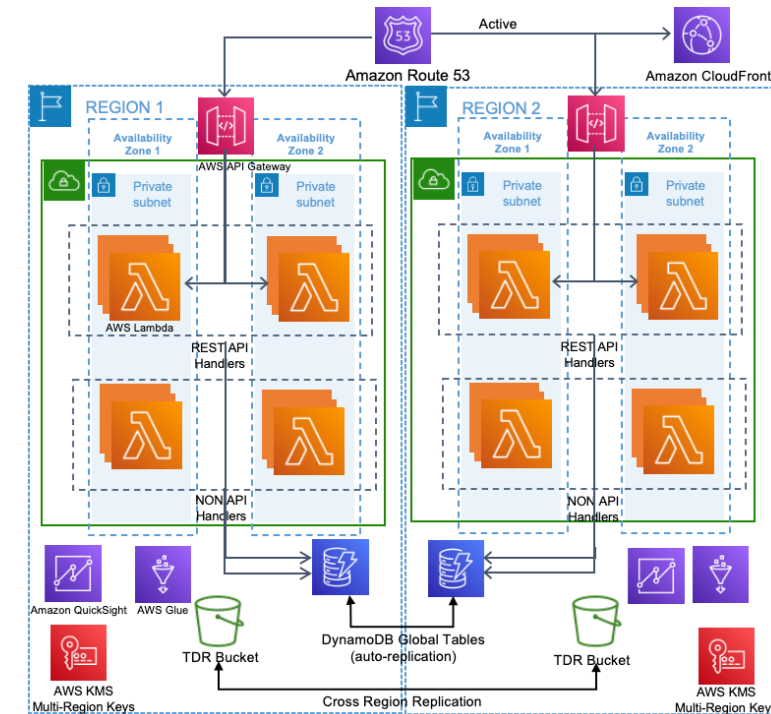
Use Cases:

- **Regulatory Compliance:** Organizations subject to regulations like GDPR, HIPAA, or CCPA can use Macie to ensure they're not inadvertently storing sensitive information where they shouldn't be.
- **Data Security:** Companies that handle sensitive customer data (like financial institutions, healthcare providers, or e-commerce platforms) can use Macie to ensure their data storage practices are secure.
- **Threat Detection:** By monitoring data access patterns, Macie can help organizations identify potential data breaches or unauthorized access.

AWS KEY MANAGEMENT SERVICE (KMS)

Key Features:

- **Centralized Key Management:**
 - Allows creation, management, and rotation of symmetric encryption keys, which are used in cryptographic functions to protect data.
 - Keeps track of all keys and their metadata centrally.
- **Integrated with AWS Services:**
 - Easily integrated with other AWS services like Amazon S3, RDS, Redshift, and others to encrypt data stored within these services.
- **Custom Key Store:**
 - You can use KMS custom key store features to gain more control over the cryptographic keys by using CloudHSM clusters.
- **Audit and Compliance:**
 - With AWS CloudTrail, you can audit key usage to ensure compliance with your internal policies and regulatory standards.
- **Access Control:**
 - Uses AWS Identity and Access Management (IAM) and KMS key policies to control access to the encryption keys.
 - Allows you to define who can use which key, for what, and when.
- **Automatic Key Rotation:**
 - Supports the automatic rotation of keys to enhance security, meaning a new key version is created every year.
- **Envelope Encryption:**
 - KMS uses envelope encryption, where data keys are used to encrypt data, and then the data key itself is encrypted using the master key.
- **Support for Hybrid Architectures:**
 - While KMS is a cloud-based service, it also supports scenarios where you need to encrypt data in your on-premises data centers.



ORCHESTRATION & WORKFLOW

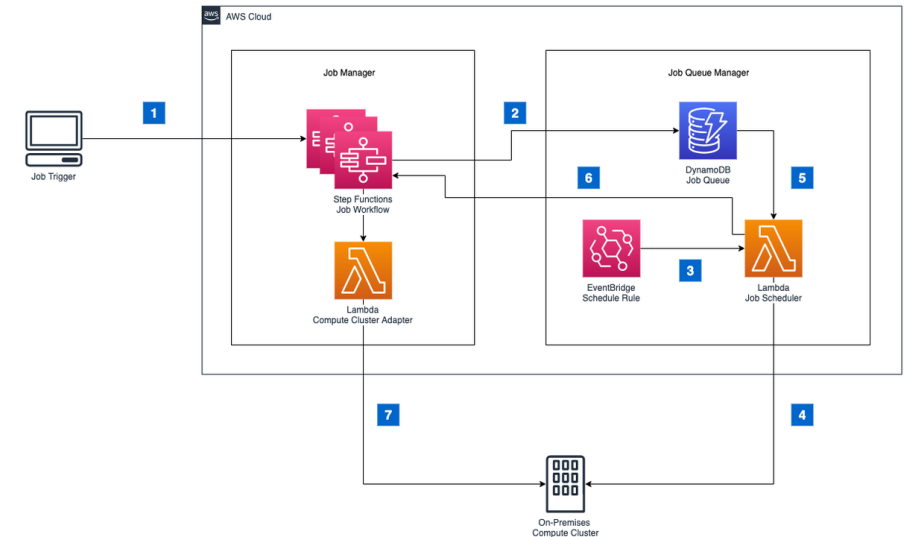
- **AWS Step Functions:** This lets you coordinate multiple AWS services into serverless workflows so you can build and update apps quickly.
- **Amazon CloudWatch:** Monitors your AWS resources and applications in real-time.

AWS STEP FUNCTIONS

- AWS Step Functions is a serverless workflow service that lets you coordinate distributed components and microservices using visual workflows. In the context of big data, Step Functions can be invaluable in orchestrating multi-step processing jobs, ensuring that each component or service works in harmony and in the right sequence.

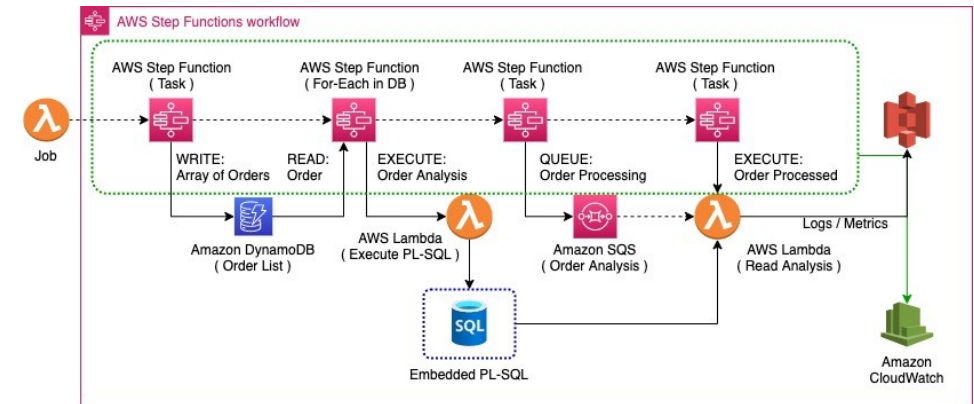
Key Features:

- **Visual Workflows:** Define and visualize the steps of your data processing or analytics tasks in a clear diagram.
- **Error Handling:** You can set retry policies for failed steps, catch specific errors, and direct the workflow based on those errors.
- **Serverless:** No infrastructure to provision or manage. You build and update applications without thinking about servers.
- **Integration:** Seamlessly integrates with various AWS services like AWS Lambda, AWS Glue, Amazon SageMaker, and more.
- **State Management:** Maintains the state of your application, even through failures or restarts, and can exist for up to a year.
- **Parallel Execution:** Can execute multiple branches of a workflow in parallel, enhancing the efficiency of your pipeline.



BIG DATA PIPELINE WITH AWS STEP FUNCTIONS

- **Data Collection:** Start with triggering workflows based on data landing in Amazon S3 or data streamed via Amazon Kinesis.
- **Transformation and Preparation:**
 - **AWS Glue Jobs:** Use Step Functions to kick off Glue ETL jobs to transform and prepare data.
 - **Lambda Functions:** Invoke AWS Lambda to execute specific transformations or cleansing routines.
- **Data Analysis and Machine Learning:**
 - **Amazon SageMaker:** Step Functions can orchestrate machine learning workflows, from data preprocessing, model training, to deployment with Amazon SageMaker.
 - **AWS Batch:** Execute batch computing workloads as a step in your workflow.
- **Notifications and Reporting:**
 - Use Amazon SNS or Lambda to notify users on the status of the workflow.
 - Update Amazon Quicksight dashboards or store results in Amazon RDS or Redshift.
- **Cleanup and Archival:**
 - Trigger Lambda functions to clean up temporary storage or move older data to Amazon Glacier for archival.



BIG DATA PIPELINE WITH AWS STEP FUNCTIONS

Use Cases:

- **ETL Workflows:** Orchestrate ETL tasks that involve multiple AWS services, transforming raw data into insights.
- **Machine Learning Pipelines:** Coordinate steps from data preprocessing, model training, validation, and deployment.
- **Data Lake Workflows:** Manage complex data lake operations, including data ingestion, processing, and cataloging.
- **Batch Processing:** Coordinate and manage batch processing tasks, ensuring they execute in the right order and handle failures gracefully.

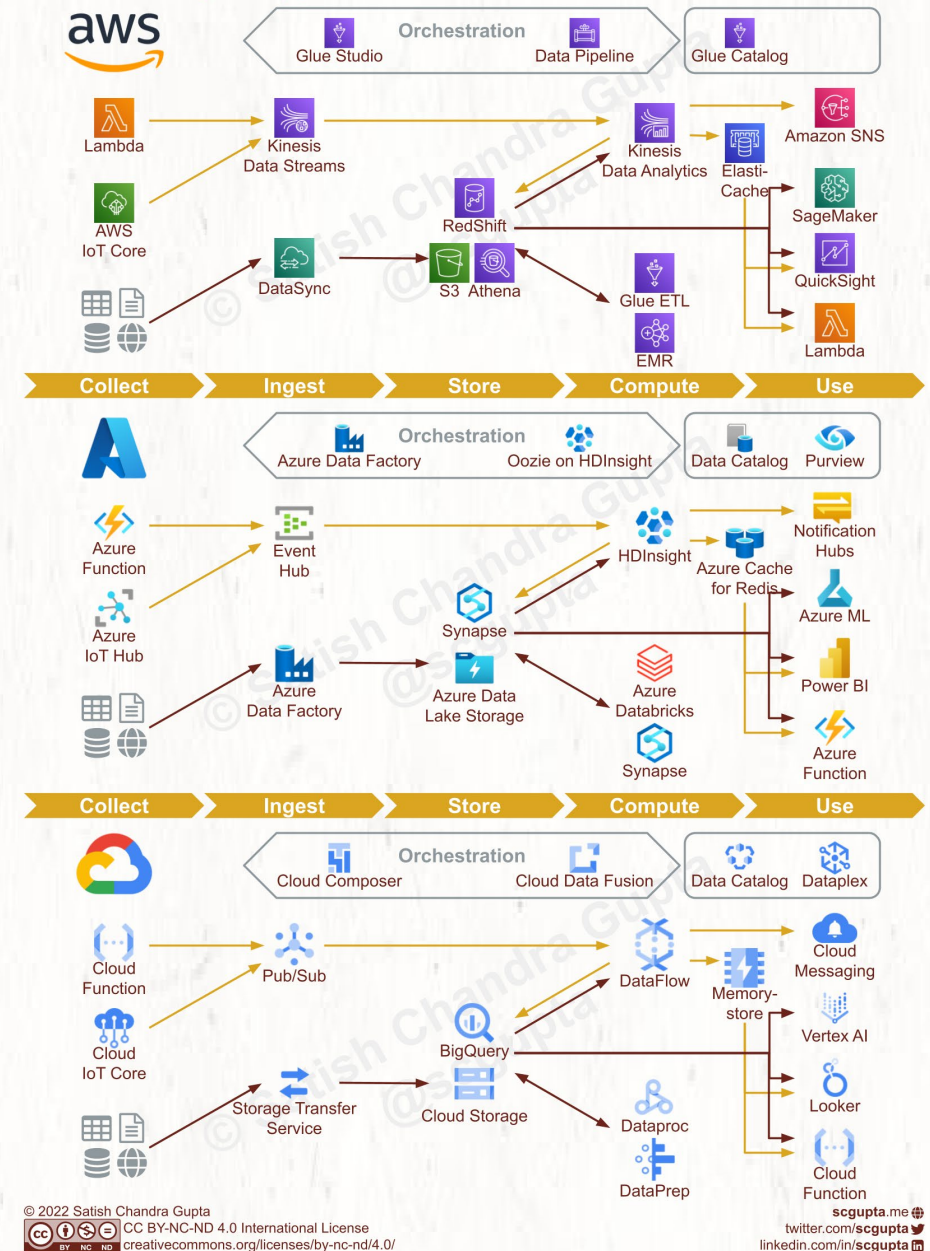
Considerations:

- **Execution Limit:** As of the last update, a single execution of a Step Function can run for up to a year.
- **Scalability:** AWS Step Functions is designed to be scalable and can handle a high number of executions, but always monitor to ensure it meets your specific needs.
- **Cost:** Costs are based on the number of state transitions (steps) and the duration of workflow executions. Ensure your workflows are efficient to manage costs.

AZURE VS GCP VS AWS

Big Data Pipelines on AWS, Azure, and Google Cloud

ml4devs.com/big-data-pipeline



REFERENCES

- <https://www.ml4devs.com/articles/scalable-efficient-big-data-analytics-machine-learning-pipeline-architecture-on-cloud/>
- <https://sapphireventures.com/blog/what-is-the-open-data-ecosystem-and-why-its-here-to-stay/>
- <https://docs.lib.purdue.edu/cctech/1/>
- <https://www.mdpi.com/2504-2289/4/3/17>
- <https://towardsdatascience.com/building-an-end-to-end-open-source-modern-data-platform-c906be2f31bd>
- <https://medium.com/@Cartelis/setting-up-an-open-source-modern-data-stack-6dd41094cd2f>
- <https://www.datafold.com/blog/the-modern-data-stack-open-source-edition>
- <https://aws.amazon.com/blogs/architecture/how-ribbon-built-a-scalable-resilient-robocall-mitigation-platform/>