
PLURALSIGHT

Data Engineering

THE HARTFORD

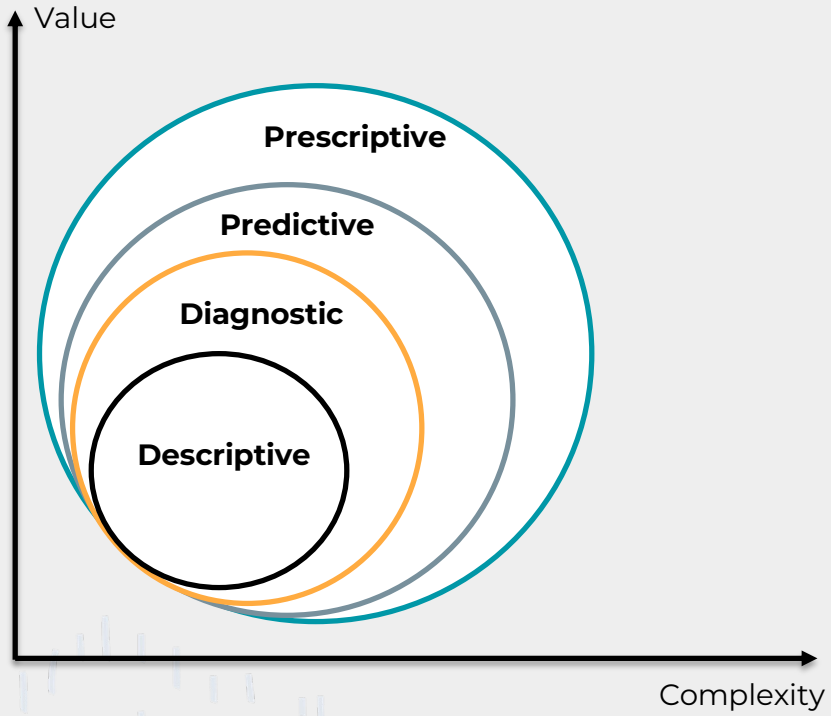
Tech Catalyst Bootcamp

Tarek Atwan
Instructor, Pluralsight

PLURALSIGHT

What is the end goal?

Types of Data Analytics



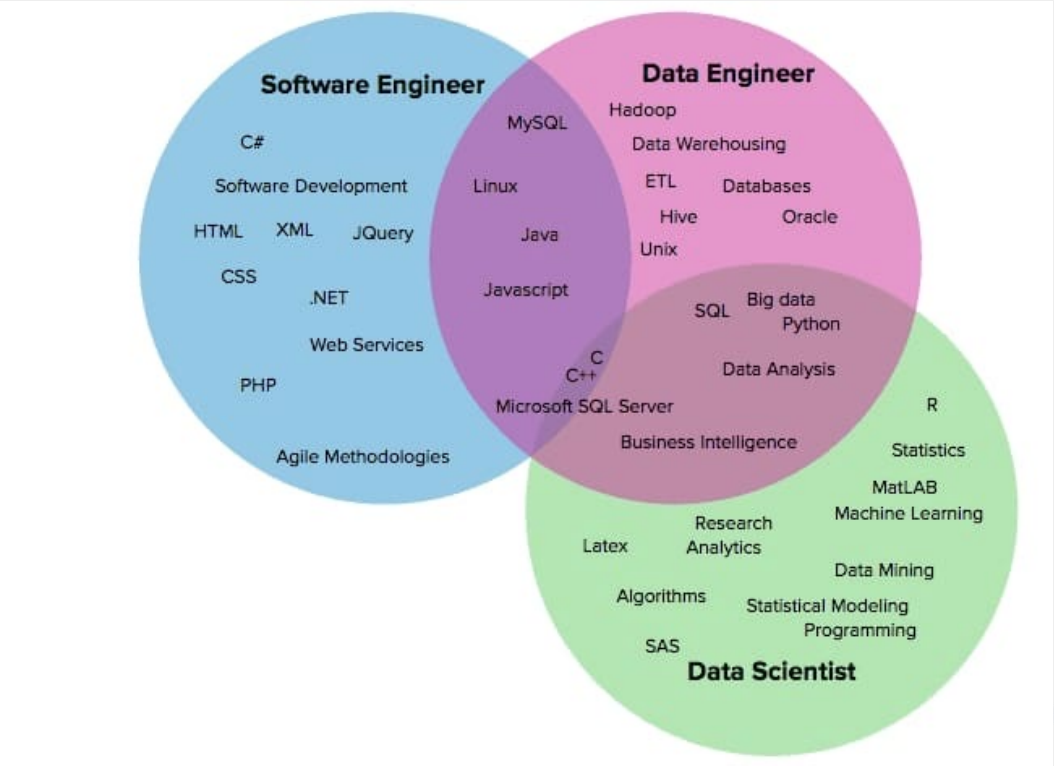
Descriptive
What is happening in my business?

Diagnostic
Why is it happening?

Predictive
What is likely to happen?

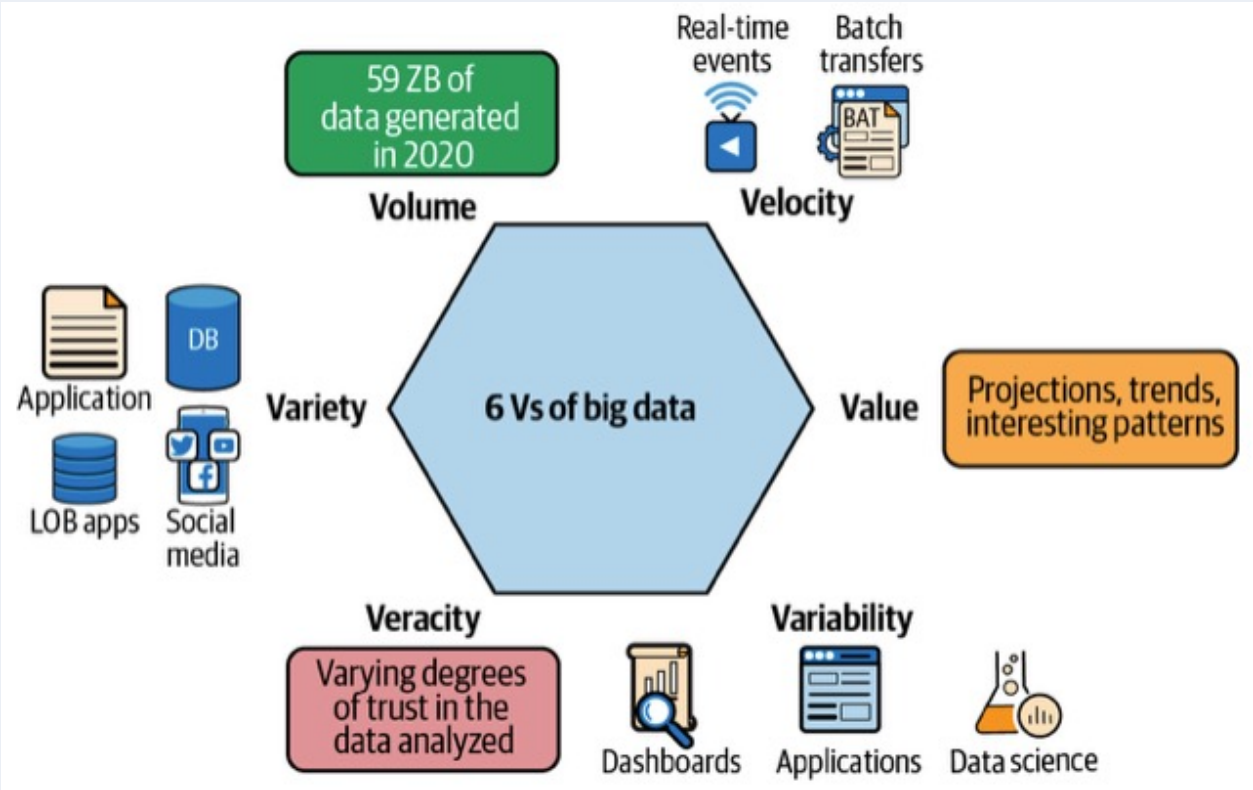
Prescriptive
What do I need to do?

Data Engineer Skills



Data Engineering Concepts Review

What is Big Data



Types of Data

Unstructured data

The university has 5600 students.
John's ID is number 1, he is 18 years old and already holds a B.Sc. degree.
David's ID is number 2, he is 31 years old and holds a Ph.D. degree. Robert's ID is number 3, he is 51 years old and also holds the same degree as David, a Ph.D. degree.

Semi-structured data

```
<University>
  <Student ID="1">
    <Name>John</Name>
    <Age>18</Age>
    <Degree>B.Sc.</Degree>
  </Student>
  <Student ID="2">
    <Name>David</Name>
    <Age>31</Age>
    <Degree>Ph.D. </Degree>
  </Student>
  ....
</University>
```

Structured data

ID	Name	Age	Degree
1	John	18	B.Sc.
2	David	31	Ph.D.
3	Robert	51	Ph.D.
4	Rick	26	M.Sc.
5	Michael	19	B.Sc.

Unstructured	Semi-structured	Structured
PDFs, JPEGs, MP3, Movies, ...	CSV, JSON, XML, MongoDB, ...	Oracle, MSSQL, MySQL, DB2, ...

Types of Data

Structured Data

Data that is **organized** in a define manner or **schema**, typically found in **relational databases.**

- Easily queryable
- Organized in rows and columns
- Has a consistent structure

Unstructured Data

Data that doesn't have a predefined structure or schema.

- Not easily queryable without preprocessing
- May come in various formats

Semi-Structured Data

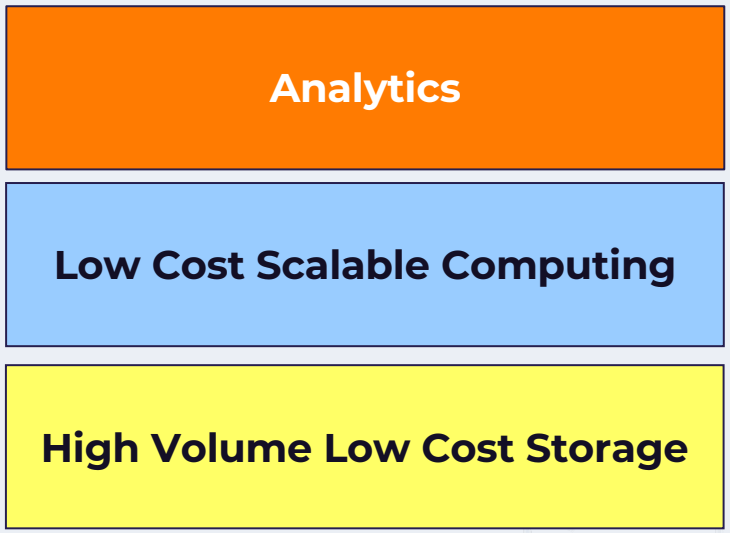
Data that is not as organized as structured data but has some level of structure in the forms of tags, hierarchies, or other patterns.

- Elements might be tagged or categorized in some way
- More flexible than structured data but not as chaotic as unstructured data.

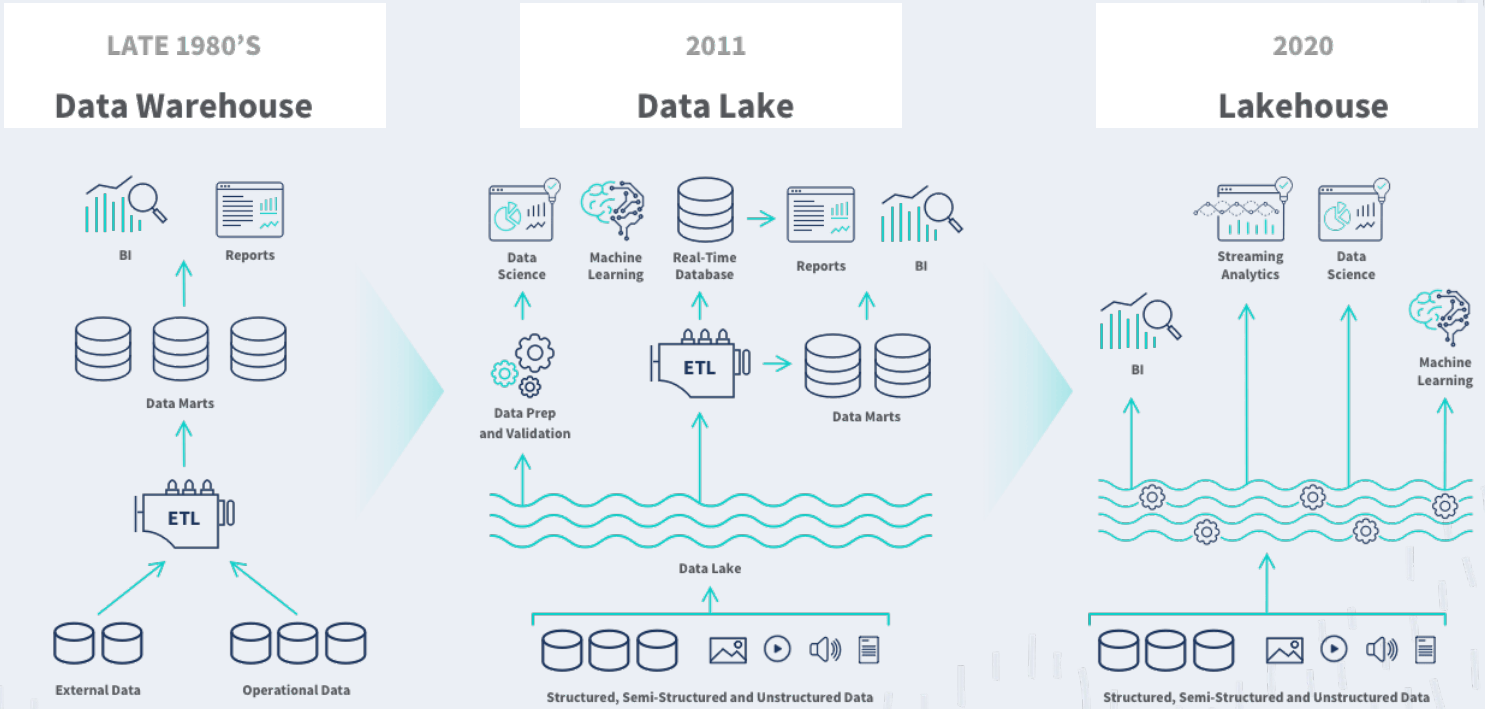
Big Data Stack

A Big Data stack typically has three layers

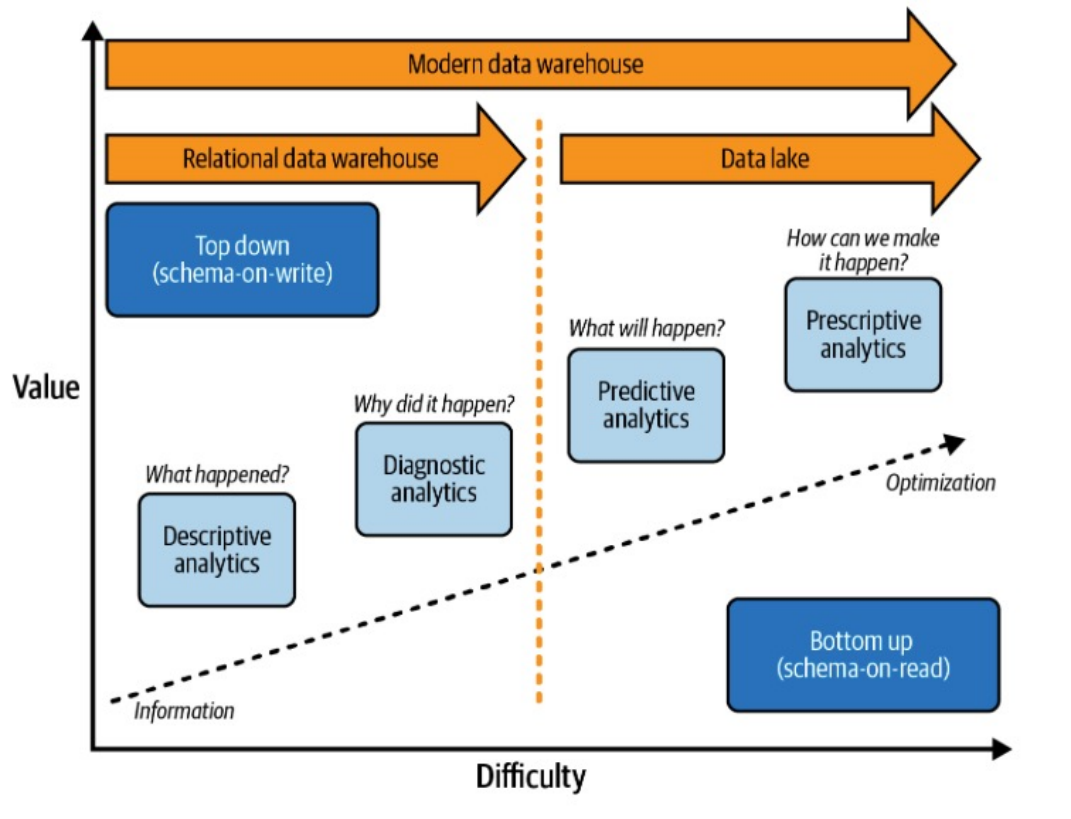
The technology and tools at each layer vary widely dependent on business needs



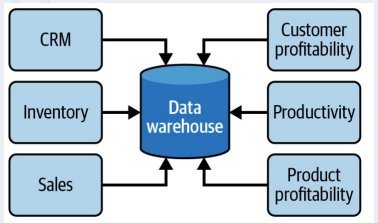
Modern Data Architecture

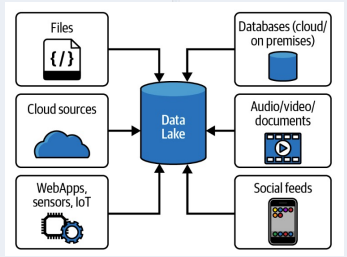


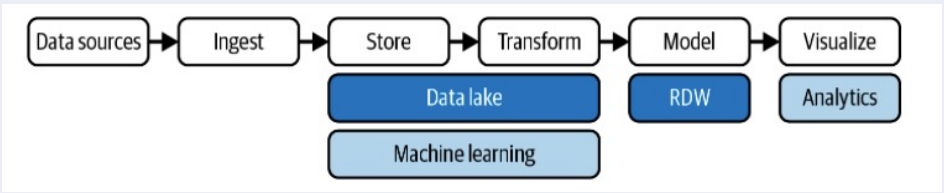
Big Data Analytics

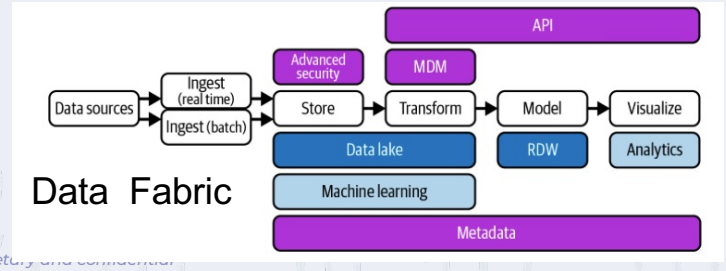


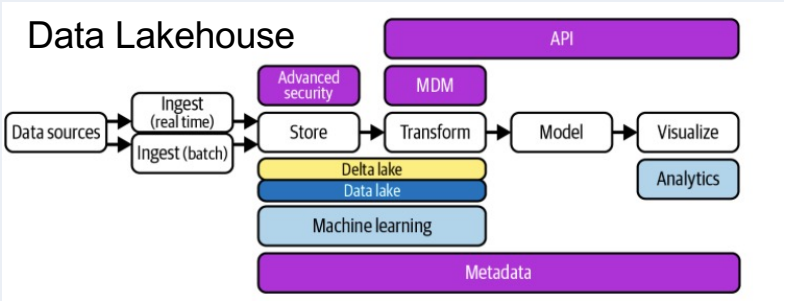
Evolution Summary











Data Sources

- **JDBC**
  - Java Database Connectivity
  - Platform-independent
  - Language-dependent
- **ODBC**
  - Open Database Connectivity
  - Platform-dependent (using drivers)
  - Language-independent
- **Raw Logs**
- **API's**
- **Streams**

- **CSV**
- **XML**
- **JSON**
- **Avro**
- **Parquet**
- **ORC**

Avro, ORC, and Parquet

Feature	Avro	Parquet	ORC
Data Format	Row-oriented	Columnar	Columnar
Schema Evolution	Excellent	Good	Good
Compression	Snappy, Deflate, Bzip2	Snappy, Gzip, LZO	Zlib, Snappy
Read Performance	Good for all rows	Excellent for reading specific columns	Excellent for reading specific columns
Write Performance	Excellent	Good	Good
Support for Complex Data Types	Yes	Yes	Yes
Use Cases	Data serialization, streaming data	Analytical queries, BI tools	Analytical queries, BI tools
Schema Storage	Included in file	Separate schema file	Separate schema file
Interoperability	Good (used in Hadoop ecosystem, Kafka)	Good (used in Hadoop ecosystem, Presto, Spark)	Good (used in Hadoop ecosystem, Hive, Spark)
Tooling and Libraries	Avro libraries available for most languages	Supported by major big data tools	Supported by major big data tools
File Size	Larger due to row-oriented storage	Smaller due to columnar compression	Smaller due to columnar compression
Suitability for Small Files	Good	Not optimal	Not optimal
Complex Queries	Moderate	Excellent	Excellent
Splitable	Yes	Yes	Yes

Data Engineering Process

What is ETL

ETL Pipelines

ETL stands for **Extract**, **Transform**, and **Load**.

It is a process used to move data from source systems into a data warehouse.

ETL Pipelines - Extract

Extract

- Retrieve raw data from source systems, which can be databases, CRMs, flat files, APIs, or other data repositories
- Ensure data integrity during the extraction phase
- Can be done in real-time or in batches, depending on requirements

ETL Pipelines - Transform

Transform

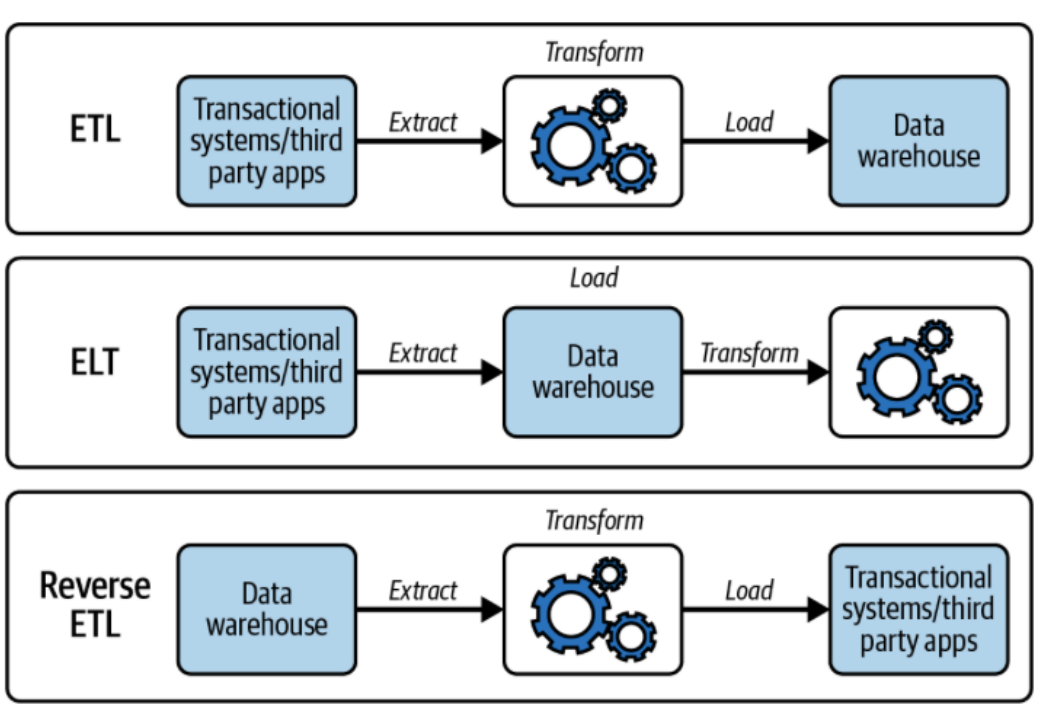
- Convert extracted data into a format suitable for the target data warehouse
- Can involve various operations such as:
  - Data cleansing (e.g. removing duplicates, fixing errors)
  - Data enrichment (e.g. adding additional data from other sources)
  - Format changes (e.g. data formatting, string manipulation)
  - Aggregates or computations (e.g. calculating totals or averages)
  - Encoding or decoding data
  - Handline missing values

ETL Pipelines - Load

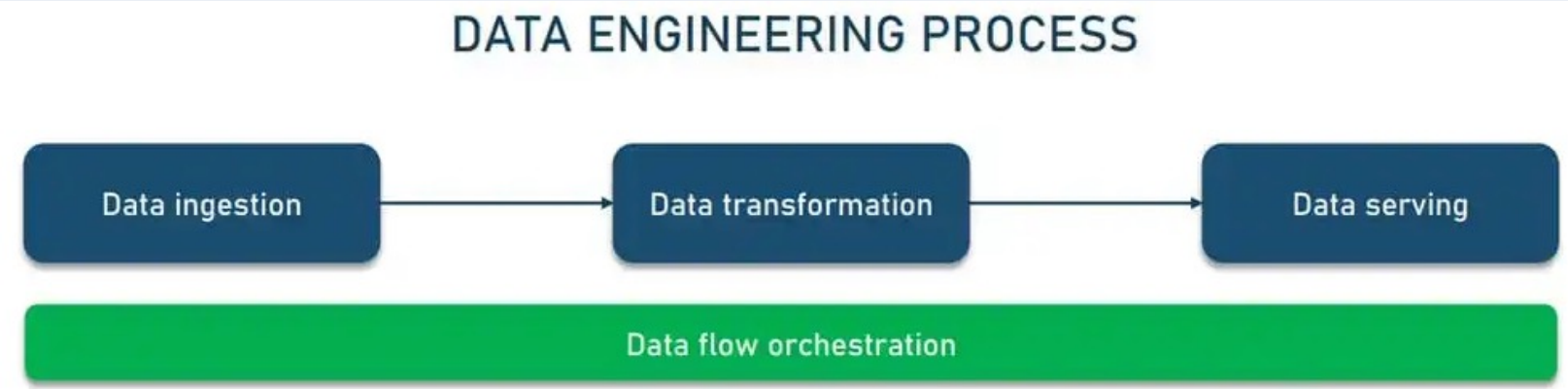
Load

- Move the transformed data into the target data warehouse or another data repository
- Can be done in batches (all at once) or in a streaming manner (as data becomes available)
- Ensure that data maintains its integrity during the loading phase

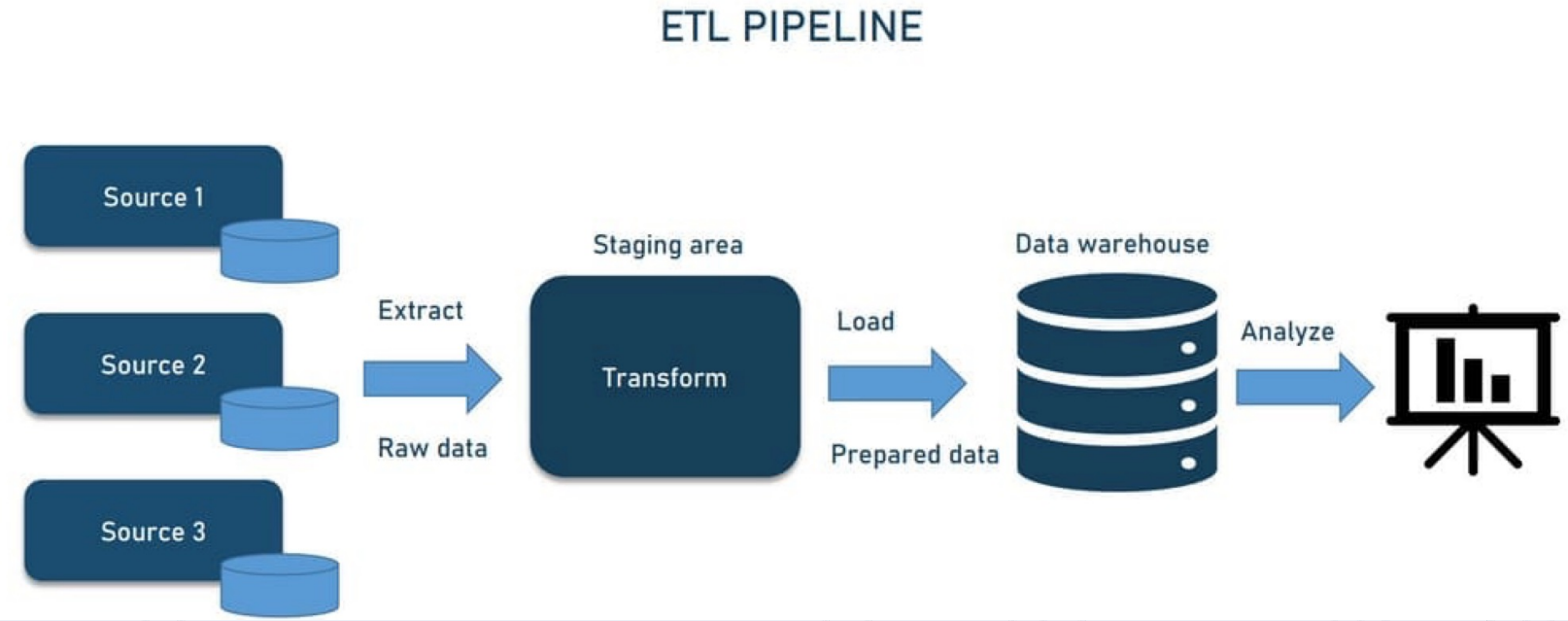
ETL, ELT, and Reverse ETL



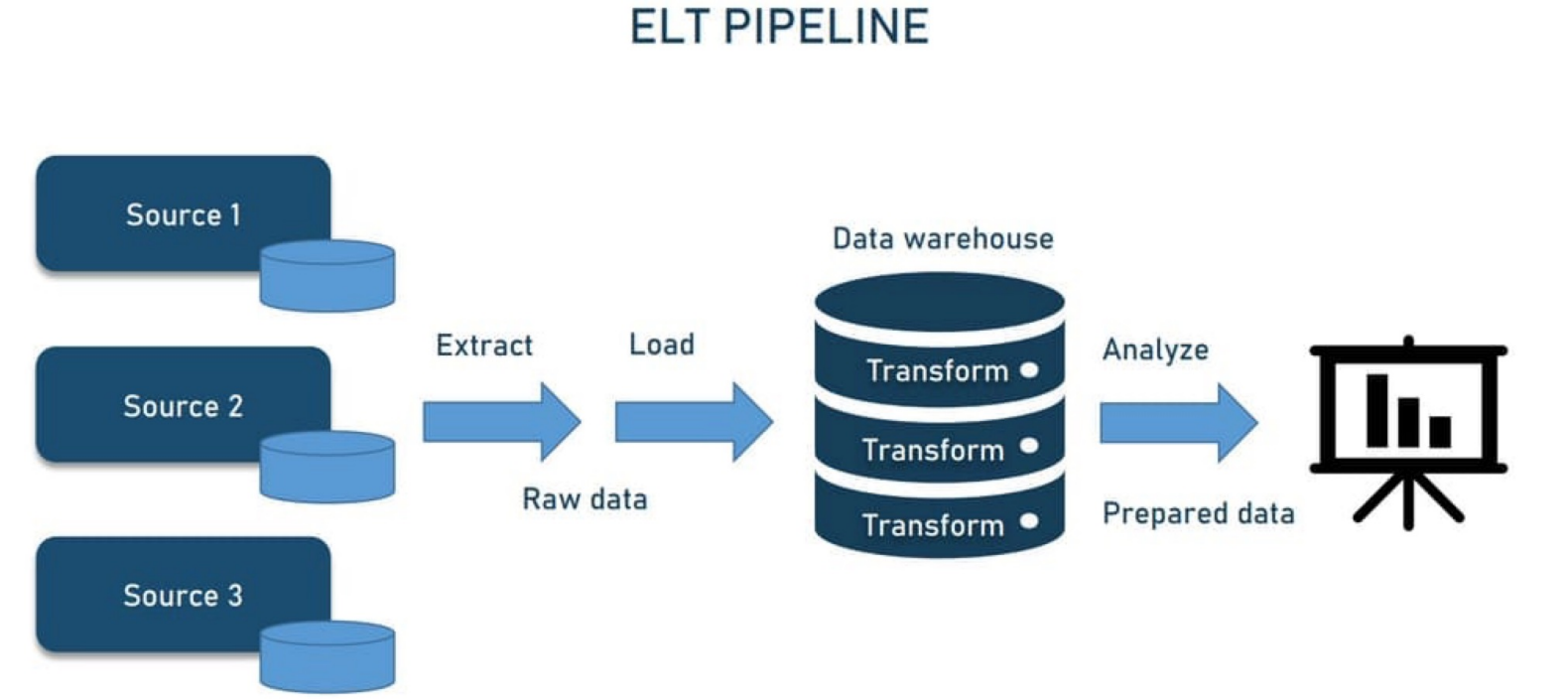
Data Engineering



Data Engineering

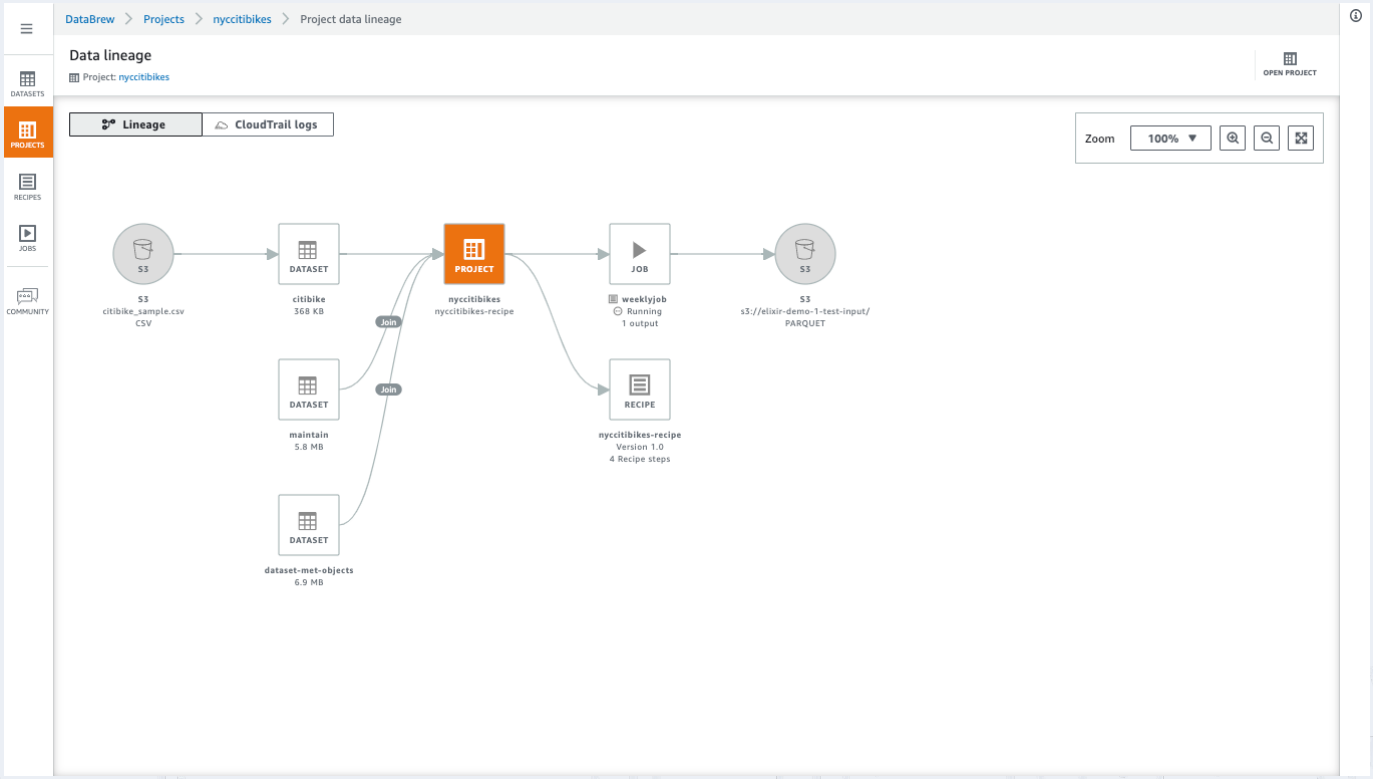


Data Engineering

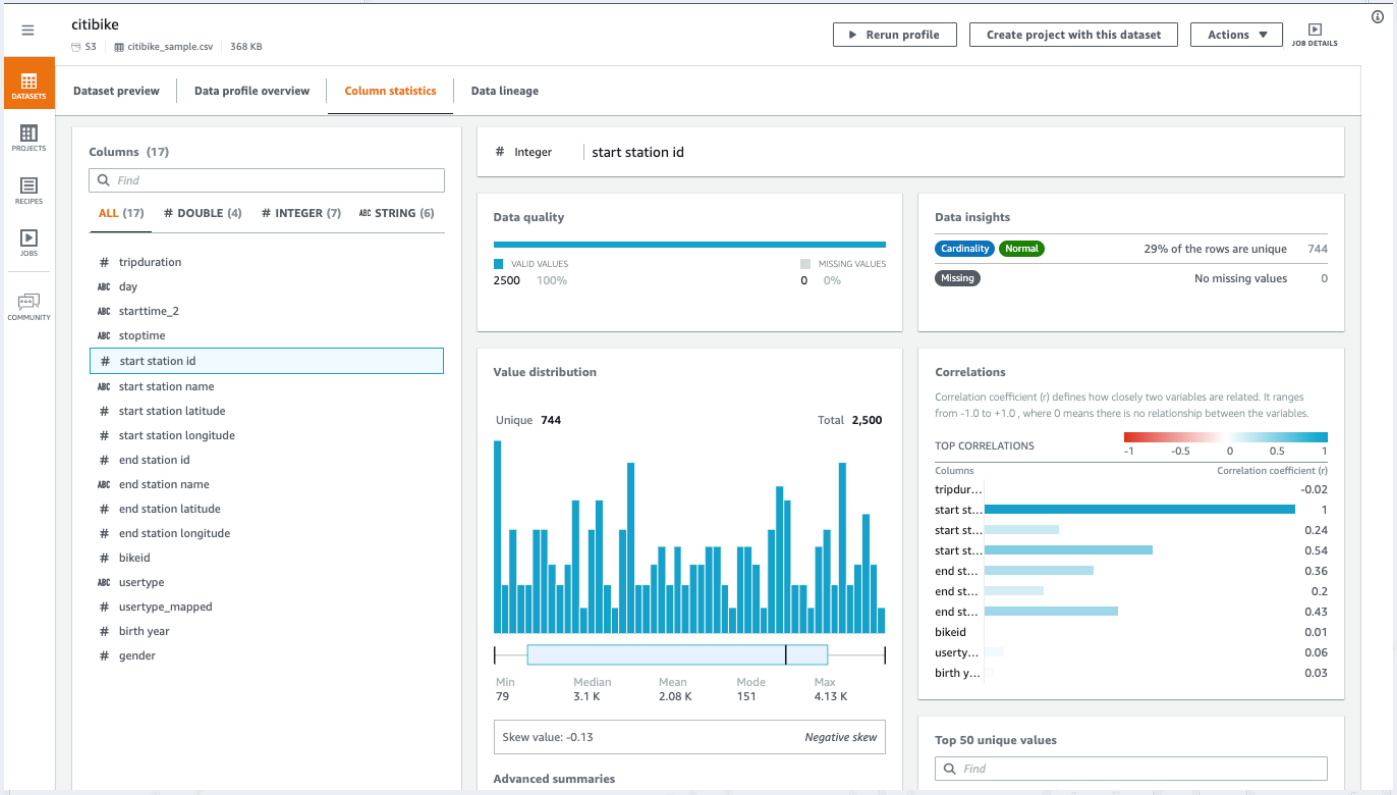


AWS Glue

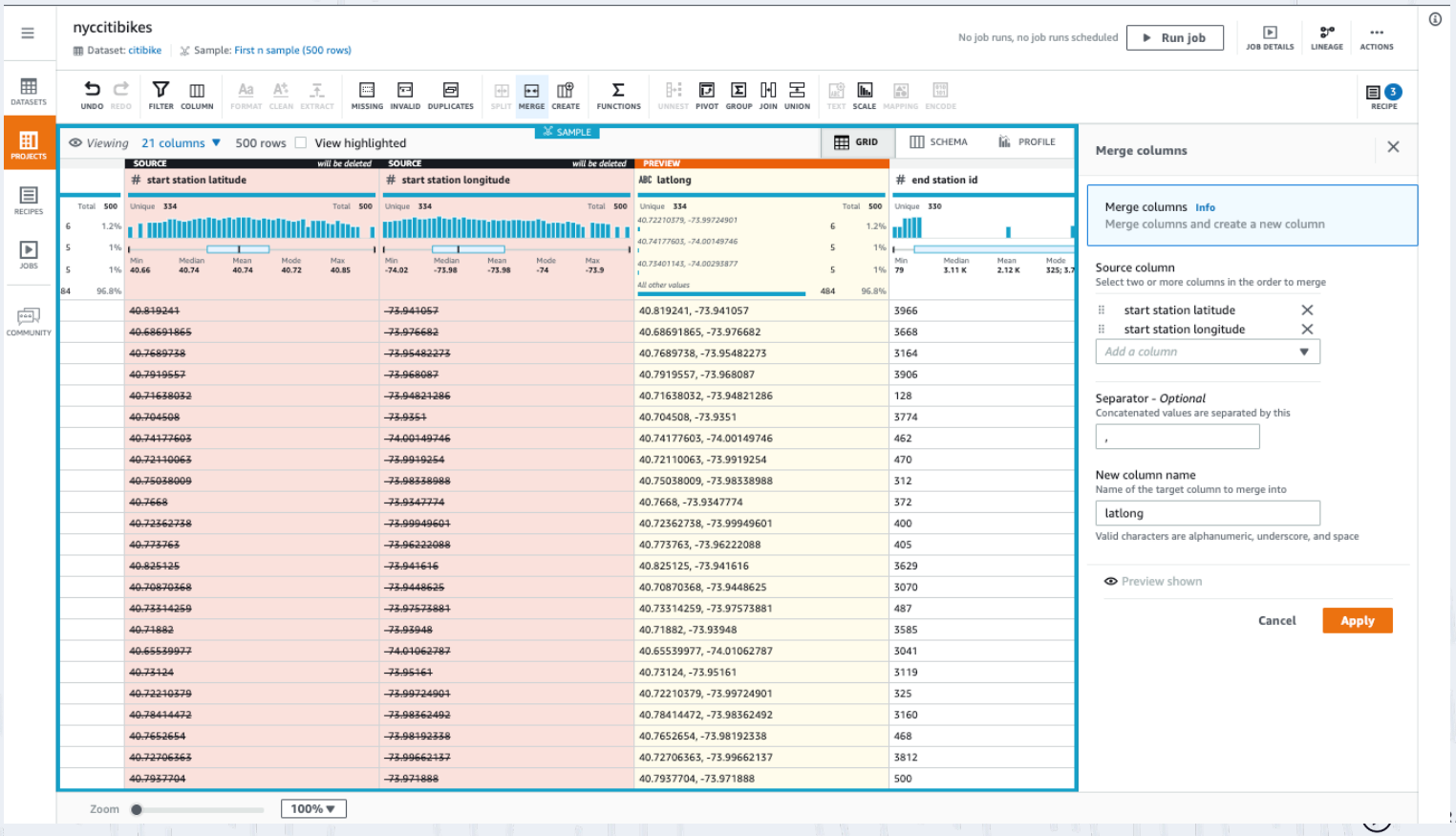
AWS Glue DataBrew



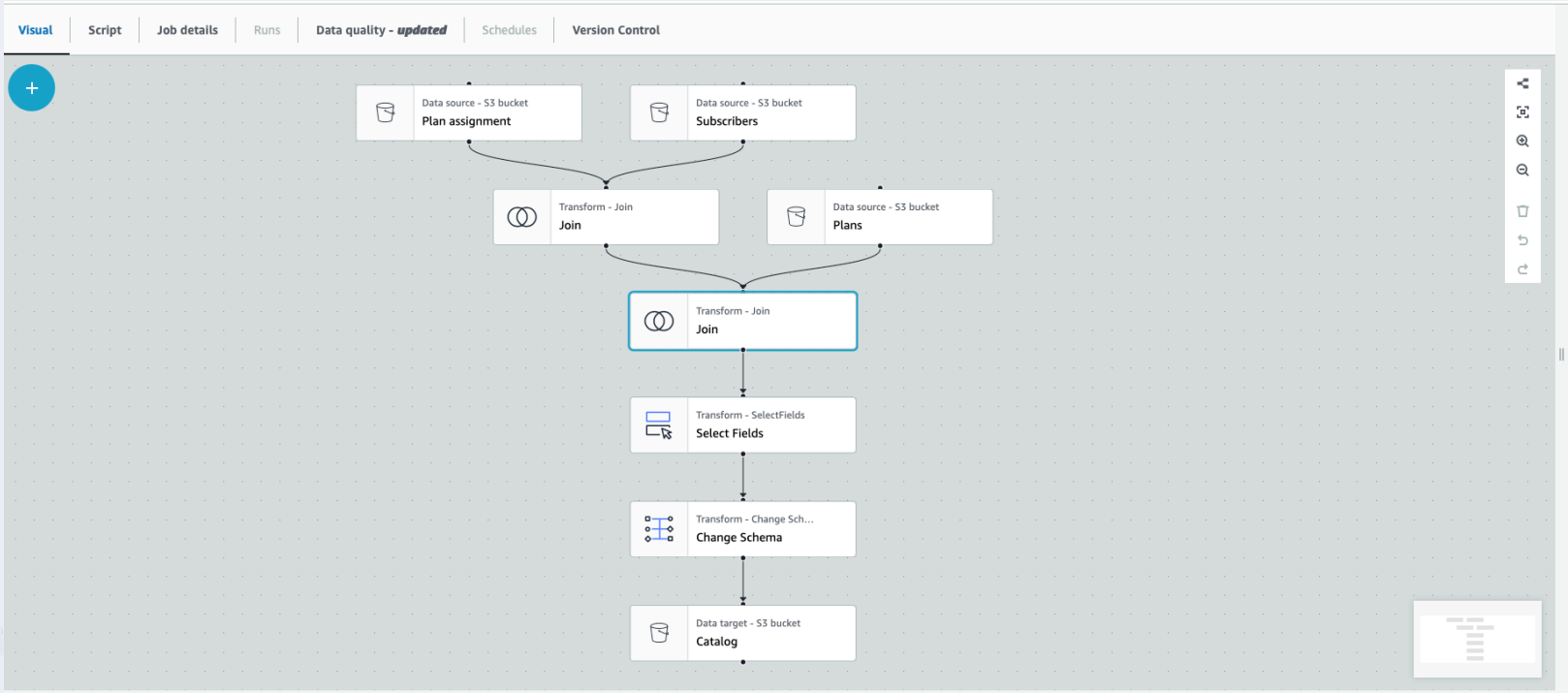
AWS Glue DataBrew



AWS Glue DataBrew



AWS Glue Studio (Visual ETL)



What is Orchestration

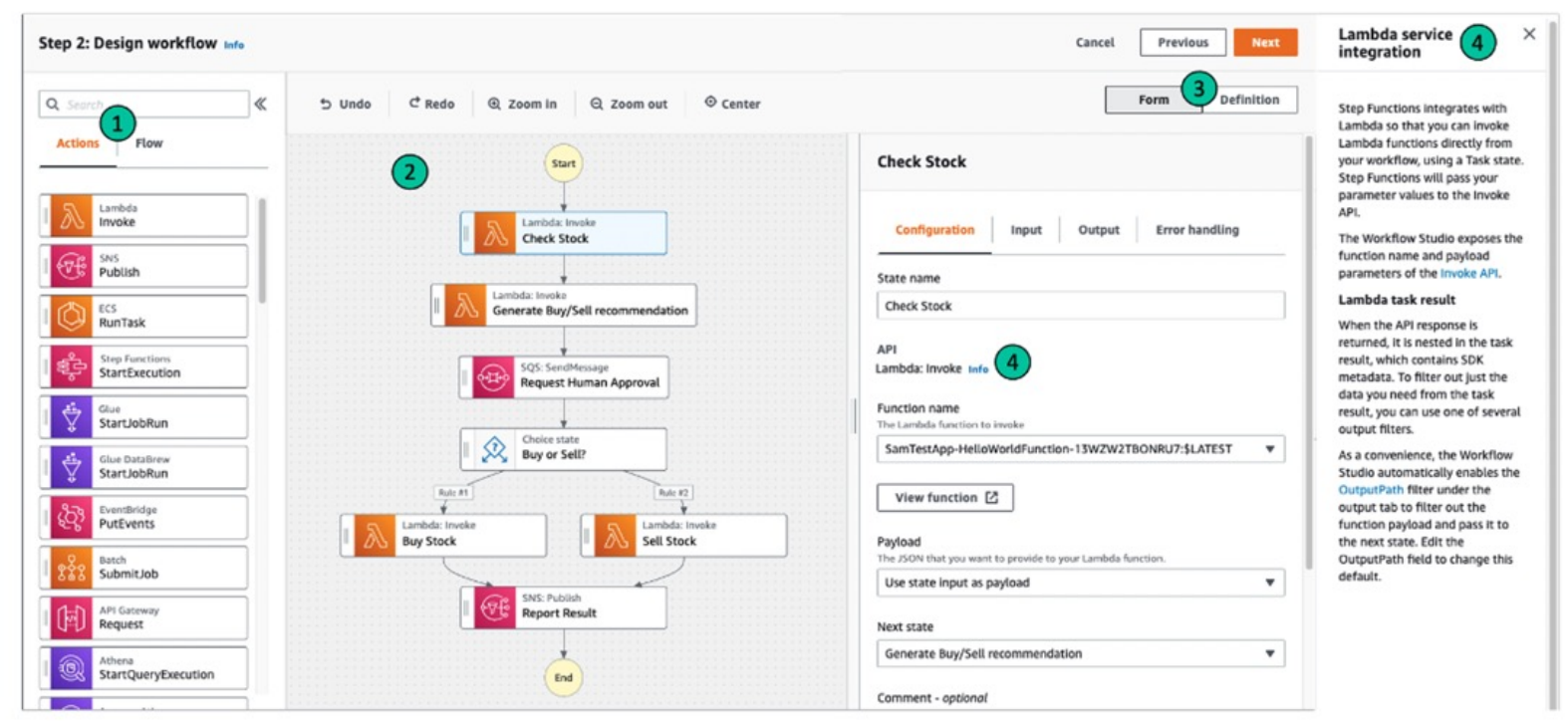
Orchestration

- A large part of data engineering is extracting, transforming, and loading data between sources.
- Orchestration is a process of dependency management, facilitated through automation.
- The data orchestrator manages scheduling, triggering, monitoring, and even resource allocation.

Orchestration

- Workflow management
- Automation
- Error handling
- Recovery
- Monitoring, alerting
- Resource optimization
- Observability
- Debugging
- Compliance/Auditing

Amazon Step Function



Apache Airflow

SQL Databases

Database Ranking

https://db-engines.com/en/ranking

- Complete ranking
- Relational DBMS
- Key-value stores
- Document stores
- Time Series DBMS
- Graph DBMS
- Search engines
- RDF stores
- Object oriented DBMS
- Vector DBMS
- Wide column stores
- Multivalue DBMS
- Spatial DBMS
- Native XML DBMS
- Event Stores
- Content stores
- Navigational DBMS

421 systems in ranking, June 2024

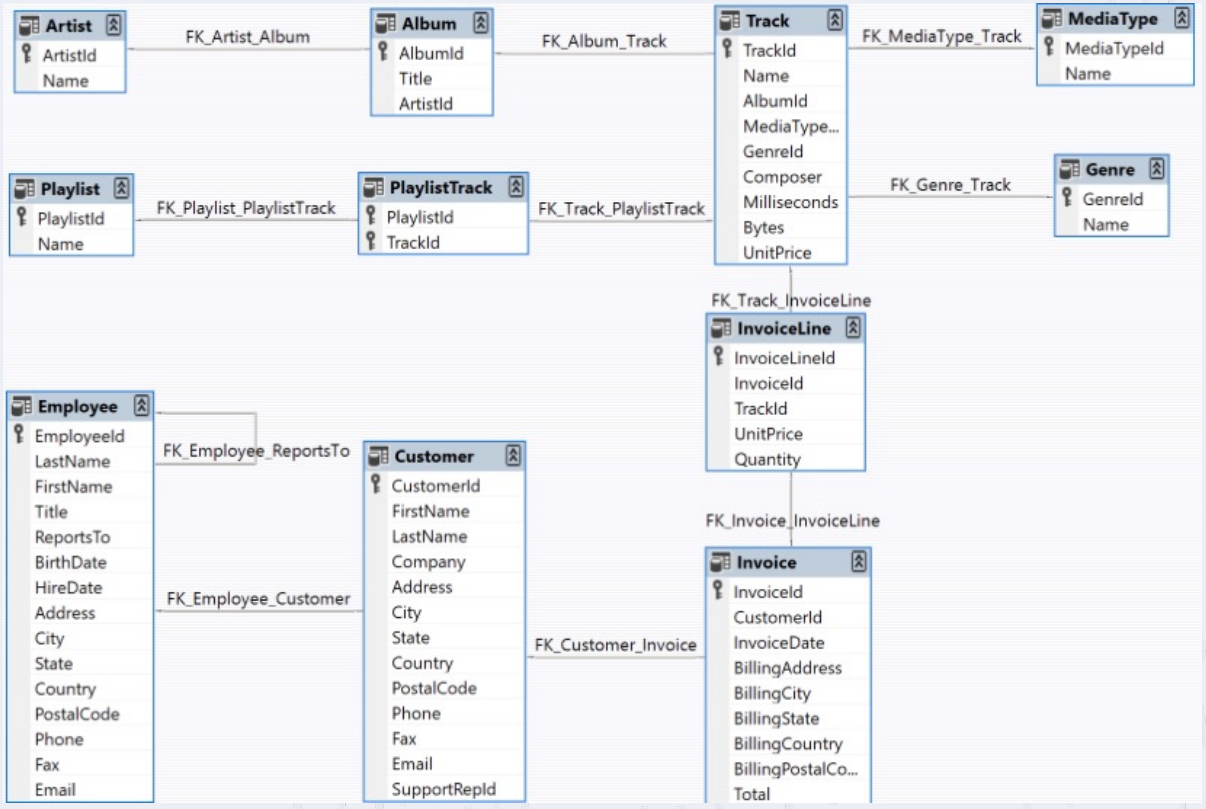
Rank			DBMS	Database Model	Score		
Jun 2024	May 2024	Jun 2023			Jun 2024	May 2024	Jun 2023
1.	1.	1.	Oracle	Relational, Multi-model	1244.08	+7.79	+12.61
2.	2.	2.	MySQL	Relational, Multi-model	1061.34	-22.39	-102.59
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	821.56	-2.73	-108.50
4.	4.	4.	PostgreSQL	Relational, Multi-model	636.25	-9.30	+23.43
5.	5.	5.	MongoDB	Document, Multi-model	421.08	-0.58	-4.29
6.	6.	6.	Redis	Key-value, Multi-model	155.94	-1.86	-11.41
7.	7.	↑ 8.	Elasticsearch	Search engine, Multi-model	132.83	-2.52	-10.92
8.	↑ 9.	↑ 11.	Snowflake	Relational	130.36	+9.03	+16.23
9.	↓ 8.	↓ 7.	IBM Db2	Relational, Multi-model	125.90	-2.56	-18.99
10.	10.	10.	SQLite	Relational	111.41	-2.91	-19.81

Relational Databases

- A relational database management system (RDBMS) is a software layer of tools and services that manages relational tables. In practice, the terms RDBMS and relational database are considered to be synonyms. A relational database provides a consistent interface between applications, users, and relational database. Organizations use RDBMS for managing large amounts of business critical information from various departments.
- Multiple users can work with the same database in different ways. For example, they can perform database operations and aggregate key data points without creating data redundancy. Relational database management systems also give your database administrator greater access control over your data.

Relational Databases - ERDs

Entities, their data types, and relationships are all illustrated in the diagram.



How Relational Databases Work

Data Model

- **Tables:** Represent real-world objects or concepts (entities).
- **Attributes:** Columns in a table holding specific kinds of data.
- **Fields:** Store the actual values of attributes.
- **Primary Key:** Unique identifier for each row in a table.
- **Foreign Key:** References the primary key of another table, creating logical connections between tables.
- **Example:** An orders table can have a foreign key (customer ID) linking to the customer table.

How Relational Databases Work

SQL (Structured Query Language)

- **Primary Interface:** Used to communicate with relational databases.
- **ANSI Standard:** Became a standard in 1986, supported by all popular relational database engines.
- **Functions:** Used to update, delete, store, retrieve data, and manage the database.
- **Ease of Use:** Uses common English keywords and integrates well with programming languages like Java.

How Relational Databases Work

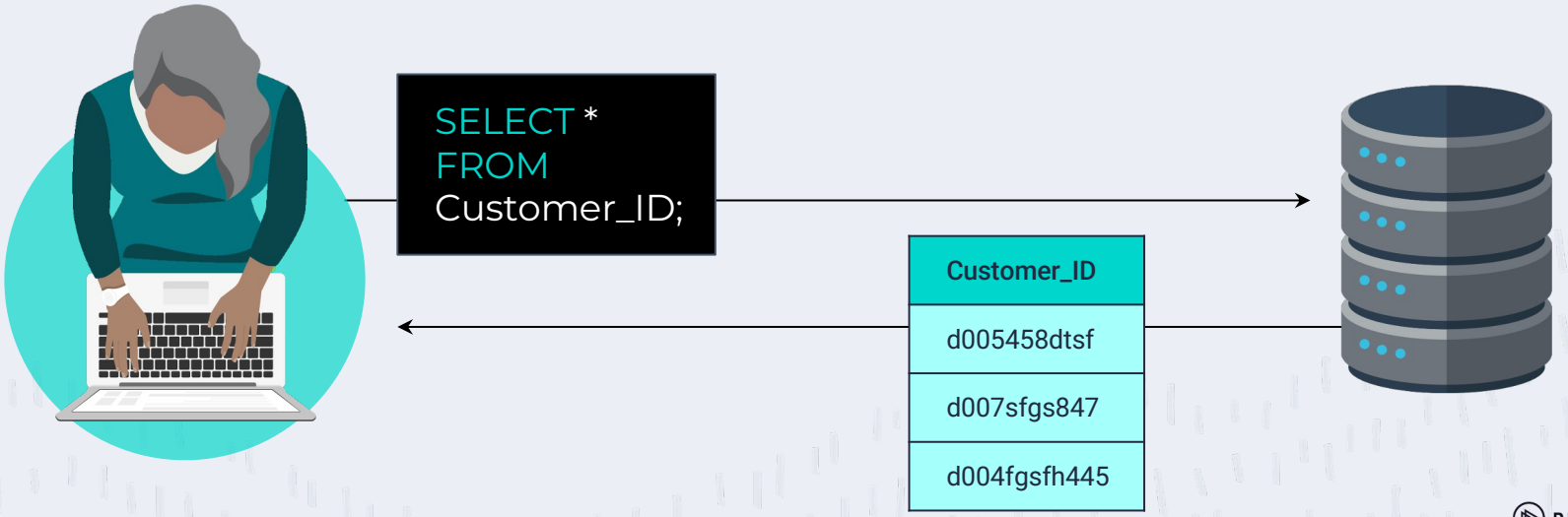
Transactions

- **Definition:** One or more SQL statements forming a single logical unit of work.
- **All-or-Nothing:** Entire transaction must complete, or none of the components go through.
- **Outcome:** Results in a COMMIT (successful) or a ROLLBACK (unsuccessful).
- **Properties:** Treated coherently, reliably, independently, and isolated from other transactions.

SQL

SQL (often pronounced "sequel") stands for Structured Query Language.

It is a powerful tool that enables programmers to create, populate, manipulate, and access databases. It also provides an easy method for dealing with server-side storage.



SQL

Data using SQL is stored in tables on the server, much like spreadsheets you would create in Microsoft Excel.

This makes the data easy to visualize and search.

Customer_ID	Date_ID
d005458dtsf	6/26/2019
d007sfgs847	8/3/2018
d004fgsfh445	12/3/2018

Order_ID	Customer_ID	Date_ID
10001	d005458dtsf	6/26/2019
10002	d007sfgs847	8/3/2018
10003	d004fgsfh445	12/3/2018

CRUD Operations

Create **R**ead **U**pdate **D**elete is a set of operations used with persistent storage.

Create	INSERT INTO table (column1, column2, column3)
Read	SELECT * FROM table
Update	UPDATE table SET column1 = VALUE WHERE id = 1
Delete	DELETE FROM table WHERE id = 5

These tools are fundamental to all programming languages, not just SQL.

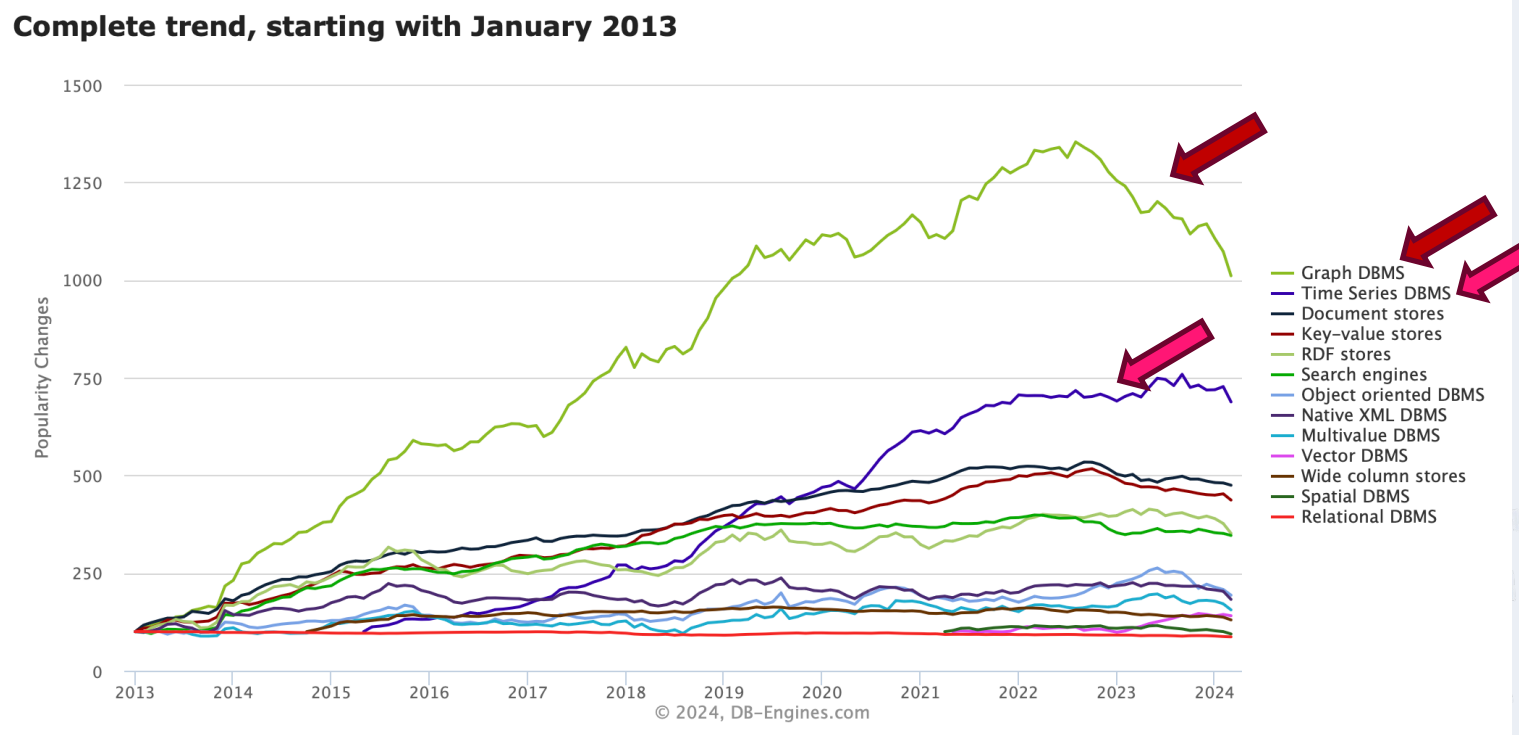
SQL JOINS

`INNER JOIN`	Returns records that have matching values in both tables.
`LEFT JOIN`	Returns all records from the left table and the matched records from the right table.
`RIGHT JOIN`	Returns all records from the right table and the matched records from the left table.
`CROSS JOIN`	Returns records that match every row of the left table with every row of the right table. This type of join has the potential to make very large tables.
`FULL OUTER JOIN`	Places null values within the columns that do not match between the two tables, after an inner join is performed.

NoSQL Databases

Database Trends

https://db-engines.com/en/ranking_categories

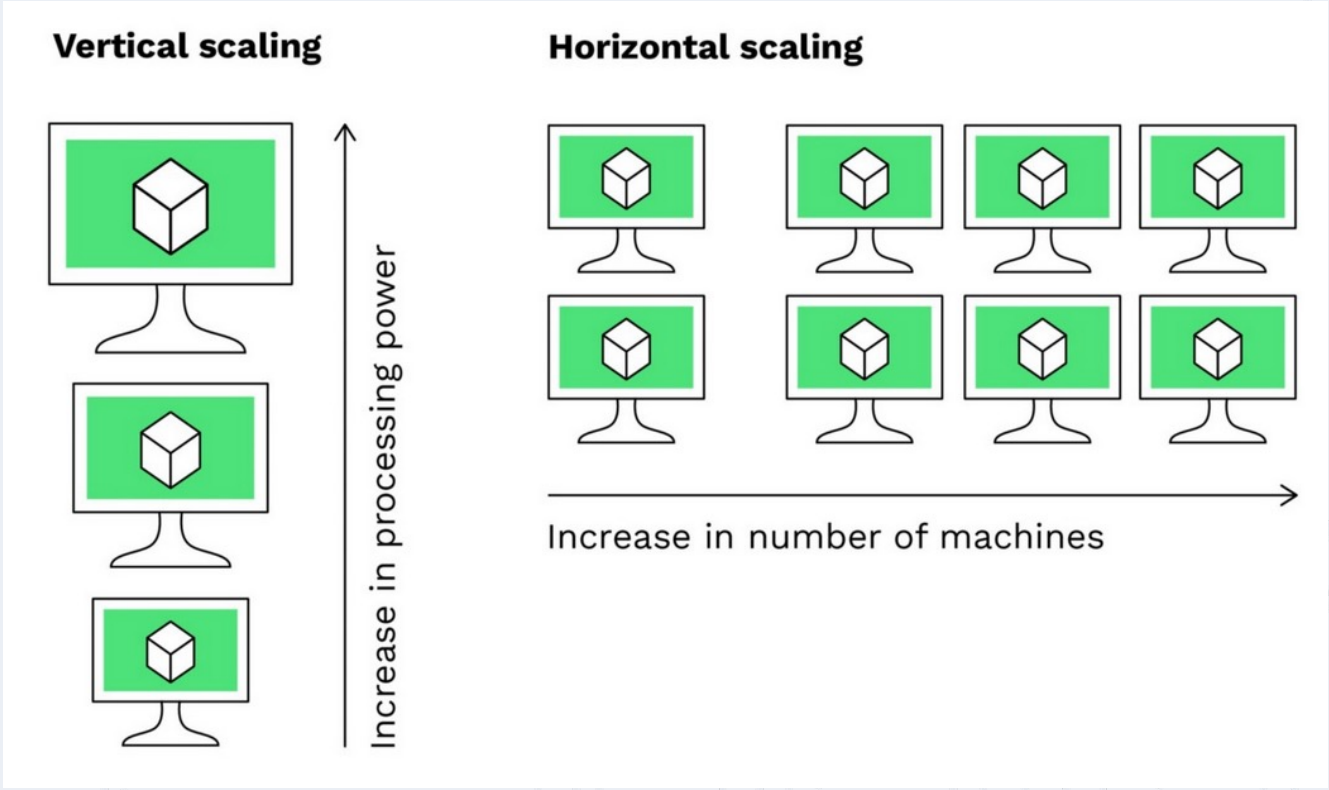


NoSQL Databases

Key Features of NoSQL Databases

1. **Flexible Data Models**: NoSQL databases store data in various formats such as JSON, XML, and key-value pairs, which allows for more flexible schema design and easier adaptation to changing data structures.
2. **Scalability**: NoSQL databases are designed for **horizontal scaling**, which enables them to handle large volumes of data and high traffic without significant performance degradation.
3. **Simple Design**: NoSQL databases often have simpler designs compared to relational databases, making them easier to implement and manage.
4. **High Availability**: NoSQL databases are designed to provide high availability and reliability by replicating data across multiple nodes and ensuring that data is accessible even if some nodes go offline.
5. **Support for SQL-like Query Languages**: Many NoSQL databases support SQL-like query languages, making it easier for developers familiar with SQL to work with these databases

Scalability



SQL vs NoSQL Terminology

SQL	MongoDB	DynamoDB	Cassandra	Couchbase
Table	Collection	Table	Table	Data bucket
Row	Document	Item	Row	Document
Column	Field	Attribute	Column	Field
Primary key	ObjectId	Primary key	Primary key	Document ID
Index	Index	Secondary index	Index	Index
View	View	Global secondary index	Materialized view	View
Nested table or object	Embedded document	Map	Map	Map
Array	Array	List	List	List

Types of NoSQL Databases

Document Databases

- Store data in flexible, semi-structured documents, often in JSON or XML format.
- Allow for nested data structures and flexible schemas, making them well-suited for handling unstructured and semi-structured data.
- Examples include MongoDB and Couchbase.

Key-Value Stores

- Store data as simple key-value pairs, where the value can be anything from a simple string to a complex object.
- Provide fast read/write operations and are optimized for simple data access.
- Examples include Redis and Memcached.

Types of NoSQL Databases

Column-Oriented Stores

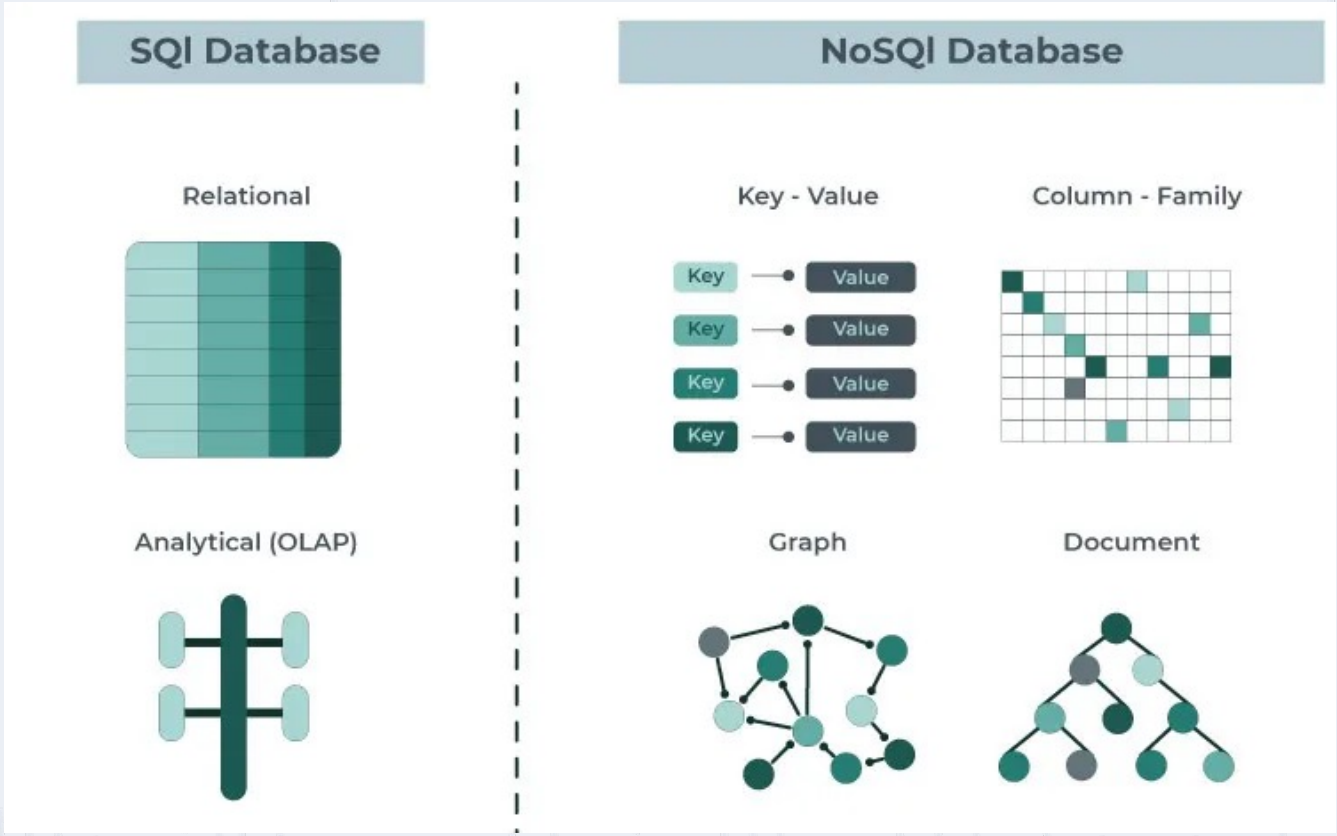
- Store data in columns instead of rows, which allows for efficient querying and analysis of large datasets.
- Organize data into column families, which are sets of columns treated as a single entity.
- Examples include Cassandra and HBase.

Graph Databases

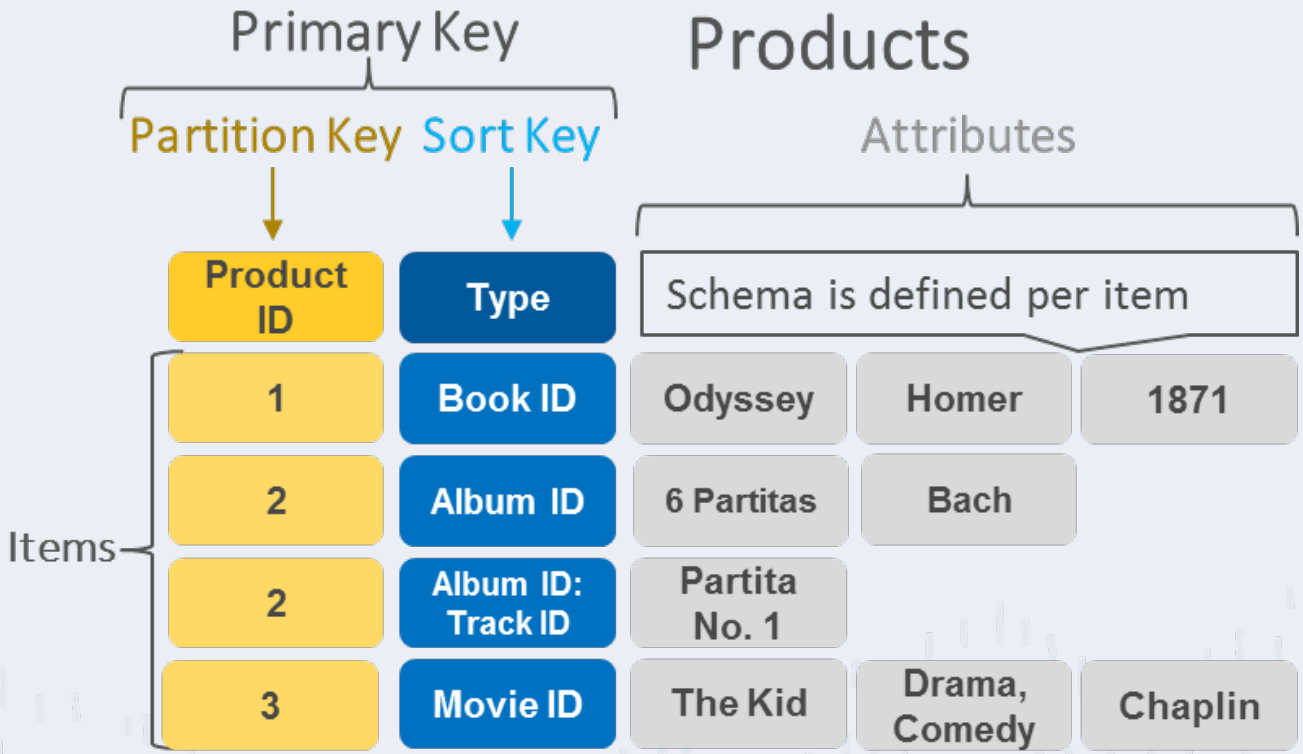
- Focus on the relationships between data, storing it in the form of nodes and edges.
- Designed to handle complex, interconnected data and are well-suited for applications that require traversing and analyzing relationships.
- Examples include Neo4j and Amazon Neptune.

abases:

SQL vs NoSQL



Key Value



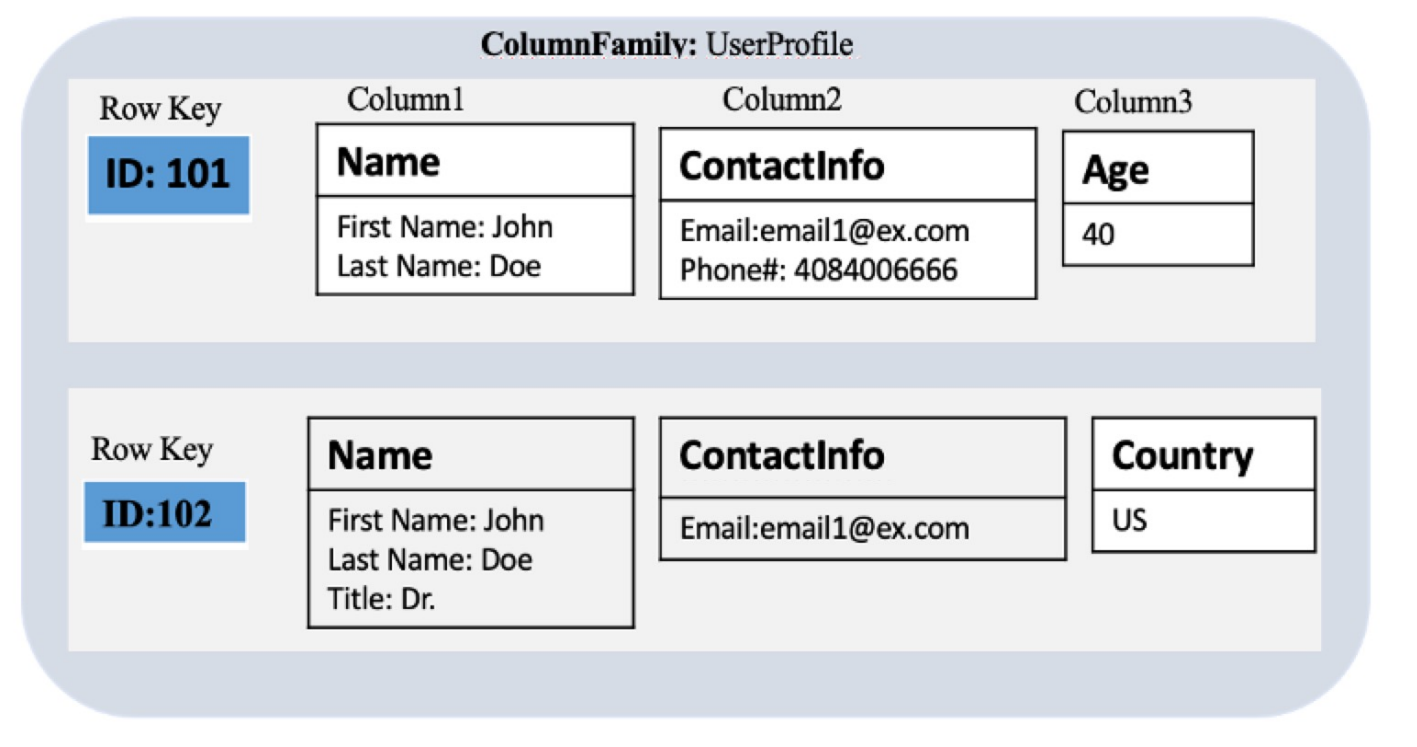
Document

Key	Document

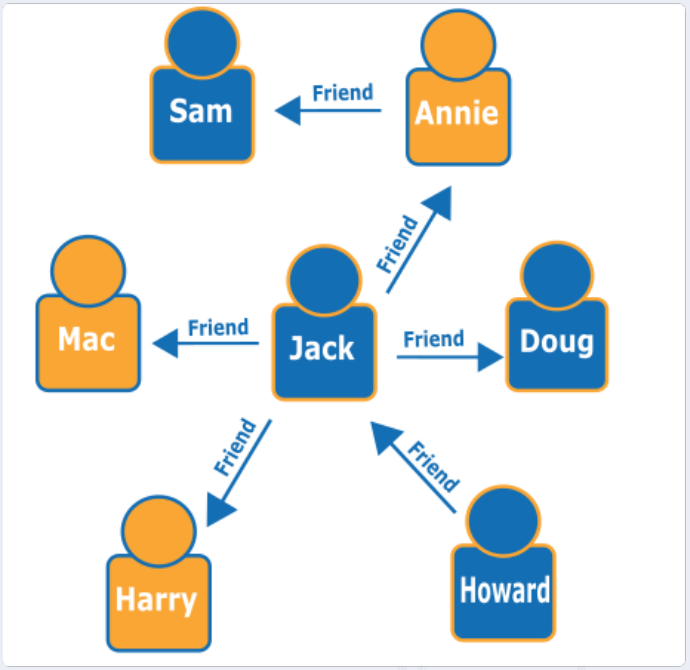
101	{
	"ID": "1001",
	"ItemsOrdered":[
	{
	"ItemID": "1",
	"Quantity": "2",
	"cost": "1000",
	},
	{
	"ItemID": "1001",
	"Quantity": "2",
	"cost": "1000",
	}] ,
	"OrderDate": "05/11/2019"
	}

102	"ID": "1002",
	"ItemsOrdered":[
	{
	"ItemID": "2890",
	"Quantity": "11",
	"cost": "10000",
	}]
	"OrderDate": "05/11/2019"
	}

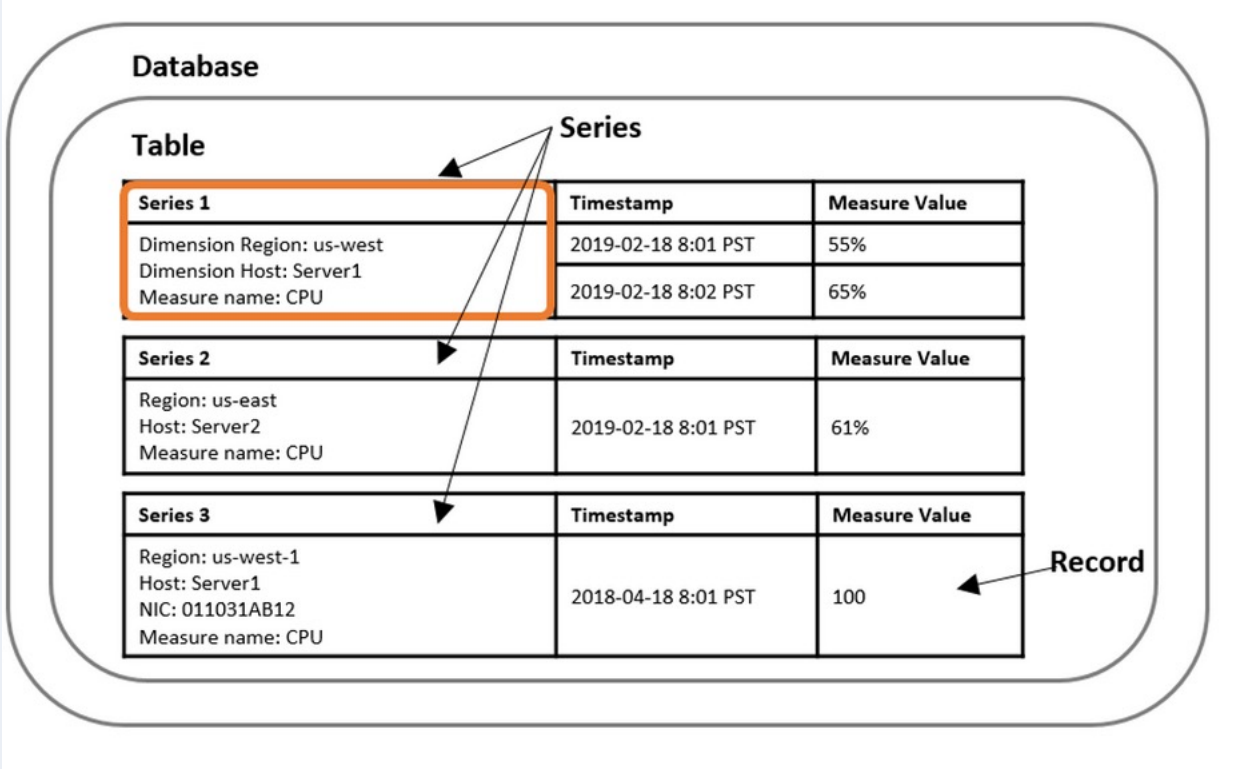
Wide Columns



Graph



Time Series



Intro to Amazon Web Services

AWS Certifications

FOUNDATIONAL

Knowledge-based certification for foundational understanding of AWS Cloud.
No prior experience needed.

PROFESSIONAL

Role-based certifications that validate advanced skills and knowledge required to design secure, optimized, and modernized applications and to automate processes on AWS.
2 years of prior AWS Cloud experience recommended.

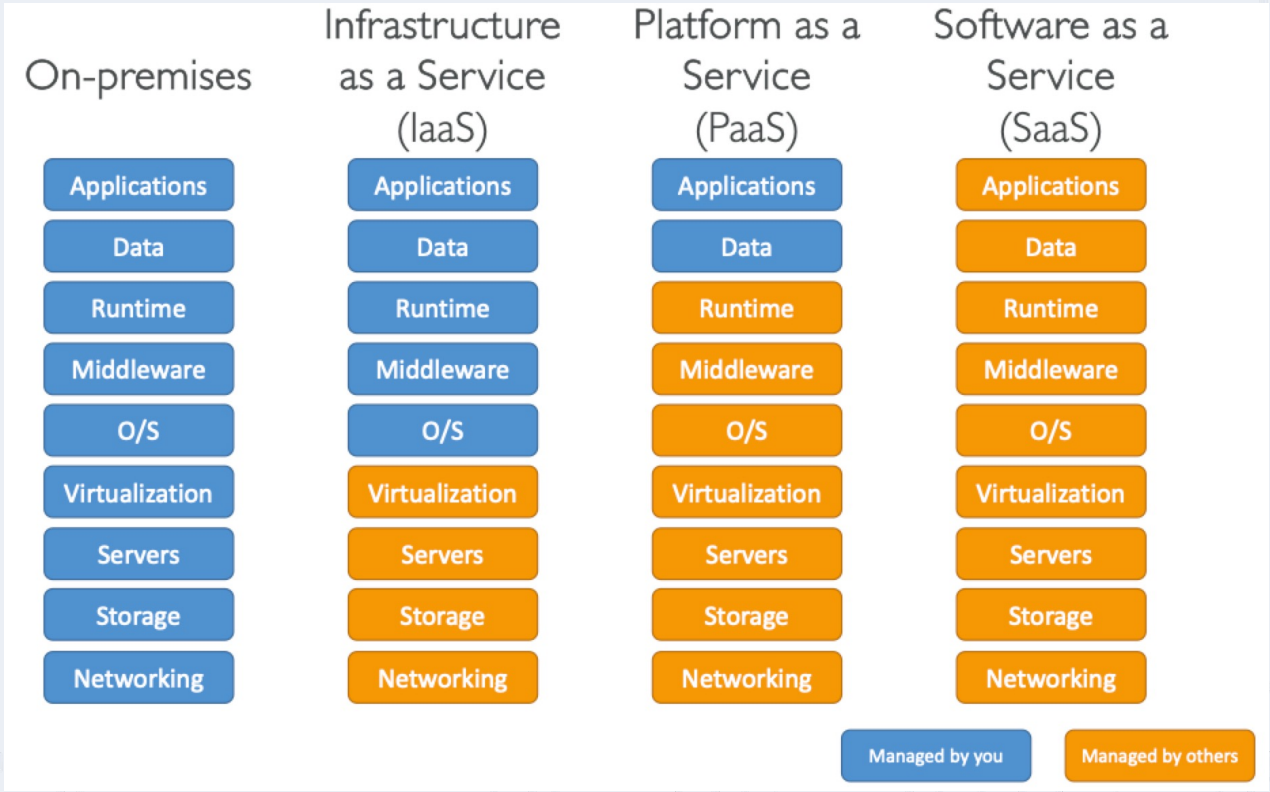
ASSOCIATE

Role-based certifications that showcase your knowledge and skills on AWS and build your credibility as an AWS Cloud professional. **Prior cloud and/or strong on-premises IT experience recommended.**

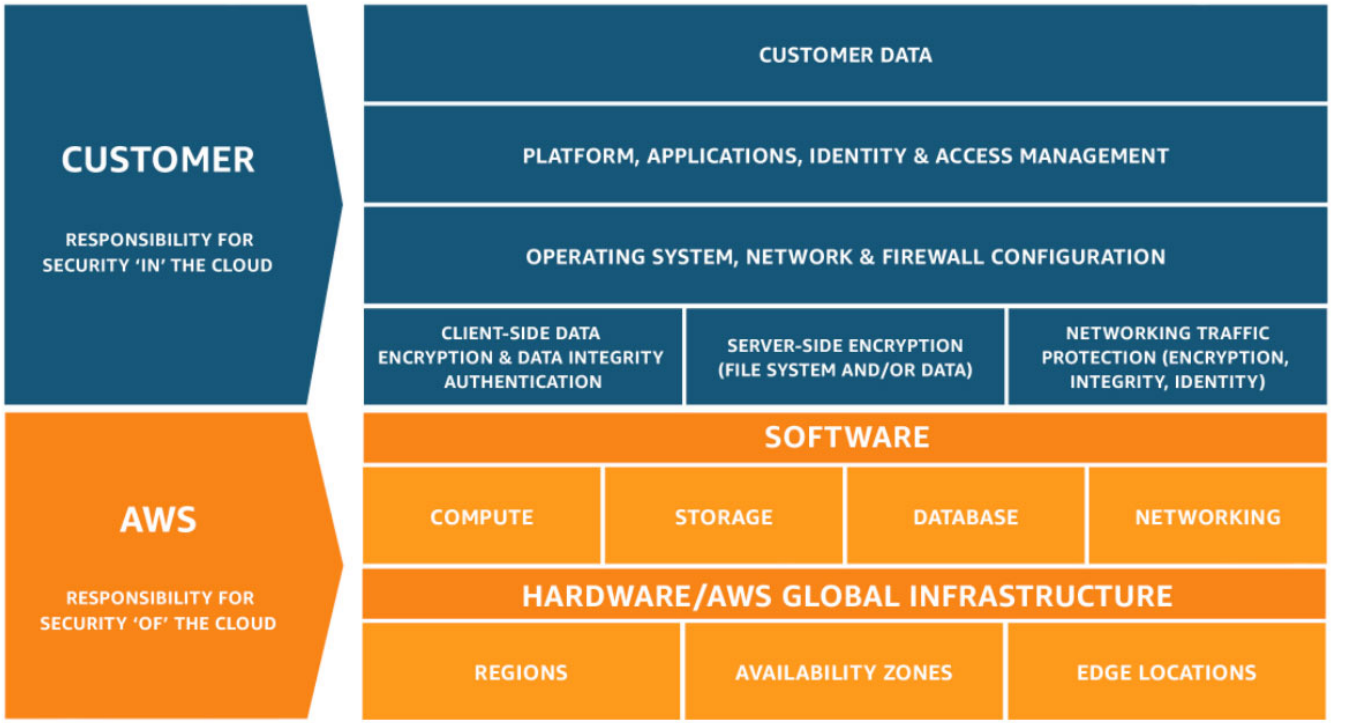
SPECIALTY

Dive deeper and position yourself as a trusted advisor to your stakeholders and/or customers in these strategic areas. **Refer to the exam guides on the exam pages for recommended experience.**

Types of Cloud Computing



AWS Shared Responsibility Model



AWS Regions and Availability Zones

https://aws.amazon.com/about-aws/global-infrastructure/regions_az/

Storage

Amazon S3

- ❑ Amazon S3 is one of the main building blocks of AWS
- ❑ Many websites for example use Amazon S3 as a backbone

Amazon S3 Use Cases

- Backup and Storage
- Disaster Recovery
- Archiving
- Hybrid Cloud Storage
- Application Hosting
- Media Hosting
- Data Lakes and Big Data Analytics
- Software Delivery
- Statis Websites

Amazon S3 - Buckets

- Store Objects “files” in buckets. Think of them as “directories”
- Buckets must have a globally unique name (across all regions and all accounts)
- Buckets are defined at the region level

Amazon S3 - Objects

- Objects (files) have a key
- The key is the full FULL path:
  s3://mybucket/myfile.txt
  s3://mybucket/folder1/subfolder1/myfile.txt
- The key is composed of a **prefix** + **object name**
  s3://mybucket/**folder1/subfolder1/myfile.txt**

- There is no concept of “directories” within buckets. Though in the UI you make think otherwise.
- They are just keys with very long names that contain ”/” slashes
- Object values are the content of the body: Max object size is 5TB, if large it must be ”multi-part” upload.
- Version ID if versioning is enabled
- Metadata

S3 Bucket Policies

JSON based policies

- Resources: buckets and objects
- Effect: Allow or Deny
- Actions: Set of API to allow or deny
- Principal: The account or user to apply the policy to

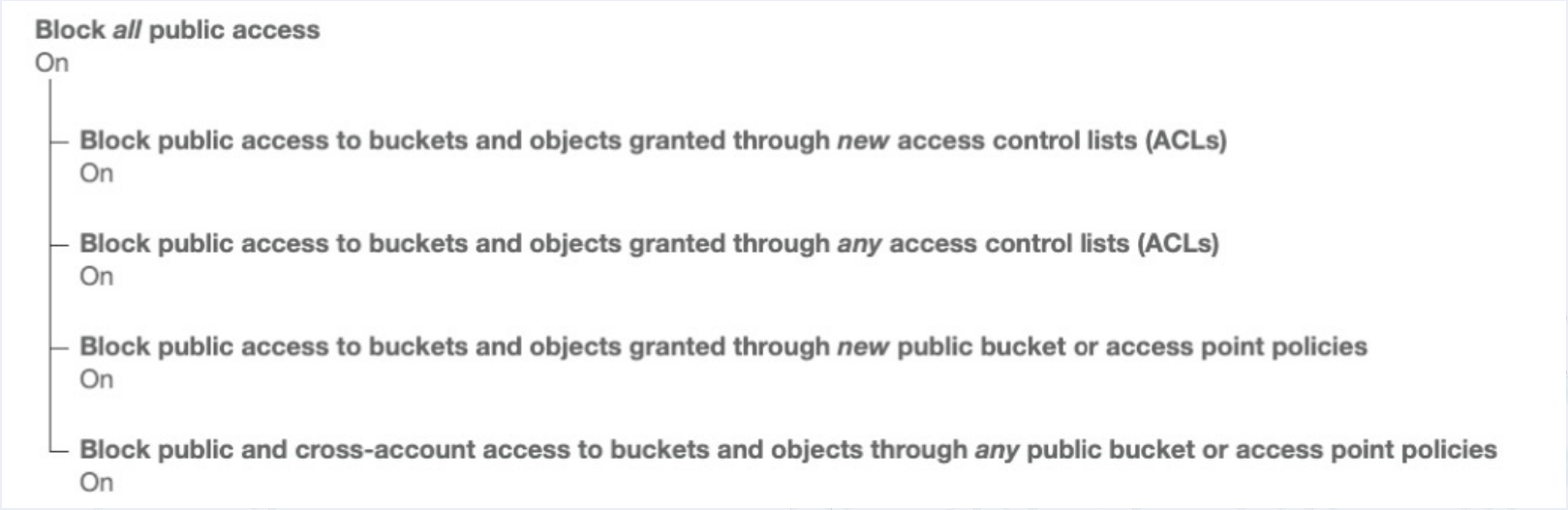
You use S3 Bucket policies to:

- Grant public access to the bucket
- Force objects to be encrypted at upload
- Grant access to another account (Cross Account)

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicRead",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::examplebucket/*"
            ]
        }
    ]
}
```

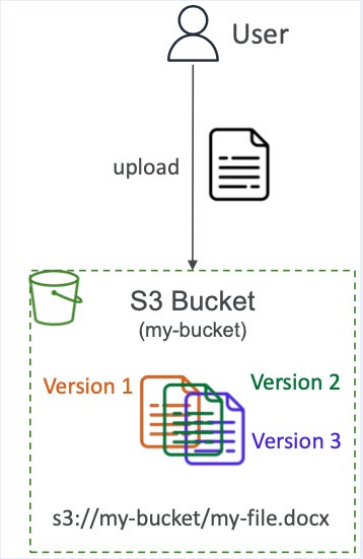
Block Public Access

- These settings were created to prevent company data leaks
- If your know your bucket should never be public, leave these on



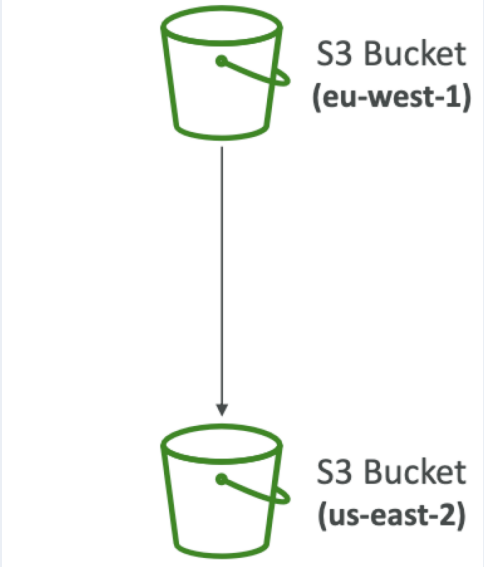
S3 Versioning

- You can version your files in Amazon S3
- It is enabled at the Bucket Level
- Same key overwrite will change the ”version”
- It is best practices to version you buckets
- A file without versioning will have a version “null”
- Suspending versioning does not delete the previous versions



S3 Replication

- Must enable Versioning in source and destination buckets
- Cross-Region Replication (CRR)
- Same-Region Replication (SRR)
- Bucket can be in different AWS accounts
- Copying is asynchronous
- Must give proper IAM permissions to S3
- Once you enable Replication, only new objects are replicated
- You can replicate existing objects using S3 Batch Replication



S3 Storage Classes

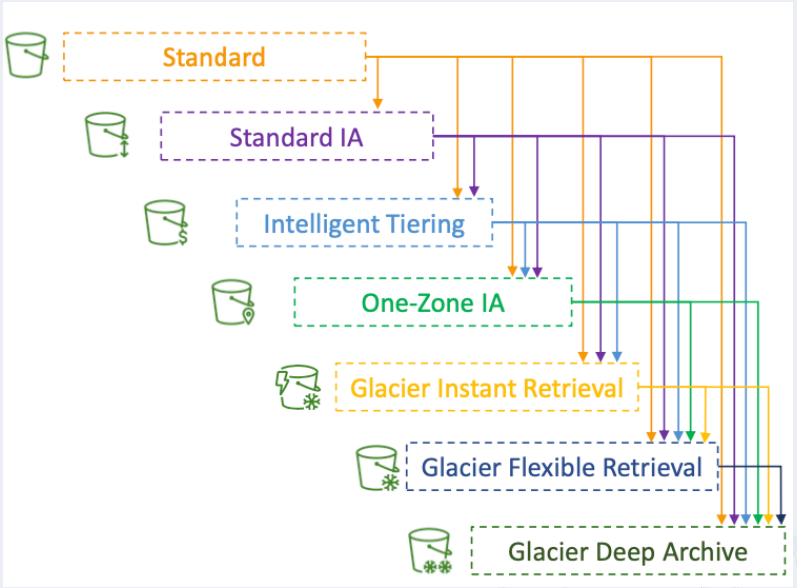
	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Durability	99.999999999% == (11 9's)						
Availability	99.99%	99.9%	99.9%	99.5%	99.9%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99%	99.9%	99.9%
Availability Zones	>= 3	>= 3	>= 3	1	>= 3	>= 3	>= 3
Min. Storage Duration Charge	None	None	30 Days	30 Days	90 Days	90 Days	180 Days
Min. Billable Object Size	None	None	128 KB	128 KB	128 KB	40 KB	40 KB
Retrieval Fee	None	None	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved

S3 Storage Classes

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Storage Cost (per GB per month)	$0.023	$0.0025 - $0.023	$0.0125	$0.01	$0.004	$0.0036	$0.00099
Retrieval Cost (per 1000 request)	**GET:** $0.0004 **POST:** $0.005	**GET:** $0.0004 **POST:** $0.005	**GET:** $0.001 **POST:** $0.01	**GET:** $0.001 **POST:** $0.01	**GET:** $0.01 **POST:** $0.02	**GET:** $0.0004 **POST:** $0.03 **Expedited:** $10 **Standard:** $0.05 **Bulk:** free	**GET:** $0.0004 **POST:** $0.05 **Standard:** $0.10 **Bulk:** $0.025
Retrieval Time	Instantaneous					**Expedited** (1 – 5 mins) **Standard** (3 – 5 hours) Bulk (5 – 12 hours)	**Standard** (12 hours) **Bulk** (48 hours)
Monitoring Cost (pet 1000 objects)		$0.0025					

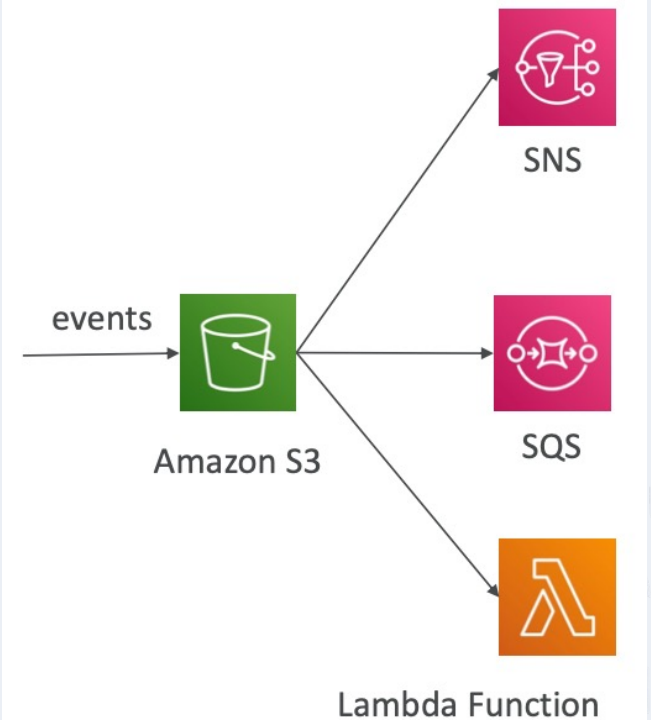
S3 Storage Classes

- You can transition objects between storage classes
- For infrequently access objects, move them to Standard IA
- For archive objects that you do not need fast access to, move them to Glacier or Glacier Deep Archive
- Moving objects can be automated using Lifecycle Rules

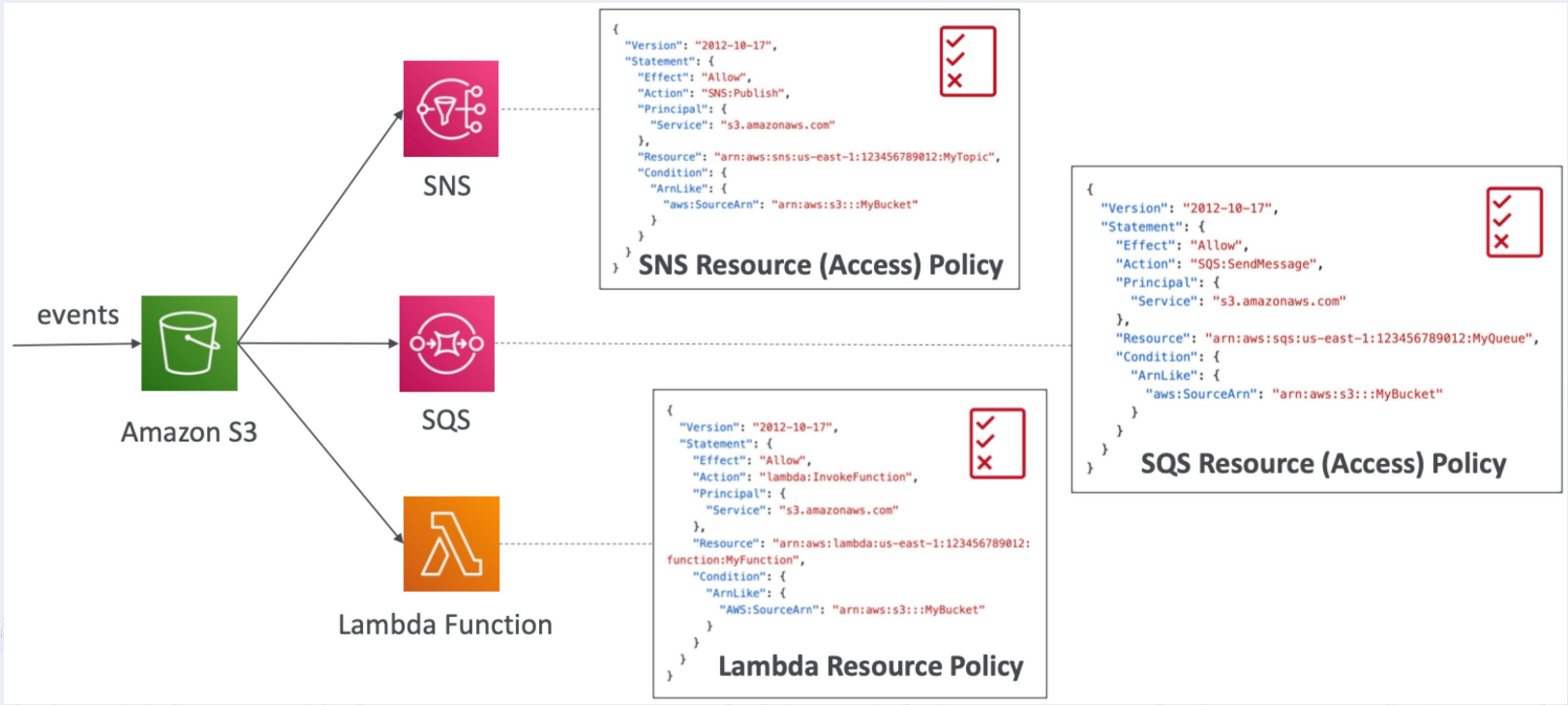


S3 Even Notification

- Event notification on ObjectCreated, ObjecteRemoved, ObjectRestore, ObjectReplication ..etc.
- Object name filtering possible example “*.jpg”
- S3 events notifications typically delivered in seconds but can sometimes take a minute or longer



S3 Even Notification – IAM Permissions



Databases

https://docs.aws.amazon.com/whitepapers/latest/aws-overview/database.html

AWS Database Services

Database	Use cases	AWS services
Relational	Traditional applications, enterprise resource planning (ERP), customer relationship management (CRM), e-commerce	•Amazon Aurora — Designed for unparalleled high performance and availability at global scale with full MySQL and PostgreSQL compatibility •Amazon RDS — Set up, operate, and scale a relational database in the cloud with just a few clicks •Amazon Redshift — Accelerate your time to insights with fast, easy, and secure cloud data warehousing at scale
Key-value	High-traffic web applications, e-commerce systems, gaming applications	•Amazon DynamoDB — Fast, flexible NoSQL database service for single-digit millisecond performance at any scale
In-memory	Caching, session management, gaming leaderboards, geospatial applications	•Amazon ElastiCache — Unlock microsecond latency and scale with in-memory caching •Amazon MemoryDB for Redis — Redis-compatible, durable, in-memory database service for ultra-fast performance
Document	Content management, catalogs, user profiles	•Amazon DocumentDB (with MongoDB compatibility) — Scale JSON workloads with ease using a fully managed document database service
Wide column	High-scale industrial apps for equipment maintenance, fleet management, and route optimization	•Amazon Keyspaces — A scalable, highly available, and managed Apache Cassandra–compatible database service
Graph	Fraud detection, social networking, recommendation engines	•Amazon Neptune — Build and run graph applications with highly connected datasets
Time series	Internet of Things (IoT) applications, DevOps, industrial telemetry	•Amazon Timestream — Fast, scalable, serverless time series database
Ledger	Systems of record, supply chain, registrations, banking transactions	•Amazon Ledger Database Service (QLDB) — Maintain an immutable, cryptographically verifiable log of data changes

Amazon RDS

Hosted Relational Database

- Amazon Aurora
- MySQL
- PostgreSQL
- MariaDB
- Oracle
- SQL Server

Analytics

https://docs.aws.amazon.com/whitepapers/latest/aws-overview/analytics.html

AWS Glue

Thank you!

If you have any additional questions, please reach out to me at: (email address).