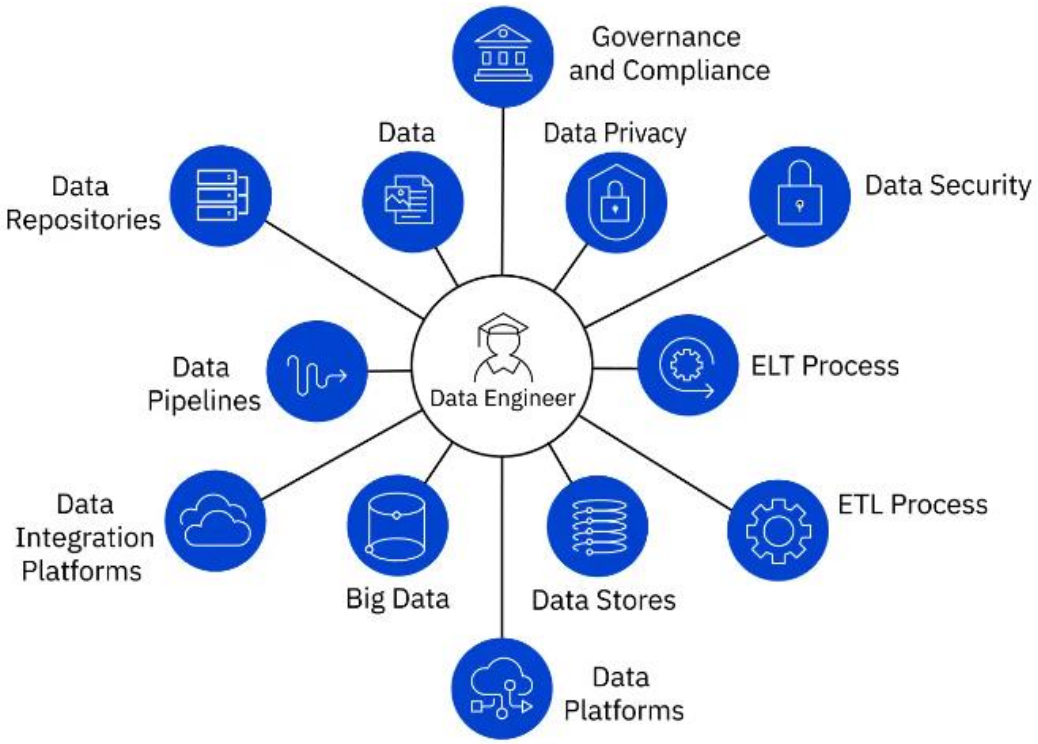


Responsibilities of a Data Engineer

At a broad level, Data Engineers:

Extract, organize, and integrate data from disparate sources

Prepare data for analysis and reporting by transforming and cleansing it

Design and manage data pipelines that encompass the journey of data from source to destination systems

Setup and manage the infrastructure required for the ingestion, processing, and storage of data
Data Platforms, Data Stores, Distributed Systems, Data Repositories

Technical Skills

Operating Systems

UNIX

Linux

Windows
Administrative Tools

System Utilities
& Commands

Databases and Data Warehouses

RDBMS

IBM DB2, MySQL,
Oracle Database,
PostgreSQL

NoSQL

Redis, MongoDB,
Cassandra, Neo4J

Data Warehouses

Oracle Exadata, IBM
Db2 Warehouse on Cloud,
IBM Netezza Performance
Server, Amazon RedShift

Data Pipelines

Apache Beam

AirFlow

DataFLow

ETL Tools

Glue

Technical Skills

Languages

Query languages

SQL for relational databases and SQL-like query languages for NoSQL databases

Programming languages

Python, R, Java

Shell and Scripting languages

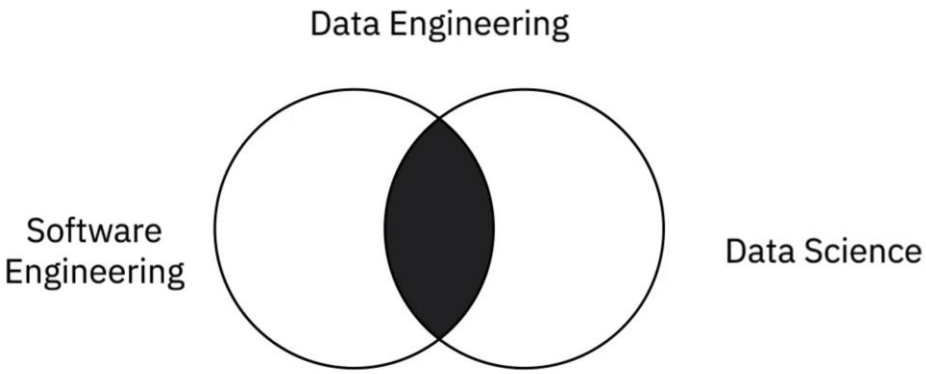
Unix/Linux Shell and PowerShell

Big Data Processing Tools

Functional Skills:

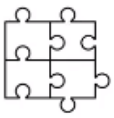
Functional Skills

Data Engineering is at the intersection of Software engineering and Data Science.



Functional Skills of a Data Engineer:

Convert business requirements into technical specifications



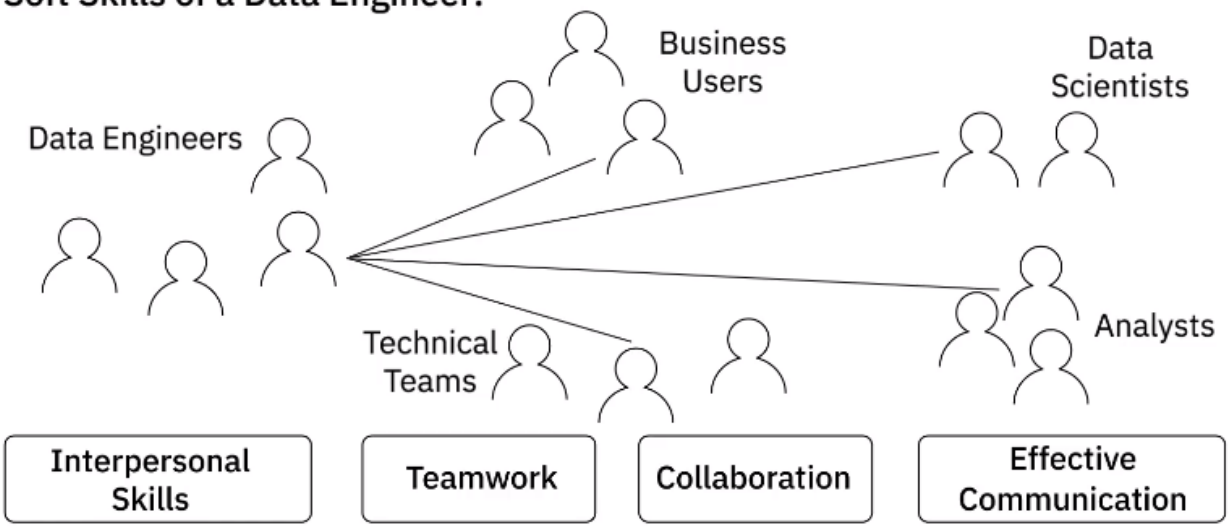
Work with the complete software development lifecycle
Ideation -> Architecture -> Design -> Prototyping -> Testing -> Deployment -> Monitoring

Functional Skills of a Data Engineer:

Understand data's potential application in business

Understand risks of poor data management
Data Quality | Data Privacy | Security | Compliance

Soft Skills of a Data Engineer:

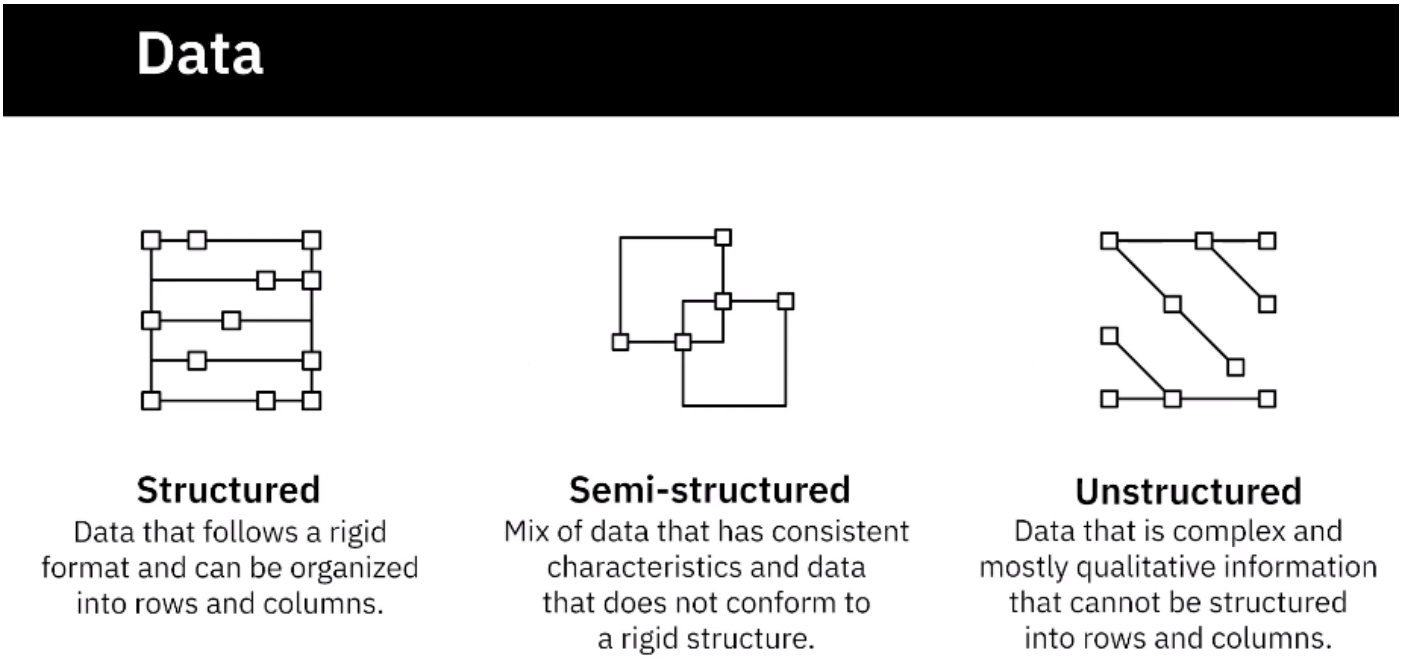


Video: Viewpoints: Skills and Qualities to be a Data Engineer (6:58)

- Experience required in Retail:
  - Relational databases
  - Cassandra
  - Google Bigtable
  - Kafka Stream
  - WebSphere MQ

Video: A Day in the Life of a Data Engineer (3:37)

Video: Overview of the Data Engineering Ecosystem (4:53)

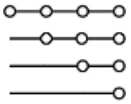


Data Sources

Data

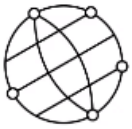
Data also comes in a wide-ranging variety of file formats being collected from a variety of data sources,

Relational Database



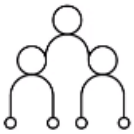
Non-Relational Database

APIs

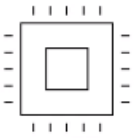


Web Services

Data Streams



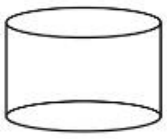
Social Platforms



Sensor Devices

Data Repositories

Data Repositories

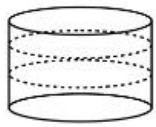


Transactional or Online Transaction Processing (OLTP) System

Designed to store high volume day-to-day operational data

Typically relational, but can also be non-relational

Data Repositories



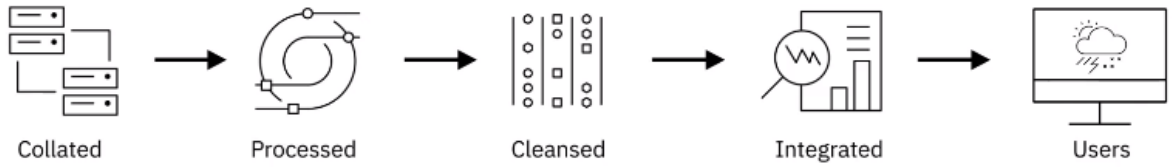
Analytical
or
Online Analytical
Processing (OLAP)
Systems

Optimized for conducting complex
data analytics

Include relational and non-relational
databases, data warehouses, data marts,
data lakes, and big data stores

Data Integartion

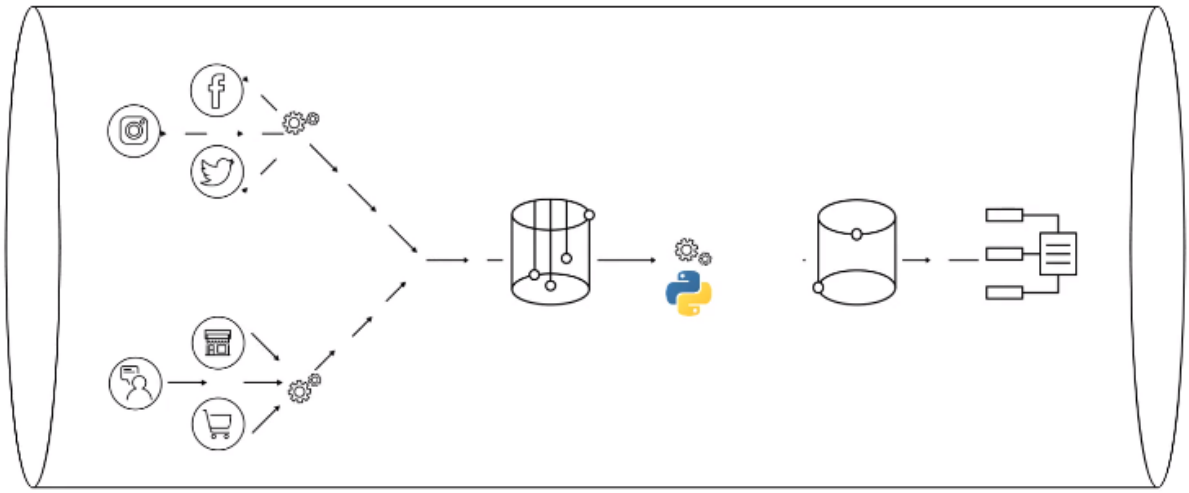
Data Integration



Combine data from disparate sources into a unified view,
accessed by users to query and manipulate the data.

Data Pipeline

Data Pipeline

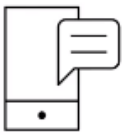


A set of tools and processes that cover the entire journey of data from source to destination systems.

Languages

Languages

Languages available in the Data Analyst Ecosystem:



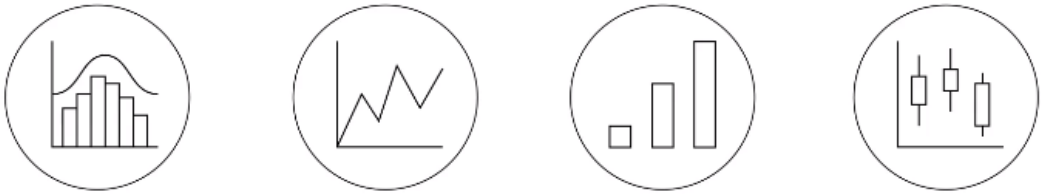
Query languages
For example, SQL for querying and manipulating data

Programming languages
For example, Python for developing data applications

Shell and Scripting languages
For repetitive operational tasks

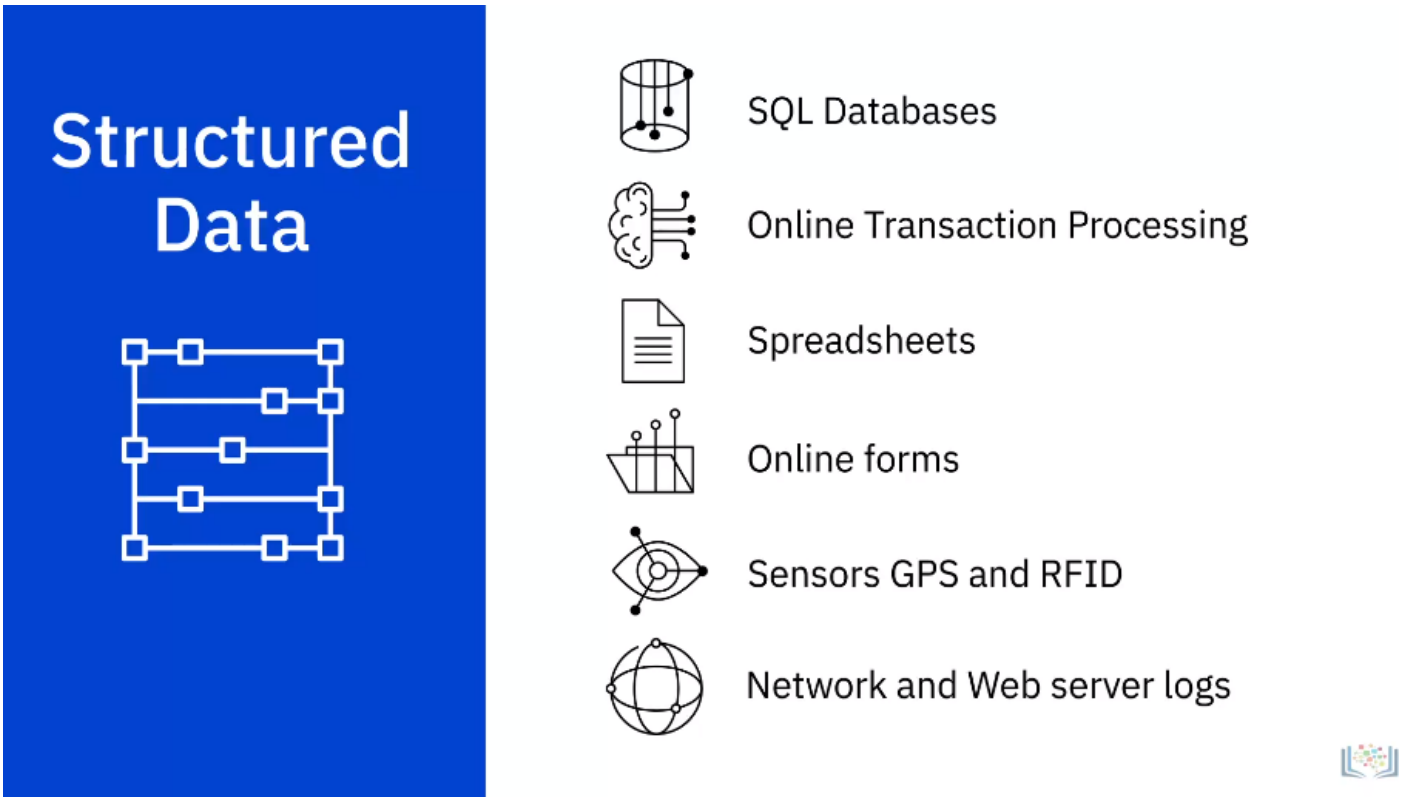
BI and Reporting Tools

Business Intelligence (BI) and Reporting Tools



- Collect data from multiple data sources and present them in a visual format, such as interactive dashboards
- Visualize data in real-time and pre-defined schedule
- Drag and drop products that do not require knowledge of programming

Video: Types of Data (4:02)



1. Structured data has a well-defined structure or adheres to a specified data model
2. can be stored in well-defined schemas such as databases

3. and in many cases can be represented in a tabular manner with rows and columns.
4. **Structured data is objective facts and numbers that can be collected, exported, stored, and**
5. organized in typical databases.
6. Some of the sources of structured data could include:
7. SQL Databases and Online Transaction Processing (or OLTP) Systems that focus on business transactions
8. Spreadsheets such as Excel and Google Spreadsheets
9. Online forms
10. Sensors such as Global Positioning Systems (or GPS) and Radio Frequency Identification
11. (or RFID) tags; and Network and Web server logs.
12. You can typically store structured data in relational or SQL databases.
13. You can also easily examine structured data with standard data analysis methods and tools.

Video: Understanding Different Types of File Formats (5:00)

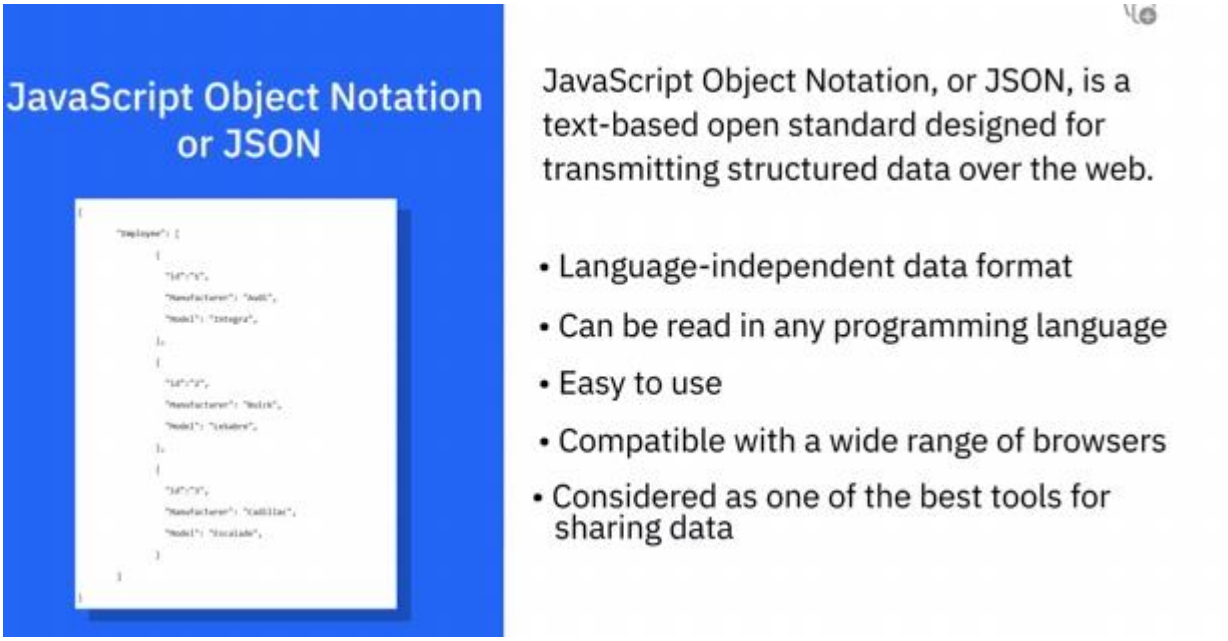
CSV

XLSX

XML

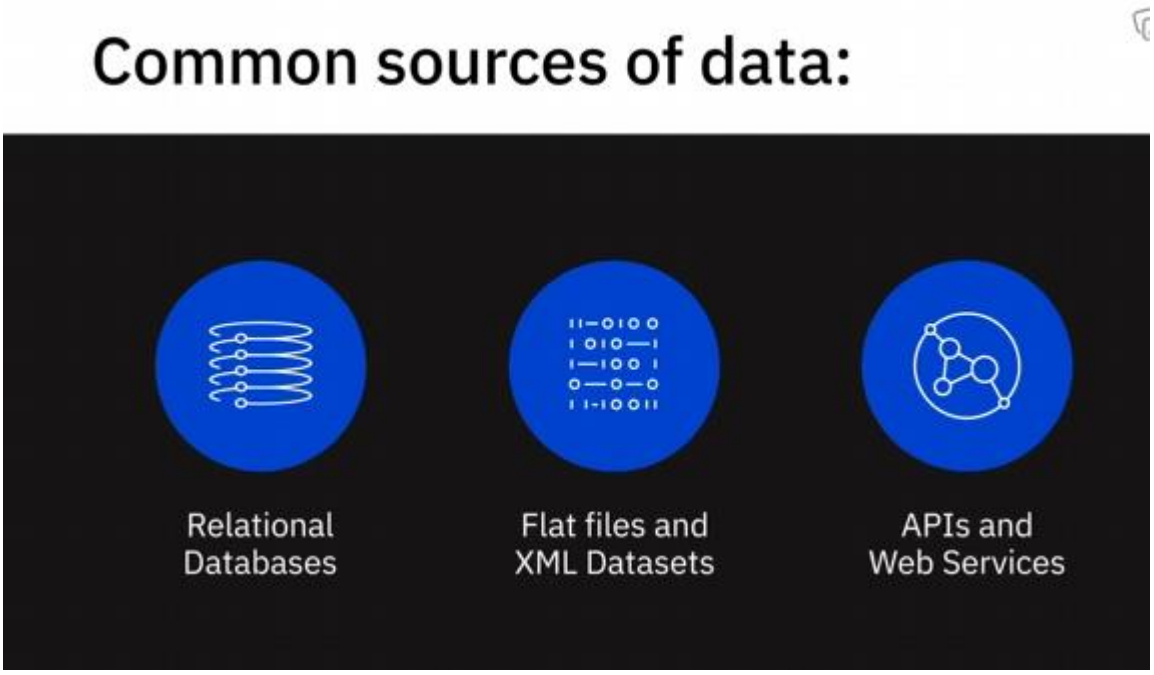
PDF

JSON



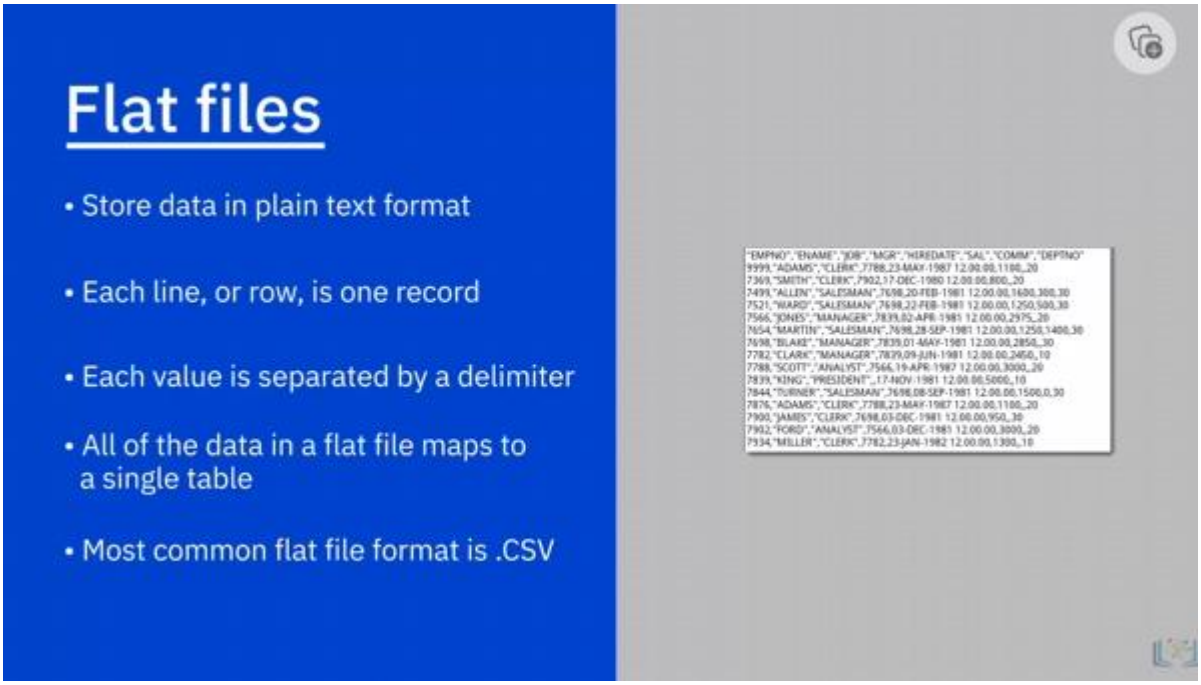
Video: Sources of Data (7:57)

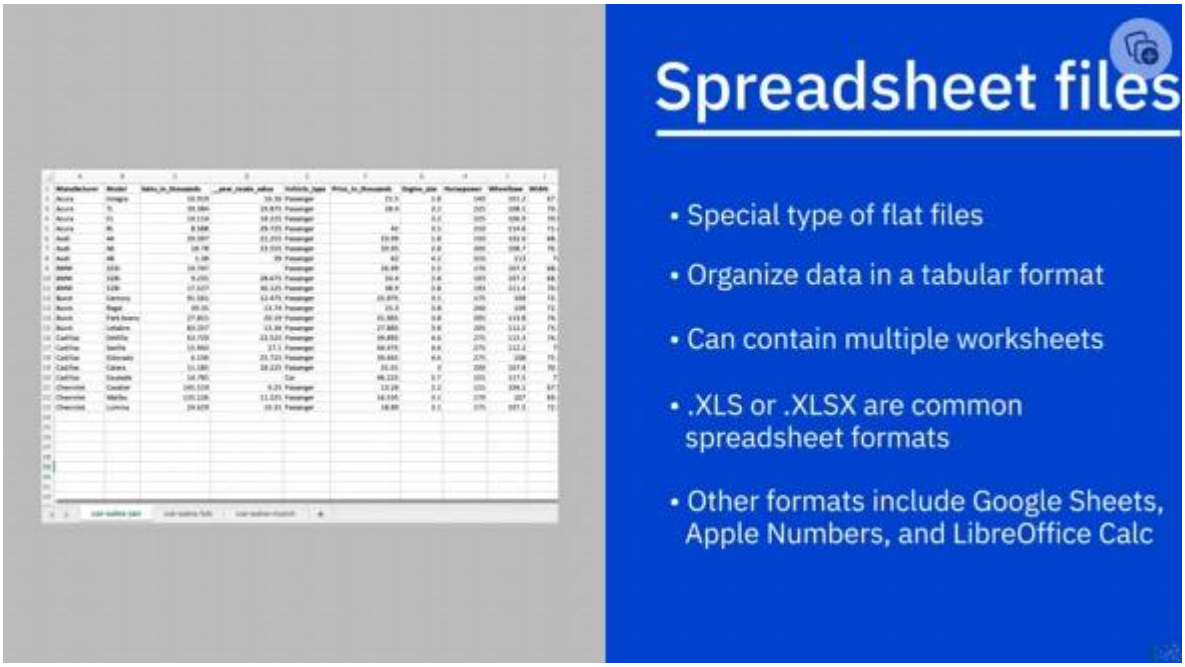
Common Sources of data

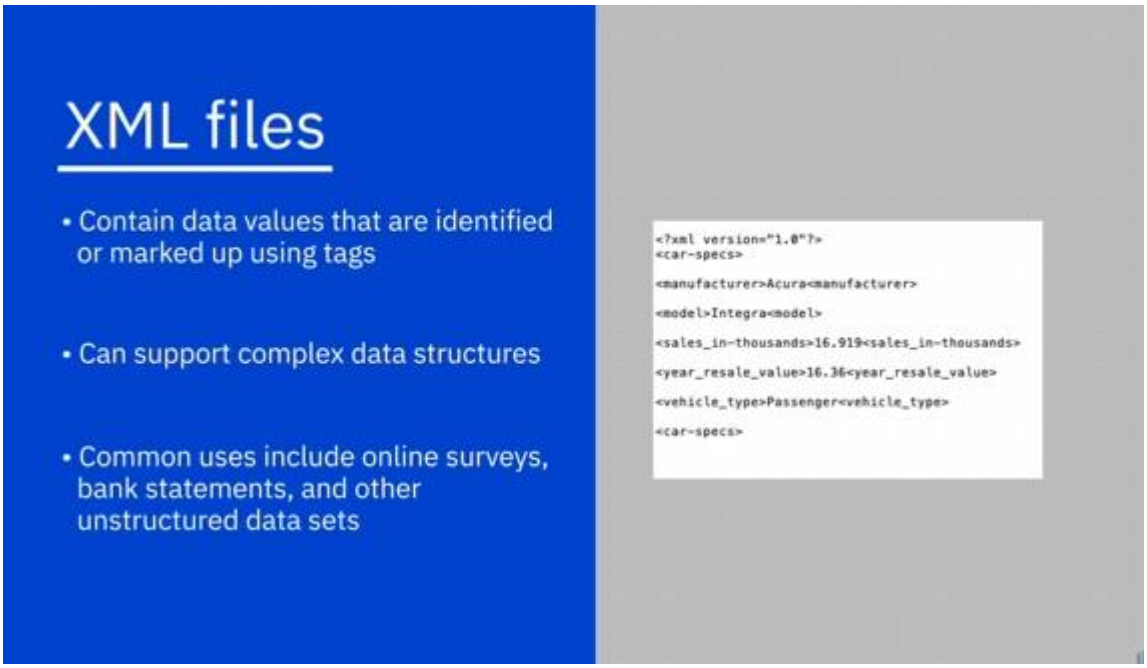


Relational Databases

Flat Files







1. XML files, contain data values that are identified or marked up using tags.
2. While data in flat files is “flat” or maps to a single table, XML files can support
3. more complex data structures, such as hierarchical.
4. Some common uses of XML include data from online surveys, bank statements, and other
5. unstructured data sets.

APIs

1. Many data providers and websites provide APIs, or Application Program Interfaces, and Web

2. Services, which multiple users or applications can interact with and obtain data for processing
3. or analysis.
4. APIs and Web Services typically listen for incoming requests, which can be in the form
5. **of web requests from users or network requests from applications, and return data in plain**
6. text, XML, HTML, JSON, or media files.
7. Let's look at some popular examples of APIs being used as a data source for data analytics:
8. The use of Twitter and Facebook APIs to source data from tweets and posts for performing
9. tasks such as opinion mining or sentiment analysis—which is to summarize the amount
10. of appreciation and criticism on a given subject, such as policies of a government, a product,
11. a service, or customer satisfaction in general.
12. Stock Market APIs used for pulling data such as share and commodity prices, earnings per
13. share, and historical prices, for trading and analysis.
14. Data Lookup and Validation APIs, which can be very useful for Data Analysts for cleaning
15. and preparing data, as well as for co-relating data—for example, to check which city or
16. state a postal or zip code belongs to.
17. APIs are also used for pulling data from database sources, within and external to the organization.

Web Scraping

1. Web ScrapingWeb scraping is used to extract relevant data from unstructured sources.
2. Also known as screen scraping, web harvesting, and web data extraction, web scraping makes
3. it possible to download specific data from web pages based on defined parameters.
4. **Web scrapers can, among other things, extract text, contact information, images, videos,**
5. product items, and much more from a website.
6. Some popular uses of web scraping include
7. collecting product details from retailers, manufacturers, and eCommerce websites to provide
8. price comparisons;
9. generating sales leads through public data sources;

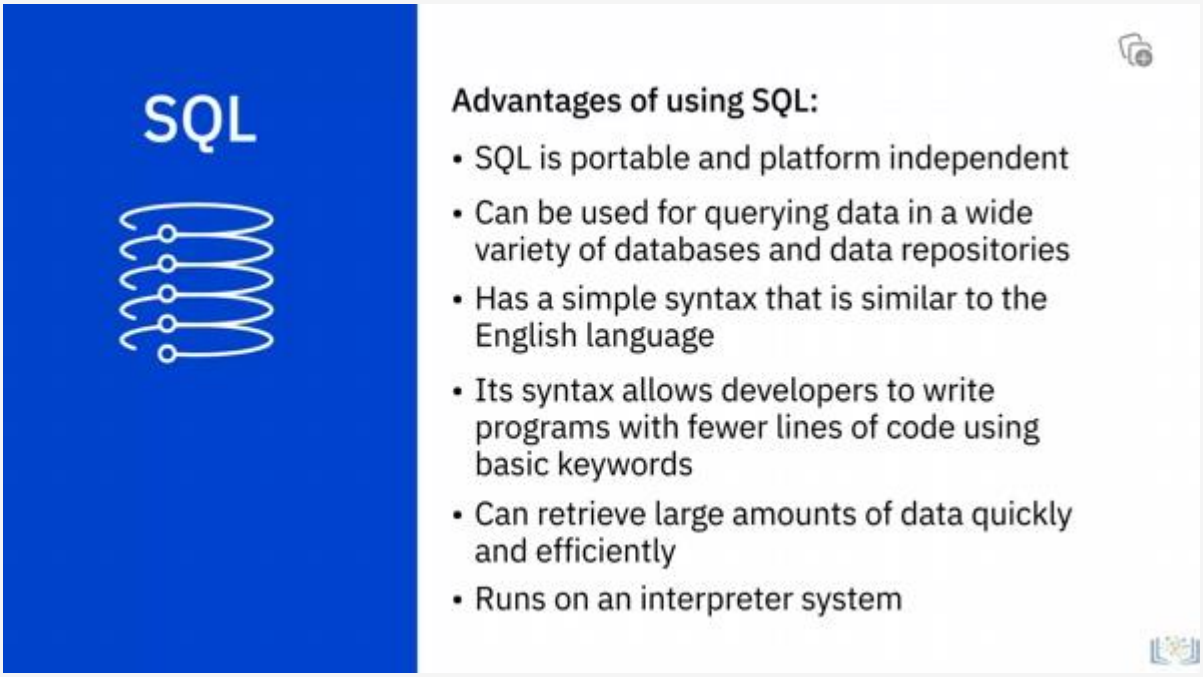
10. extracting data from posts and authors on various forums and communities;
11. and collecting training and testing datasets for machine learning models
12. Some of the popular web scraping tools include BeautifulSoup, Scrapy, Pandas, and Selenium.

Data Streams and Feeds

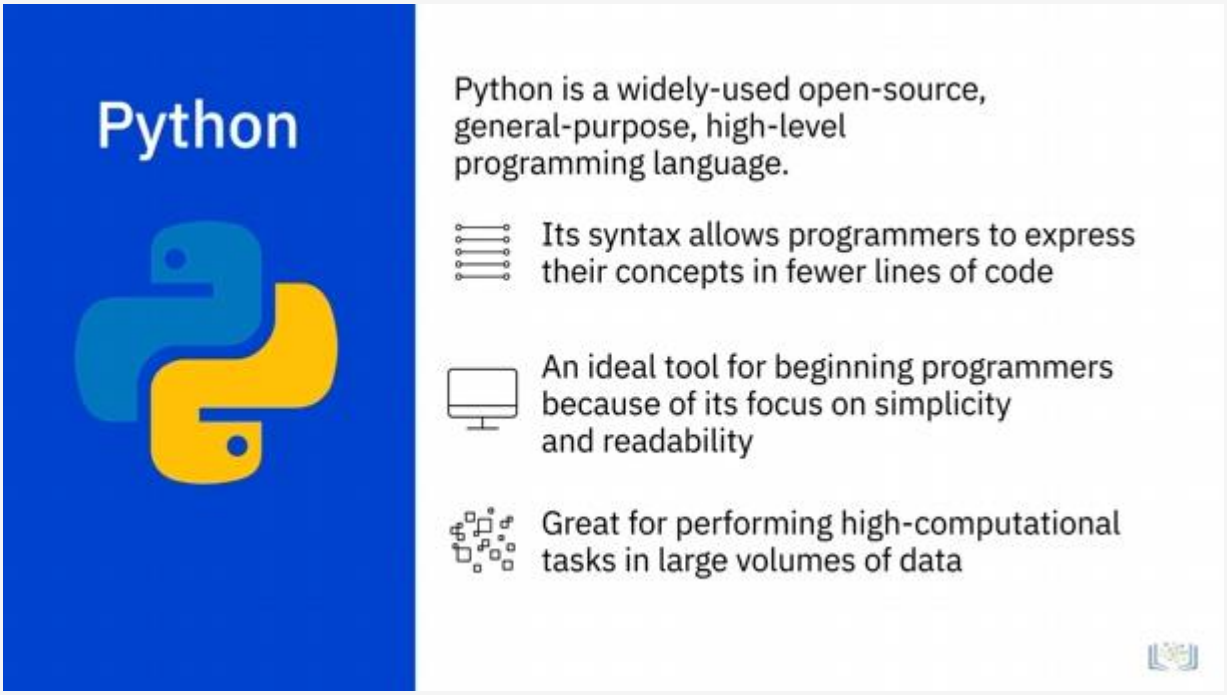
1. Data streams are another widely used source for aggregating constant streams of data flowing
2. from sources such as instruments, IoT devices, and applications, GPS data from cars, computer
3. programs, websites, and social media posts.
4. This data is generally timestamped and also geo-taggedfor geographical identification.
5. Some of the data streams and ways in which they can be leveraged include:
6. stock and market tickers for financial trading;
7. retail transaction streams for predicting demand and supply chain management;
8. surveillance and video feeds for threat detection;
9. **social media feeds for sentiment analysis;•sensor data feeds for monitoring industrial or farming**
10. machinery;
11. web click feeds for monitoring web performanceand improving design; and
12. real-time flight events for rebooking and rescheduling.
13. Some popular applications used to process data streams include Apache Kafka, Apache
14. Spark Streaming, and Apache Storm.
15. RSS (or Really Simple Syndication) feeds, are another popular data source.
16. These are typically used for capturing updated data from online forums and news sites where
17. data is refreshed on an ongoing basis.
18. Using a feed reader, which is an interface that converts RSS text files into a stream
19. of updated data, updates are streamed to user devices.

Video: Languages for Data Professionals (8:20)

SQL:


SQL
Advantages of using SQL:
• SQL is portable and platform independent
• Can be used for querying data in a wide variety of databases and data repositories
• Has a simple syntax that is similar to the English language
• Its syntax allows developers to write programs with fewer lines of code using basic keywords
• Can retrieve large amounts of data quickly and efficiently
• Runs on an interpreter system

Python


Python
Python is a widely-used open-source, general-purpose, high-level programming language.
Its syntax allows programmers to express their concepts in fewer lines of code
An ideal tool for beginning programmers because of its focus on simplicity and readability
Great for performing high-computational tasks in large volumes of data

Python

Python is a widely-used open-source, general-purpose, high-level programming language.

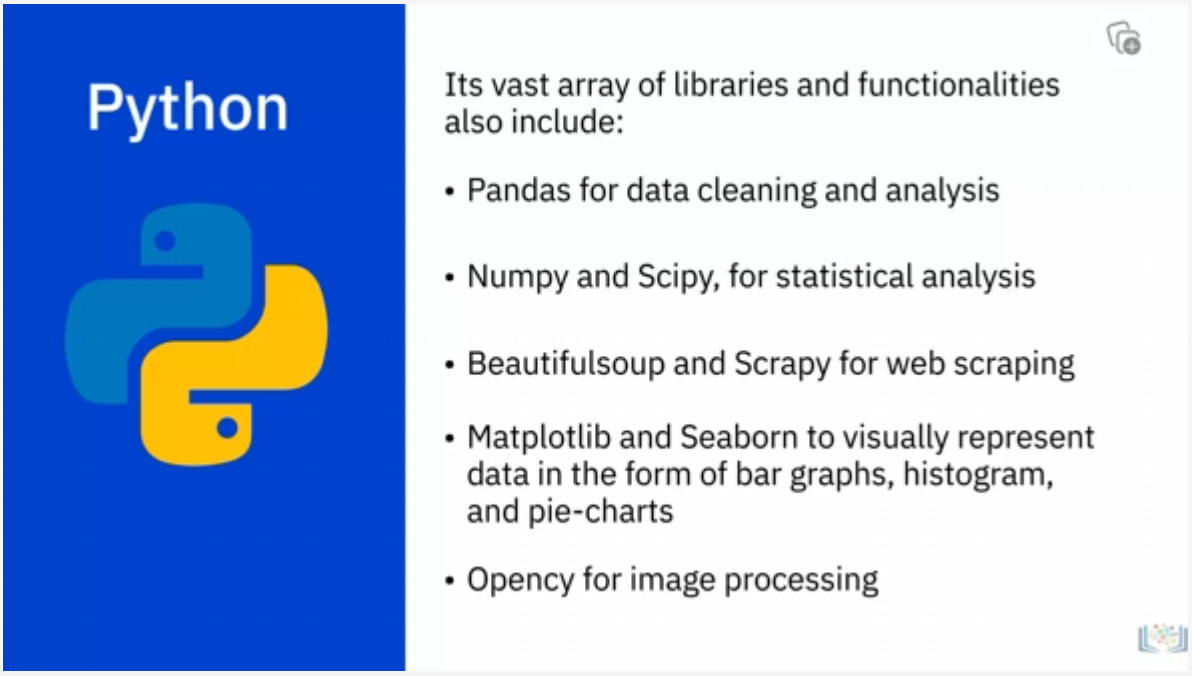
Has in-built functions for frequently used concepts

Supports multiple programming paradigms – object-oriented, imperative, functional, and procedural

Python

Python is one of the fastest-growing programming languages in the world.

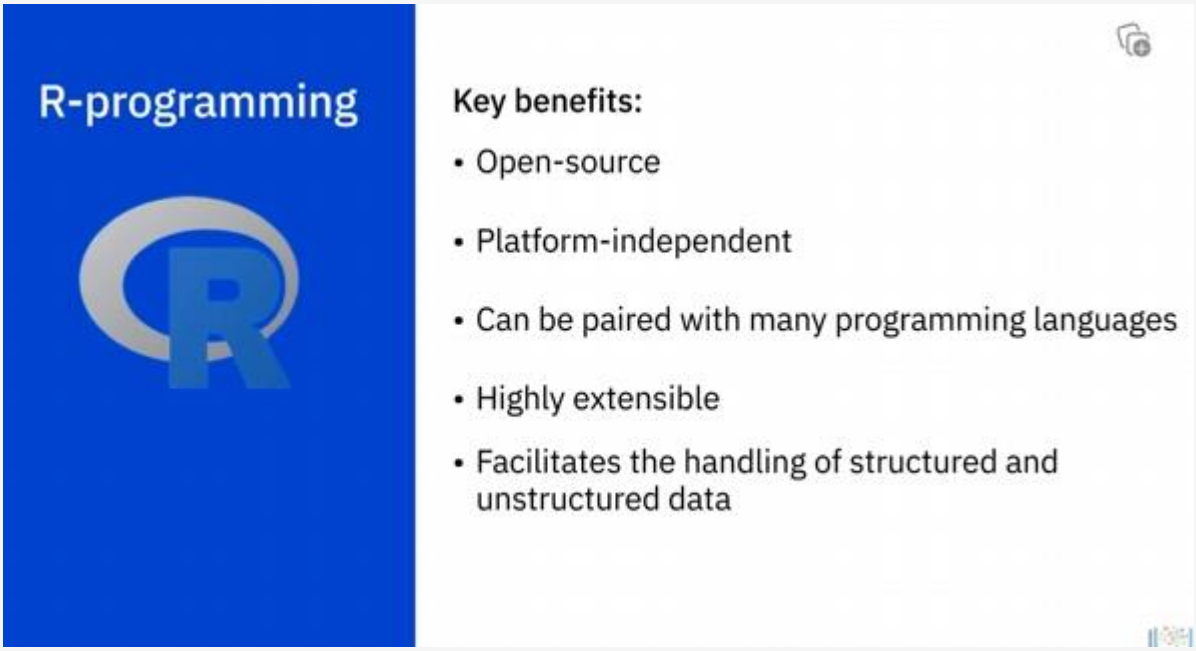
- Easy to learn
- Open-source
- Can be ported to multiple platforms
- Has widespread community support
- Provides open-source libraries for data manipulation, data visualization, statistics, mathematics

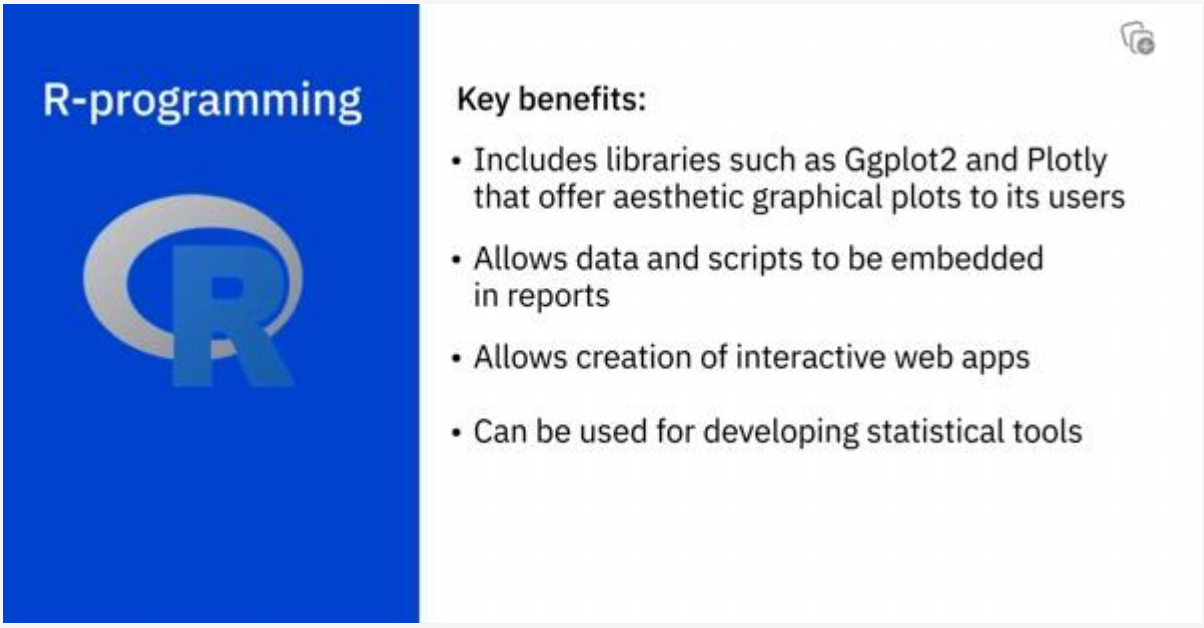

Python
Its vast array of libraries and functionalities also include:
• Pandas for data cleaning and analysis
• Numpy and Scipy, for statistical analysis
• Beautifulsoup and Scrapy for web scraping
• Matplotlib and Seaborn to visually represent data in the form of bar graphs, histogram, and pie-charts
• Opency for image processing

R

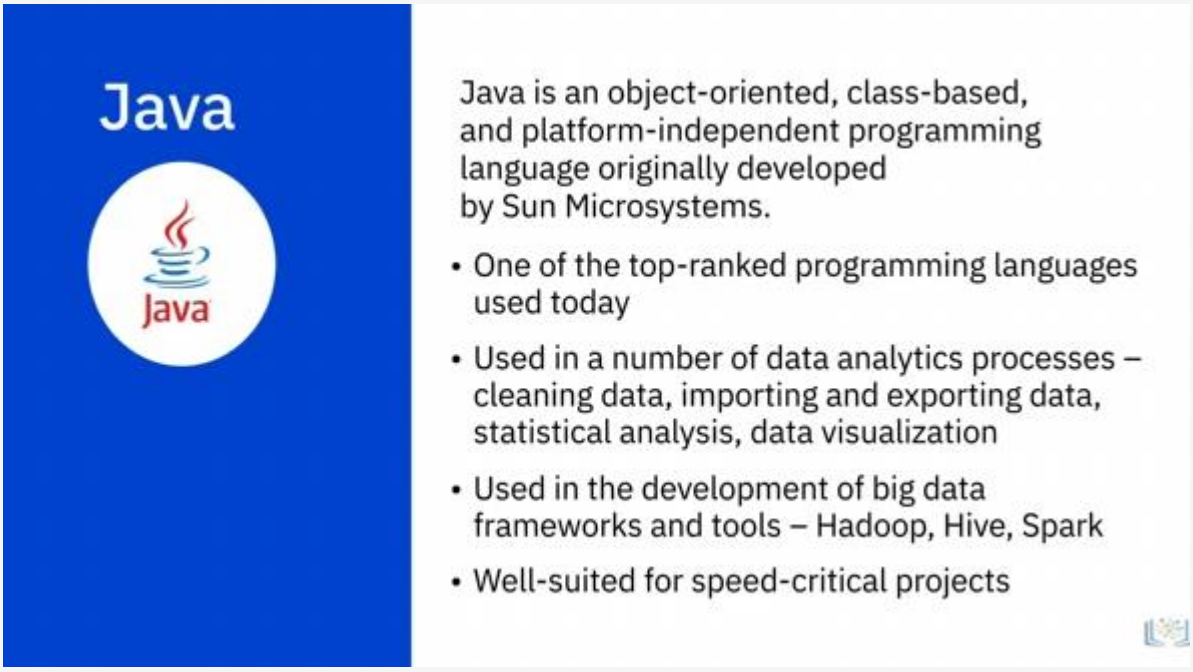

R-programming
R is an open-source programming language and environment for data analysis, data visualization, machine learning, and statistics.
Widely used for:
• Developing statistical software
• Performing data analytics
• Creating compelling visualizations

R Benefits

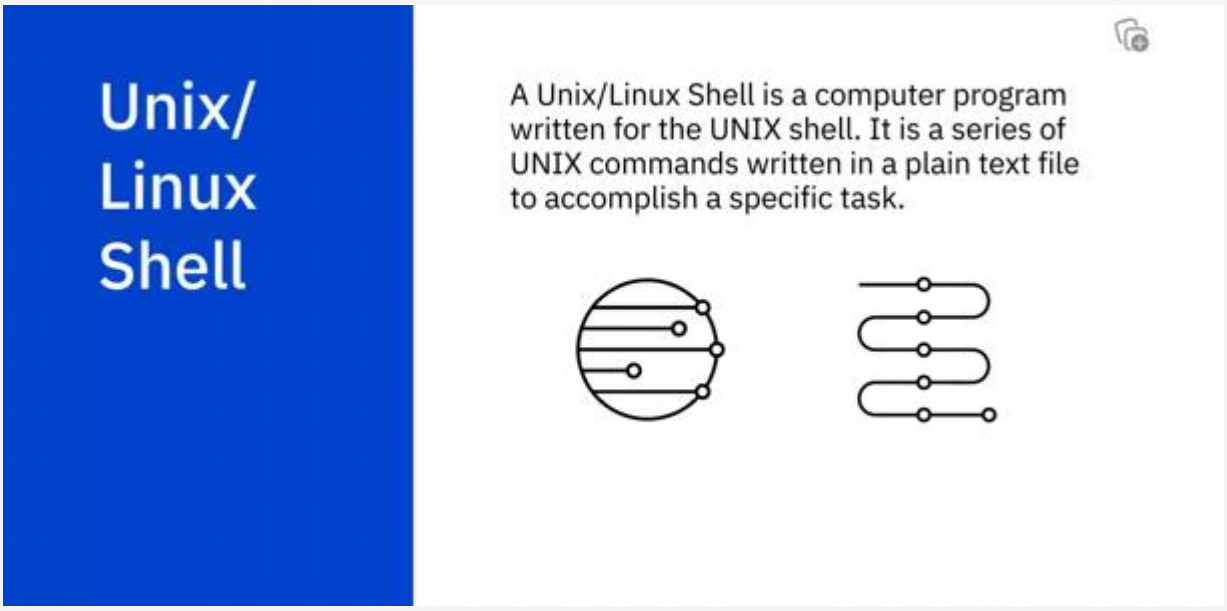

R-programming
Key benefits:
• Open-source
• Platform-independent
• Can be paired with many programming languages
• Highly extensible
• Facilitates the handling of structured and unstructured data


R-programming
Key benefits:
• Includes libraries such as Ggplot2 and Plotly that offer aesthetic graphical plots to its users
• Allows data and scripts to be embedded in reports
• Allows creation of interactive web apps
• Can be used for developing statistical tools

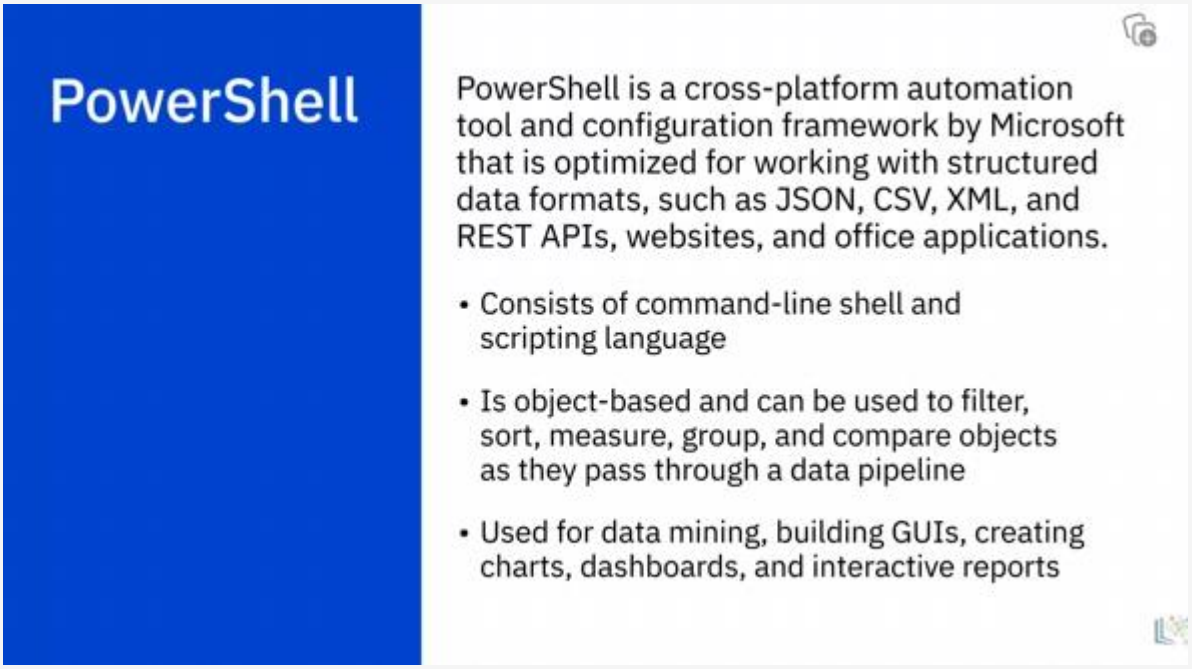
Java



[Unix/Linux Shell]



[PowerShell]



Video: Viewpoints: Working with Varied Data Sources and Types (6:38)

Flexibility

Cassandra and Hbase: for solving problems

Heavy write application

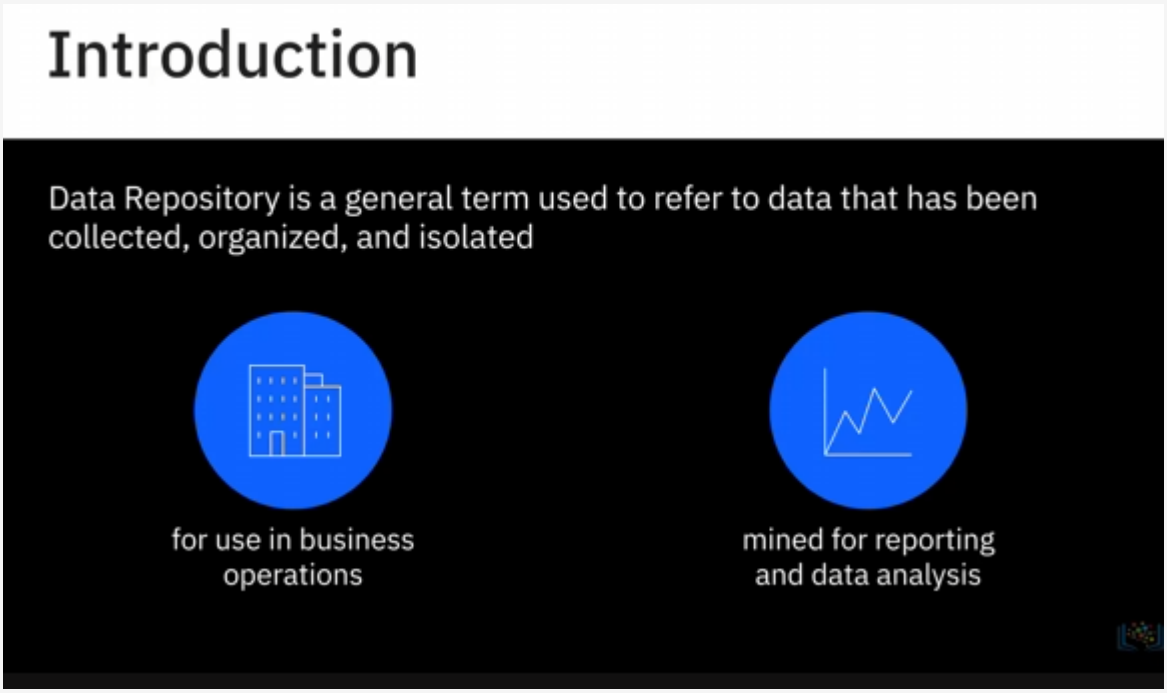
Variety of data sources

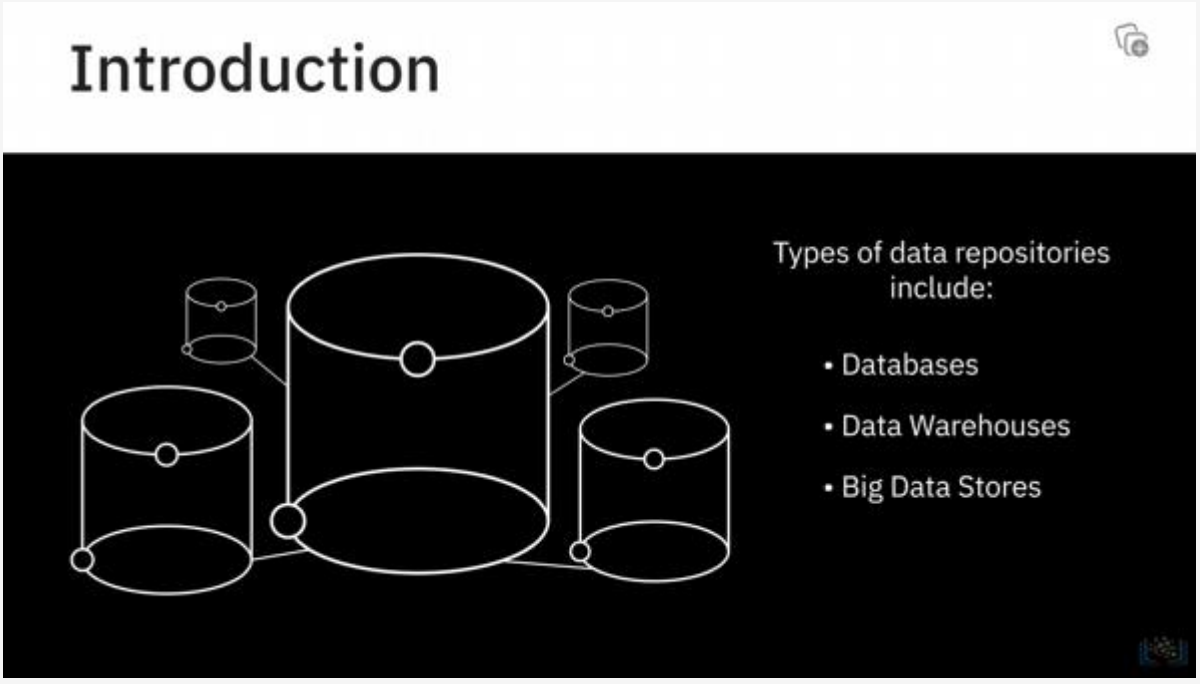
Pros and Cons of Different Data Formats

Apache Arvo: efficiency of storage of data

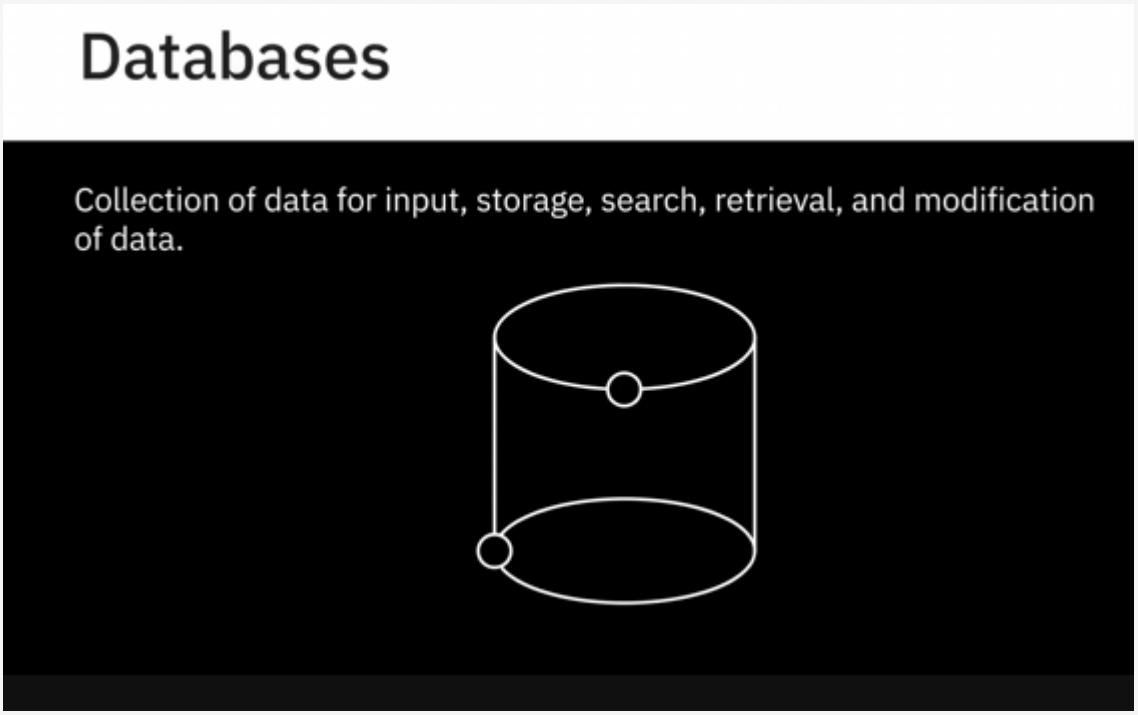
Video: Overview of Data Repositories (4:34)

Data Repositories Introduction

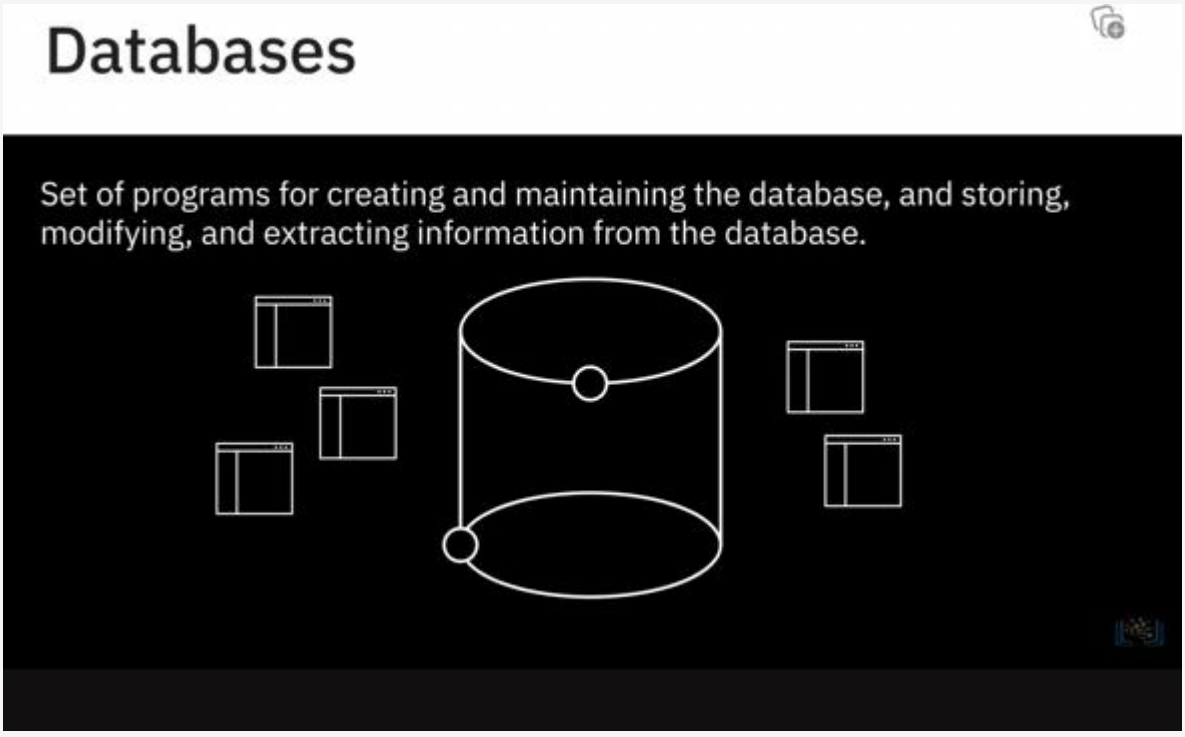




Databases

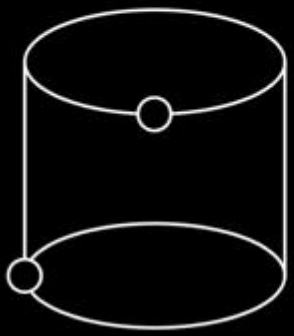


[Database Management System/ DBMS]



Databases

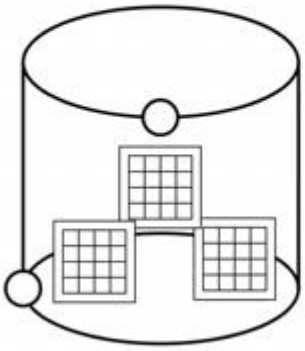
Factors governing choice of database include:



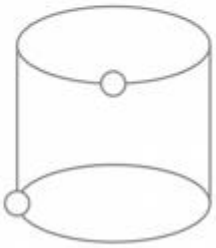
- Data type
- Data structure
- Querying mechanisms
- Latency requirements
- Transaction speeds
- Intended use of data

Relational Database:

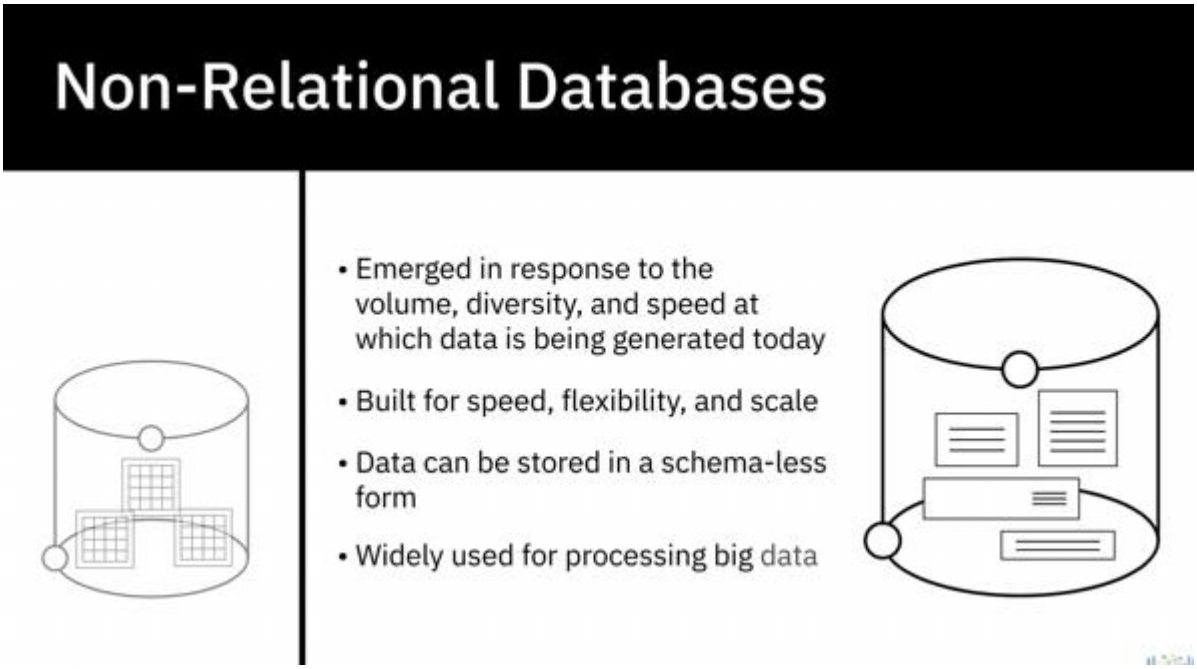
Relational Databases



- Data is organized into a tabular format with rows and columns
- Well-defined structure and schema
- Optimized for data operations and querying
- Use SQL as the standard querying language

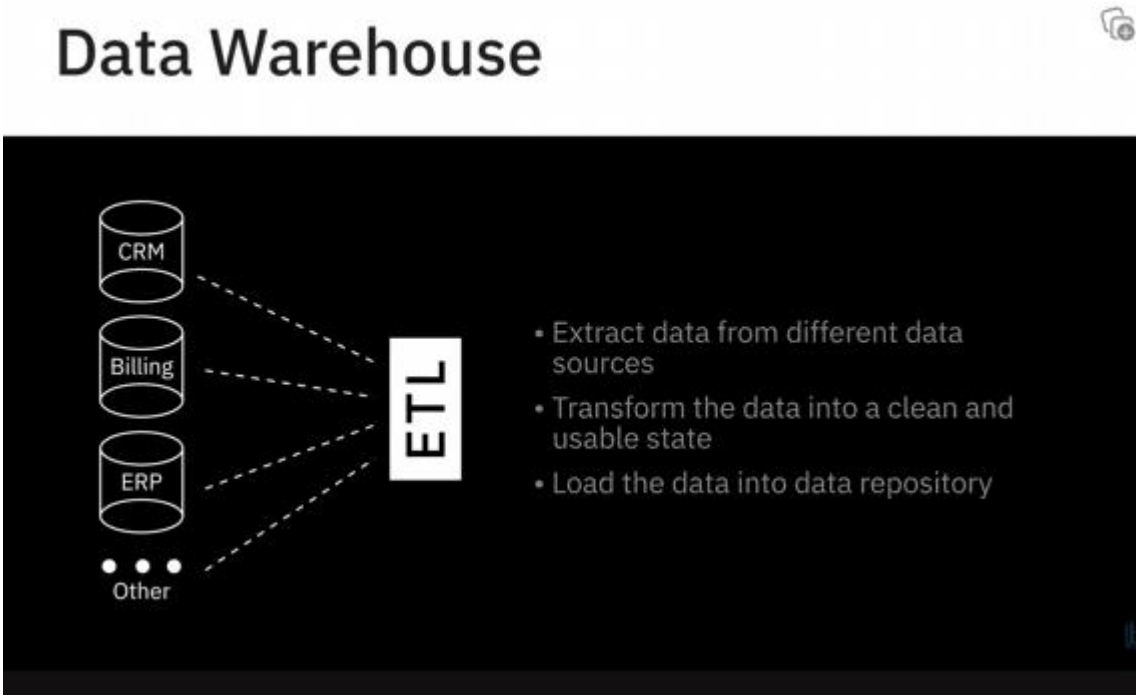


Non-Relational Database



Data Warehouse

1. A data warehouse works as a central repository that merges information coming from disparate
2. sources and consolidates it through the extract, transform, and load process, also known as
3. **the ETL process, into one comprehensive database for analytics and business intelligence.**

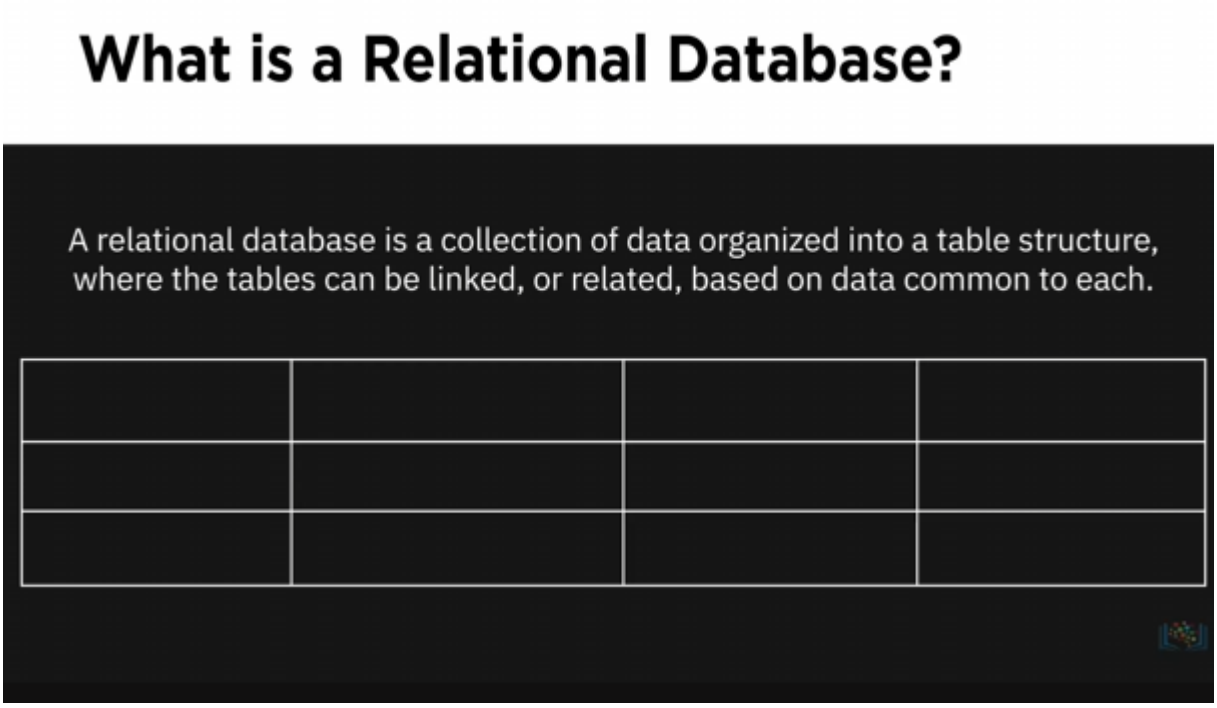


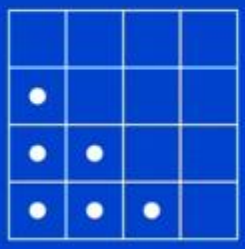
Data Marts and Data Lakes

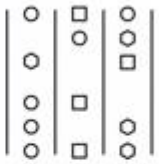
Big Data Stores

Video: RDBMS (7:38)

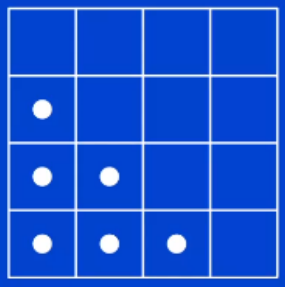
A relational database is a collection of data organised into a table structure, where the tablkes can be lineked, or ralted, based on the data common to each





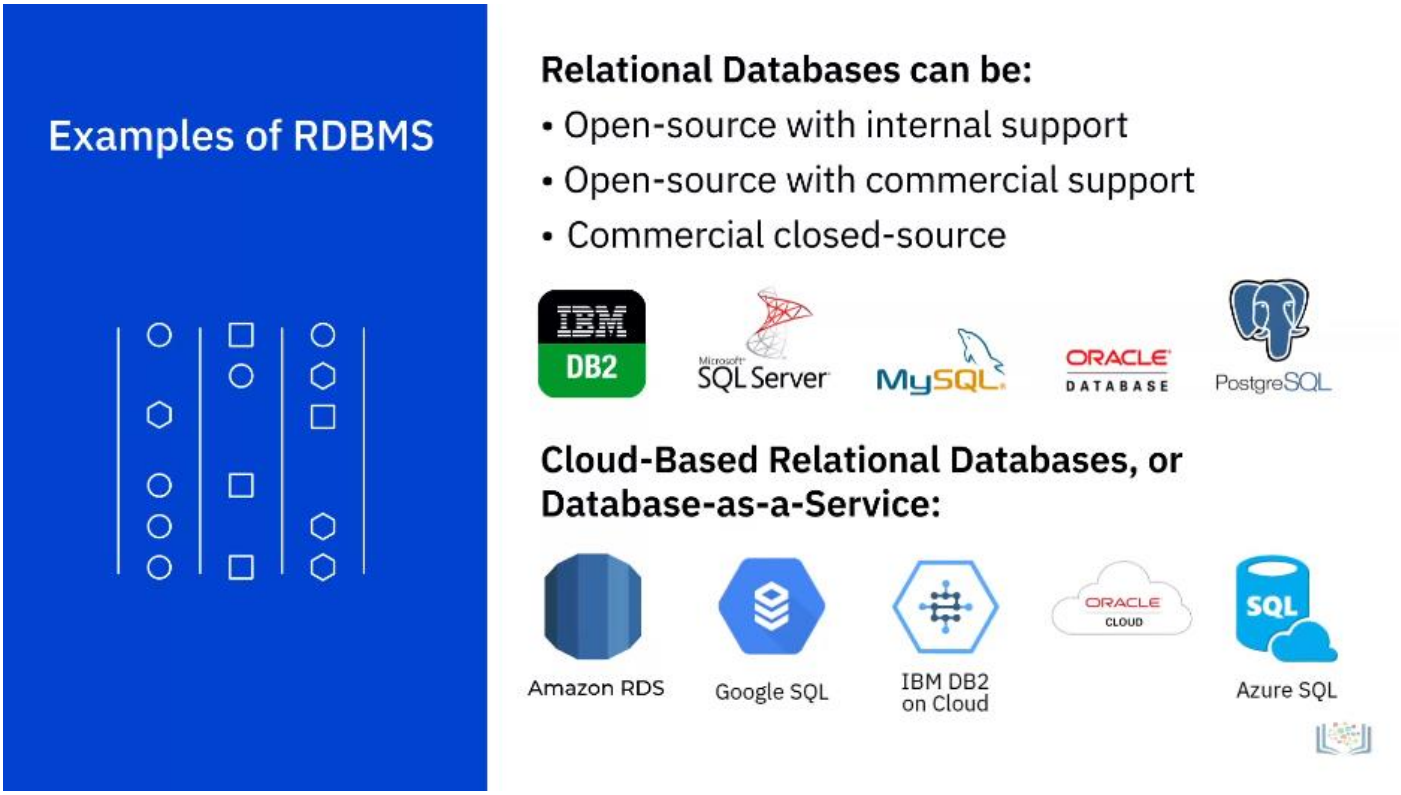


Similarities between relational databases and spreadsheets:
Relational databases build on the organizational principles of flat files such as spreadsheets, with data organized into rows and columns following a well-defined structure and schema.

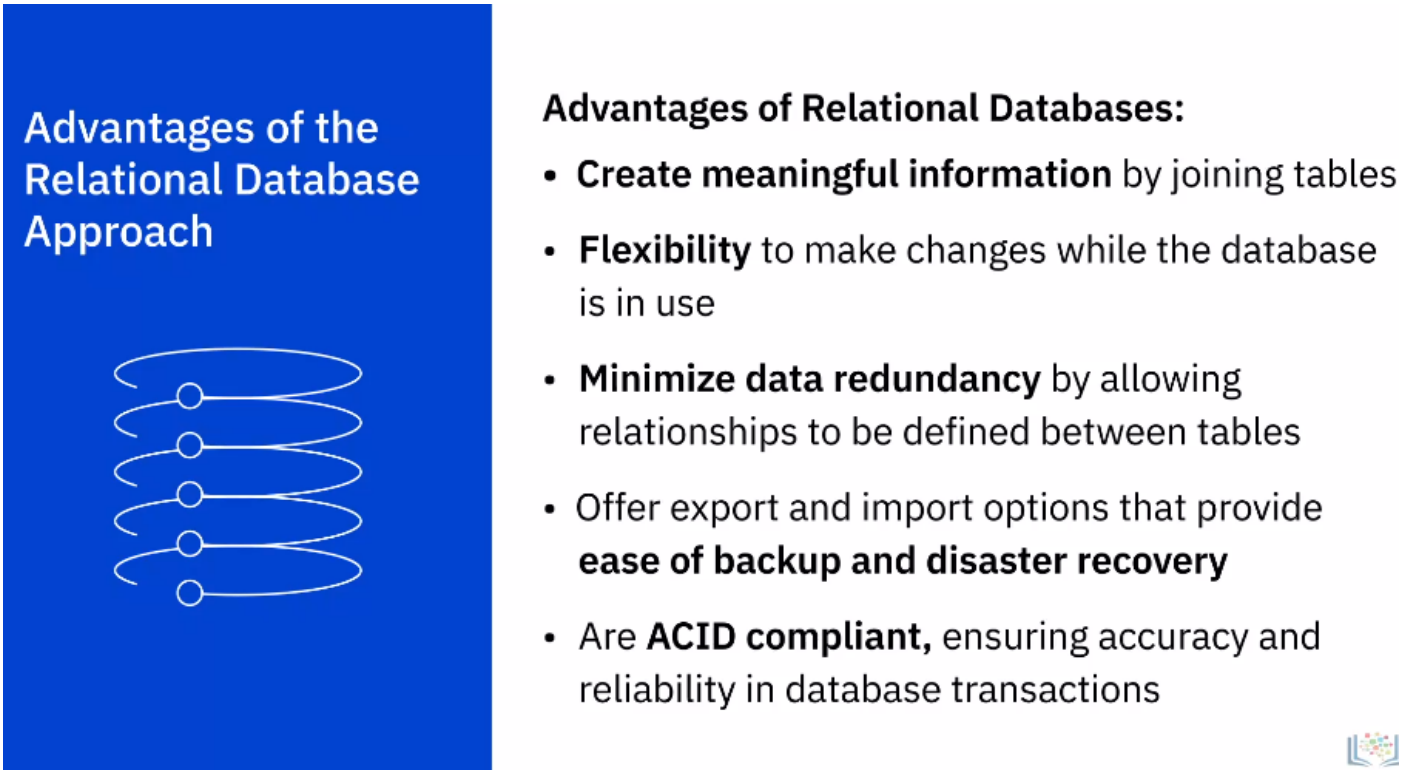


- Ideal for the optimized storage, retrieval, and processing of data for large volumes of data
- Each table has a unique set of rows and columns
- Relationships can be defined between tables
- Fields can be restricted to specific data types and values
- Can retrieve millions of records in seconds using SQL for querying data
- Security architecture of relational databases provides greater access control and governance

Examples of RDBMS



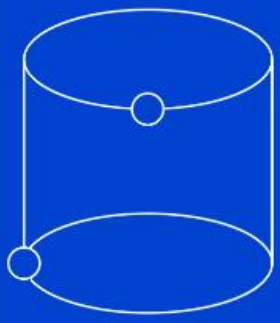
Advantages of Relational Database Approach



1. ACID-compliance: ACIDstands for Atomicity, Consistency, Isolation, and Durability.
2. **And ACID compliance implies that the data in the database remains accurate and consistent**

RDBMS Use Cases

Use Cases for RDBMS



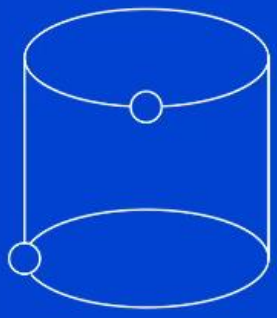
Relational Databases are well suited for:

Online Transaction Processing (OLTP) application

Can support transaction-oriented tasks that run at high rates and

- Accommodate large number of users
- Manage small amounts of data
- Support frequent queries and fast response times

Use Cases for RDBMS



Relational Databases are well suited for:

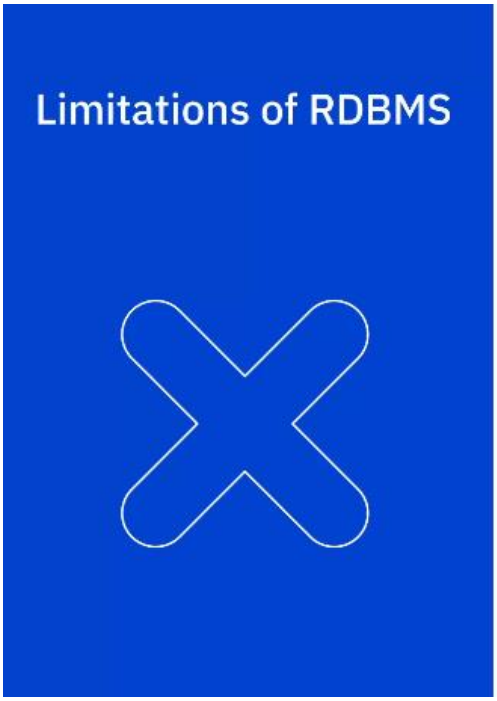
Data Warehouses

Can be optimized for online analytical processing (OLAP)

IoT Solutions

Provide the speed and ability to collect and process data from edge devices

Limitations of RDBMS



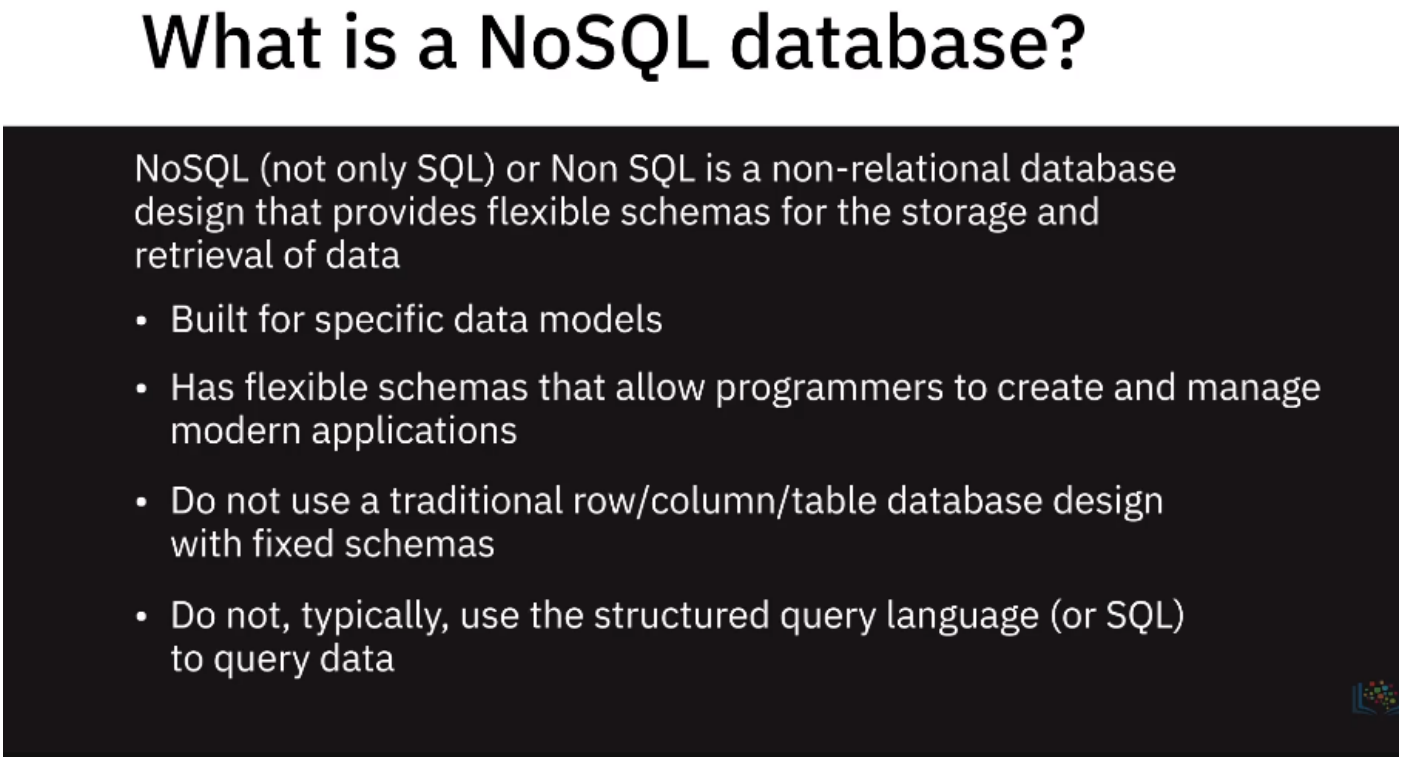
Limitations of RDBMS:

- Does not work well with semi-structured and unstructured data
- Migration between two RDBMS's is possible only when the source and destination tables have identical schemas and data types
- Entering a value greater than the defined length of a data field results in loss of information

Video: NoSQL (7:35)

What is NoSQL database

Not only SQL



Schema-less or Free-form Fashion

Key value store

Key-Value Store

Based on the model being used for storing data, there are four common types of NoSQL databases:

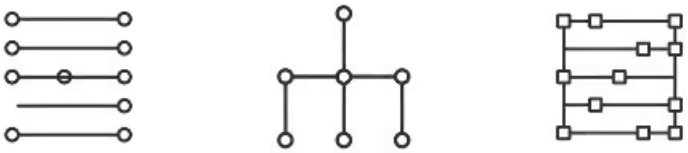
Key-value store:

- Both keys and values can be anything from simple integers or strings to complex JSON documents.
- Great for storing user session data, user preferences, real-time recommendations, targeted advertising, in-memory data caching.

Key-Value Store

Not a great fit if you want to:

- Query data on specific data value
- Need relationships between data values
- Need multiple unique keys



Key-Value Store

Redis Memcached DynamoDB

Document Based

Document-Based

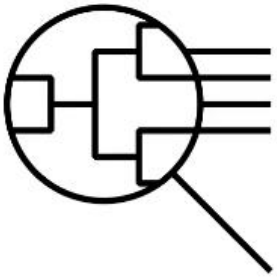
Document-based:

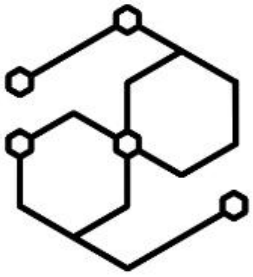
- Document databases store each record and its associated data within a single document.
- They enable flexible indexing, powerful ad hoc queries, and analytics over collections of documents.
- Preferred for eCommerce platforms, medical records storage, CRM platforms, and analytics platforms.

Document-Based

Not a great fit if you want to:

- Run complex search queries
- Perform multi-operation transactions





Document-Based

MongoDB

DocumentDB

CouchDB

Cloudant

Column Based

Column-Based

Column-based:

- Data is stored in cells grouped as columns of data instead of rows.
- A logical grouping of columns is referred to as a column family.

1. For example, a customer's name and profile information will most likely be accessed together
2. but not their purchase history.
3. **So,customer name and profile information data can be grouped into a column family.**
4. Since column databases store all cells corresponding to a column as a continuous disk entry, accessing
5. and searching the data becomes very fast.

Column-Based

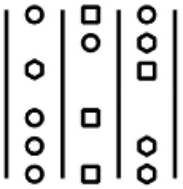
Column-based:

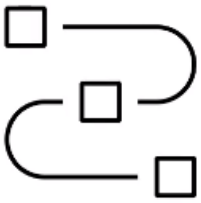
- All cells corresponding to a column are saved as a continuous disk entry, making access and search easier and faster.
- Great for systems that require heavy write requests, storing time-series data, weather data, and IoT data.

Column-Based

Not a great fit if you want to:

- Run complex queries
- Change querying patterns frequently





Column-Based

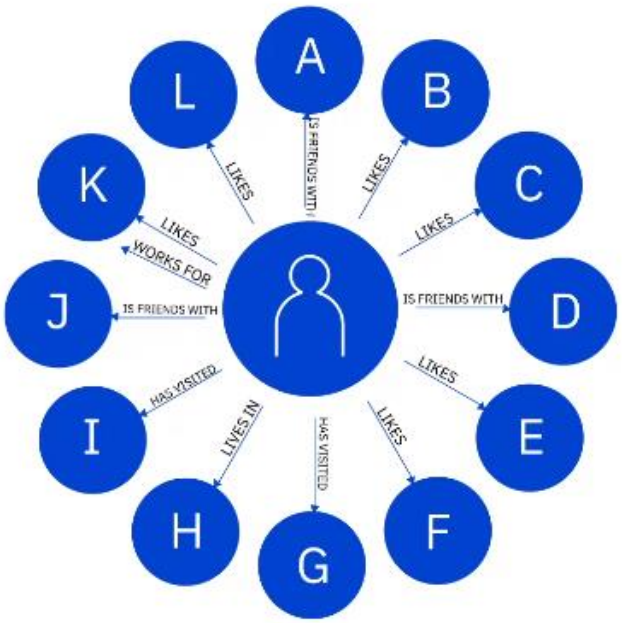
Graph Based

Graph-Based

Graph-based:

- Graph-based databases use a graphical model to represent and store data.
- Useful for visualizing, analyzing, and finding connections between different pieces of data.

Graph-Based



1. The circles arenodes, and they contain the data.
2. The arrows represent relationships.

1. Graph databases are an excellent choice for working with connected data, which is data
2. that contains lots of interconnected relationships.

Graph-Based

Social networks

Product recommendations

Network diagrams

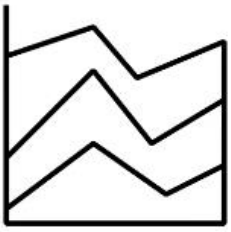
Fraud detection

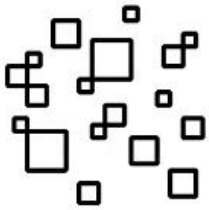
Access management

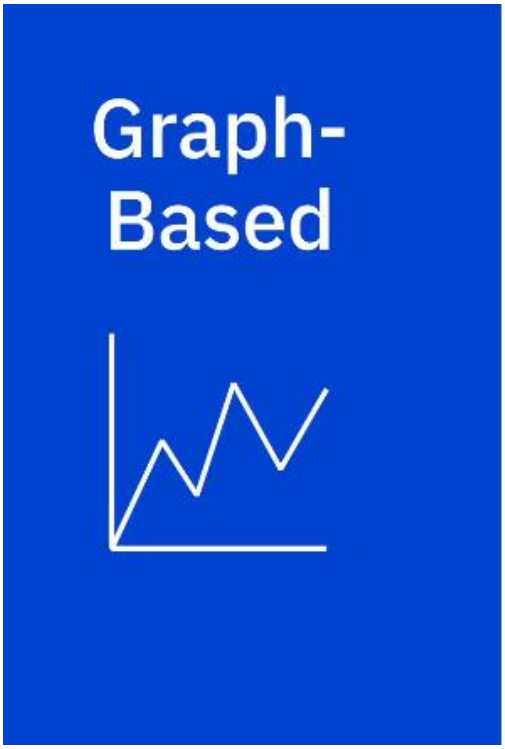
Graph-Based

Not a great fit if you want to:

- Process high volumes of transactions



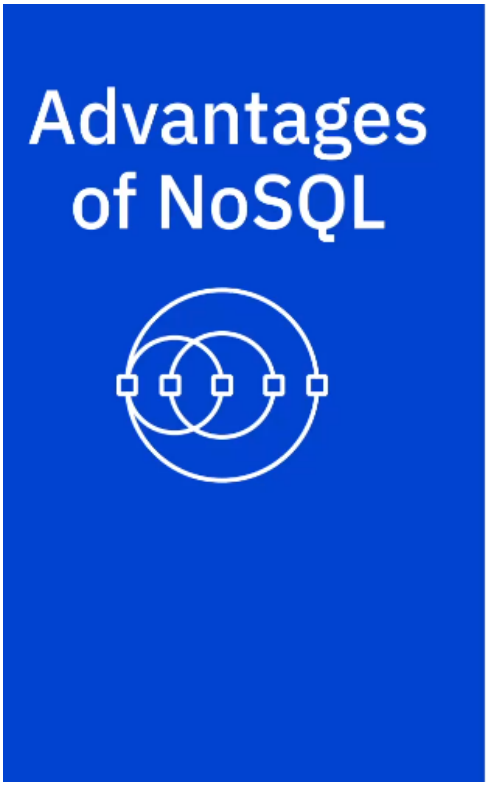




Neo4J

CosmosDB

Advantages of NoSQL



- Its ability to handle large volumes of structured, semi-structured, and unstructured data
- Its ability to run as a distributed system scaled across multiple data centers
- An efficient and cost-effective scale-out architecture that provides additional capacity and performance with the addition of new nodes
- Simpler design, better control over availability, and improved scalability that makes it agile, flexible, and support quick iterations

Key Differences between Relational and Non-relational Databases

Key differences

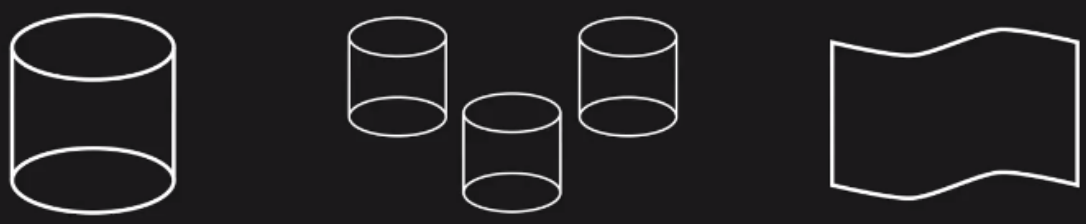
Relational databases	Non-Relational databases
• RDBMS schemas rigidly define how all data inserted into the database must be typed and composed • Maintaining high-end, commercial relational database management systems can be expensive • Support ACID-compliance, which ensures reliability of transactions and crash recovery • A mature and well-documented technology, which means the risks are more or less perceivable	• NoSQL databases can be schema-agnostic, allowing unstructured and semi-structured data to be stored and manipulated • Specifically designed for low-cost commodity hardware • Most NoSQL databases are not ACID compliant • A relatively newer technology

Video: Data Warehouses, Data Marts, and Data Lakes (7:16)

Introduction

Data Mining Repositories store data for:

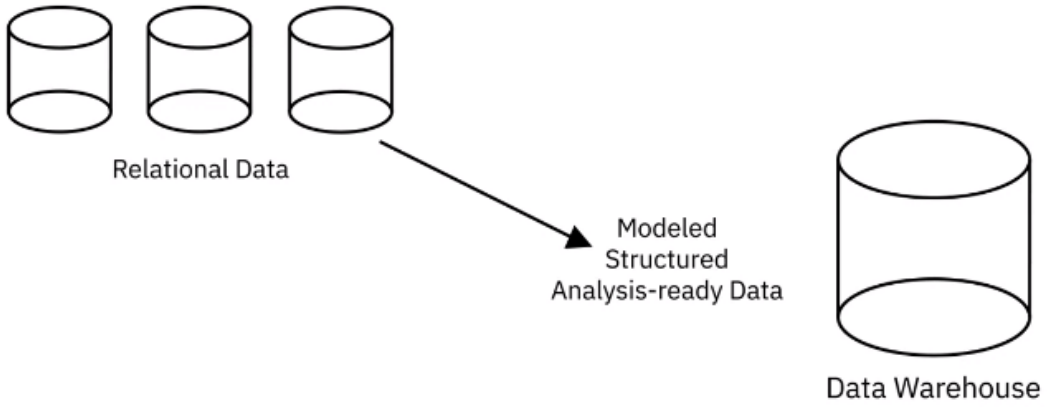
- Reporting
- Analysis
- Deriving insights



Data Warehouses | Data Marts | Data Lakes

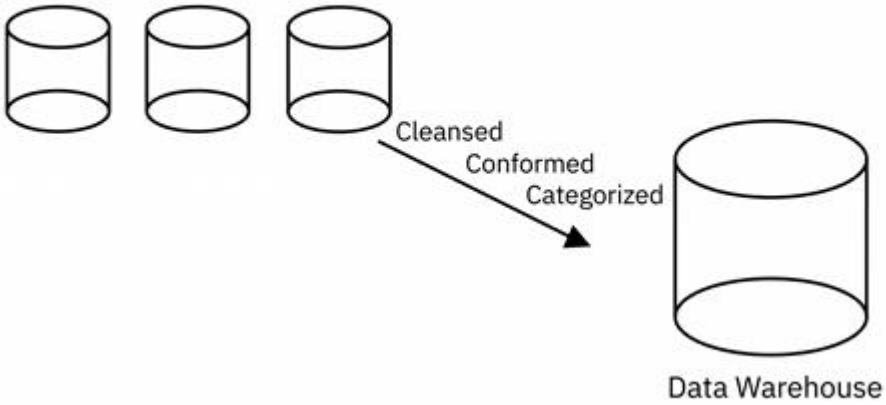
Data Warehouses

Data Warehouses



- **Relational data from transactional systems and operational databases**

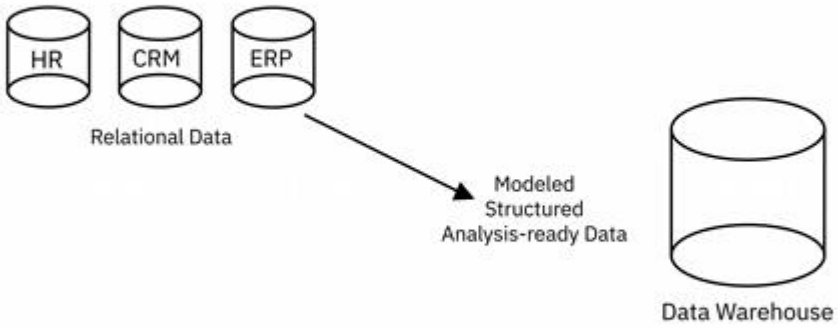
Data Warehouses



1. When data gets loaded into the data warehouse, it is already modeled and structured for a
2. specific purpose, meaning it's analysis-ready.

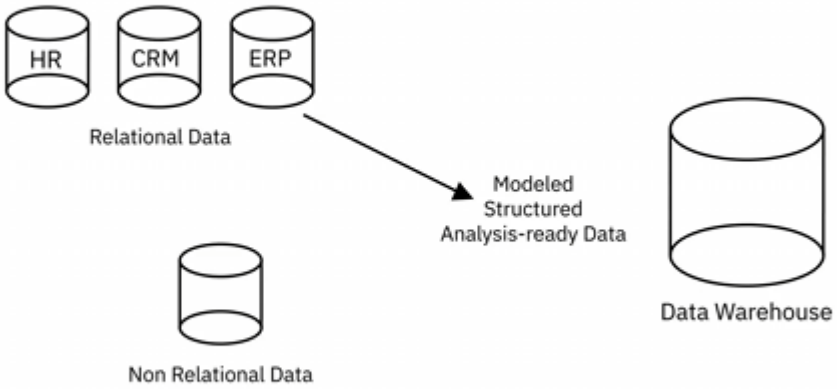
Examples:

Data Warehouses



- Relational data from transactional systems and operational databases

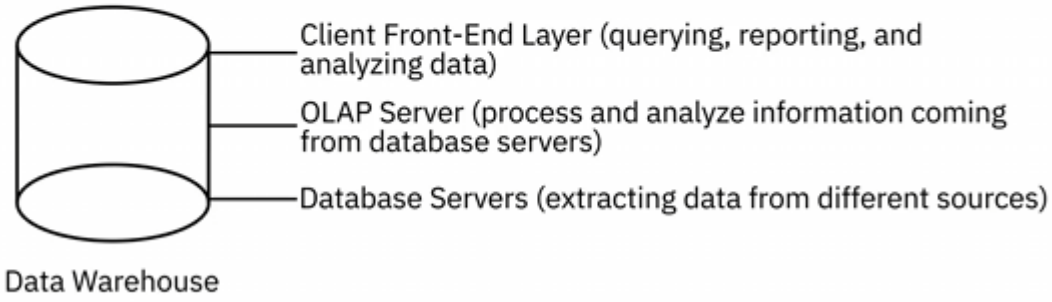
Data Warehouses



- Relational data from transactional systems and operational databases
- Non-relational data

Data Warehouses

A Data Warehouse has a 3-tier architecture:



Data Warehouses

Benefits of cloud-based data warehouses:

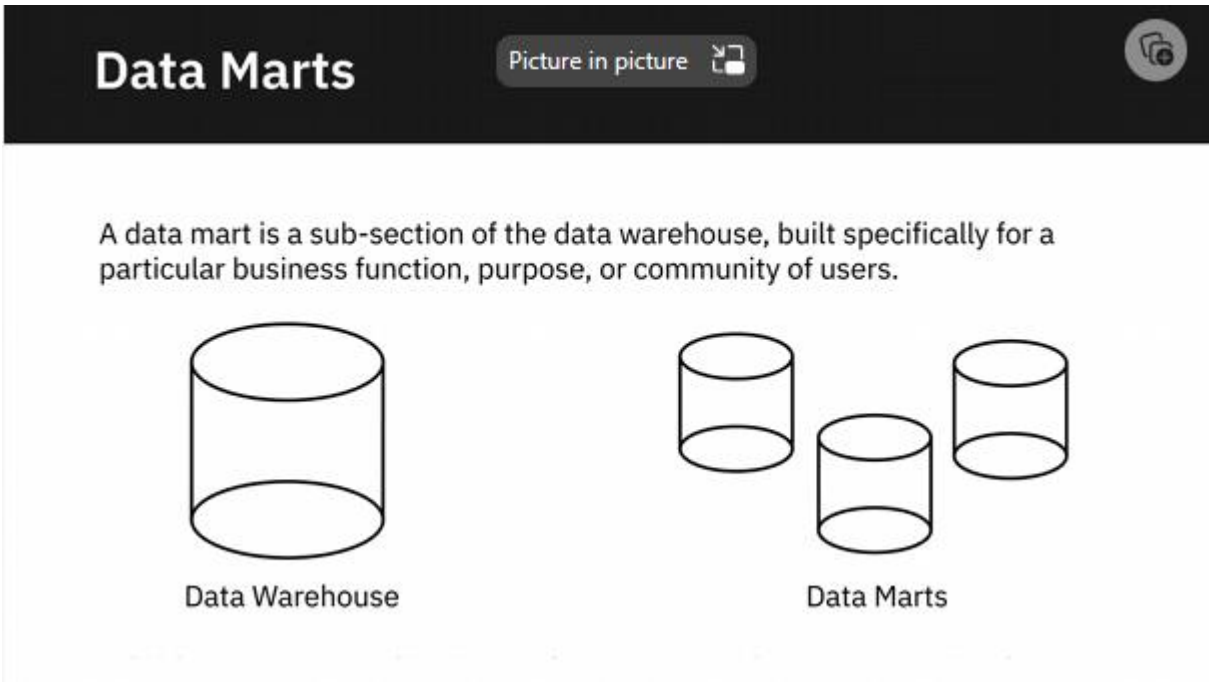
- Lower costs
- Limitless storage and compute capabilities
- Scale on a pay-as-you-go basis
- Faster disaster recovery

Data Warehouses

teradata. ORACLE EXADATA IBM Db2 NETEZZA

amazon REDSHIFT Google BigQuery cloudera snowflake

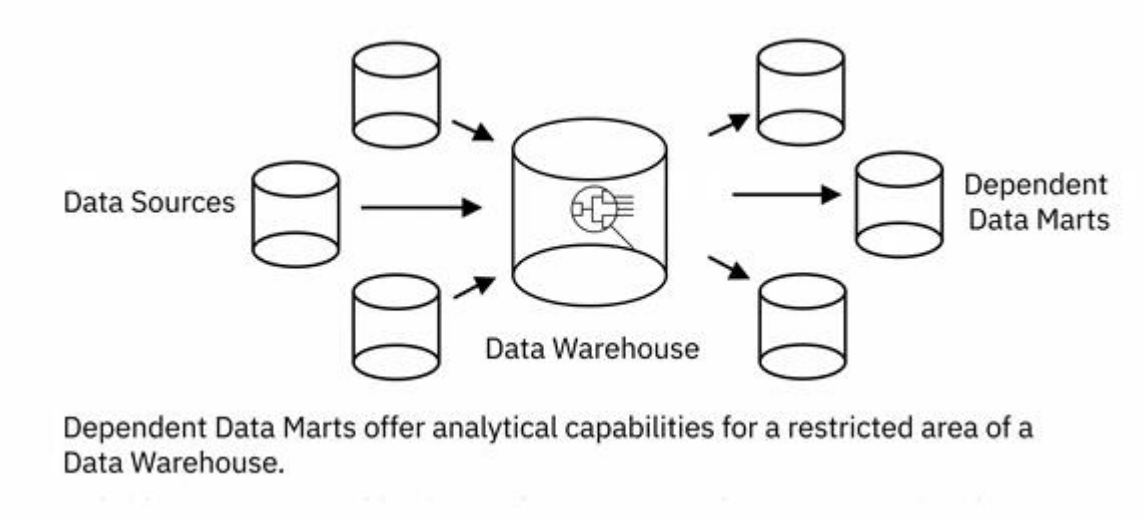
Data Marts



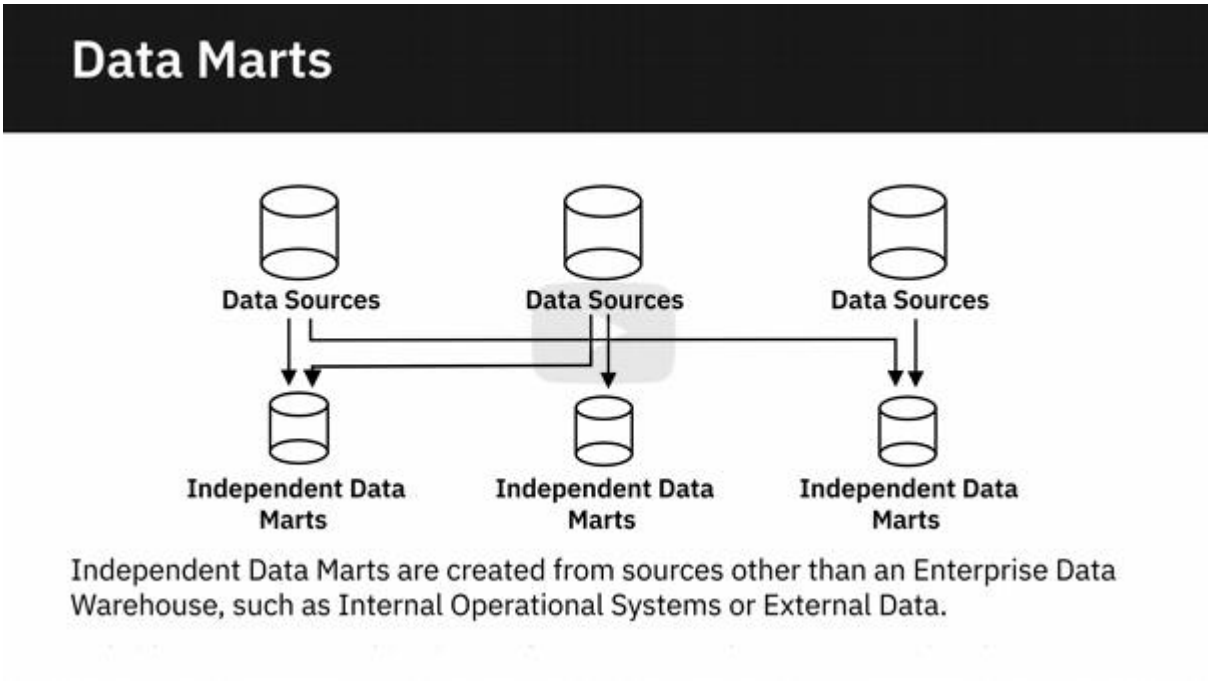
Three types of data marts:

- Dependent
- Independent
- Hybrid

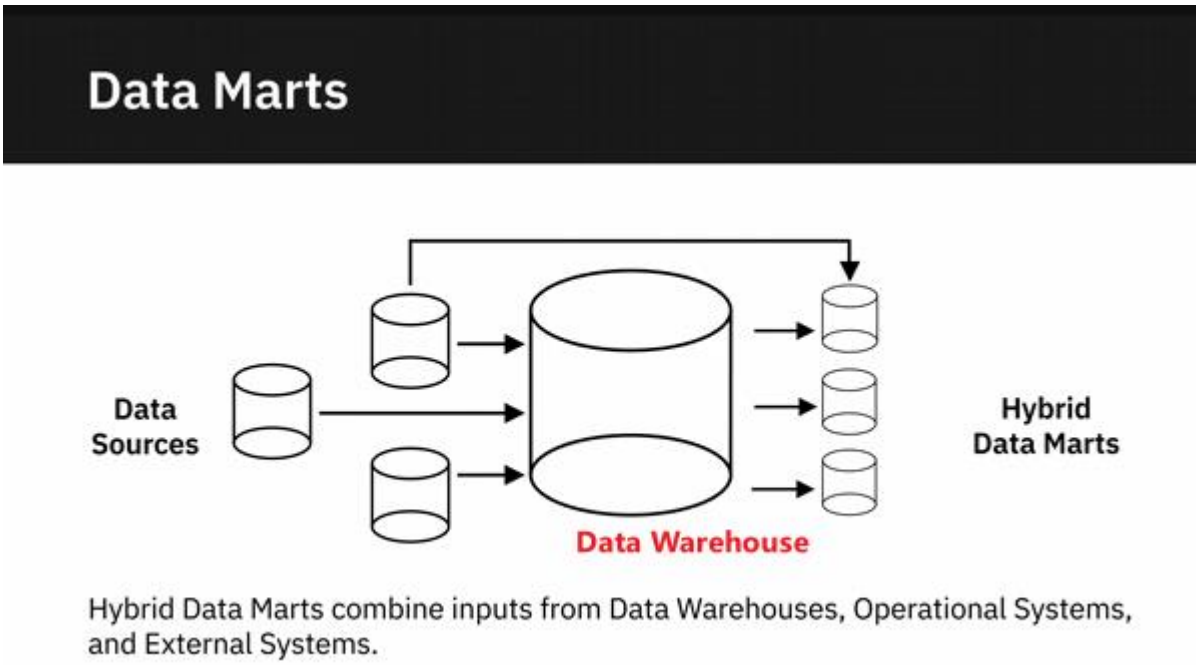
Dependent Data Mart



Independent Data Mart

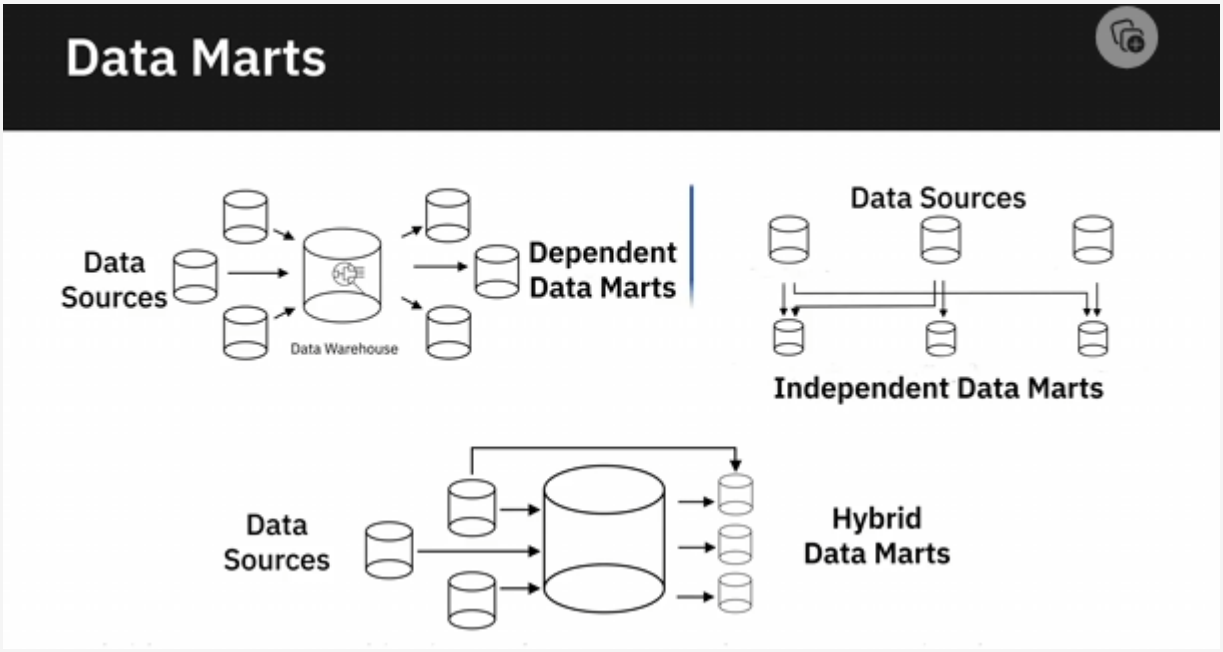


Hybrid Data Mart

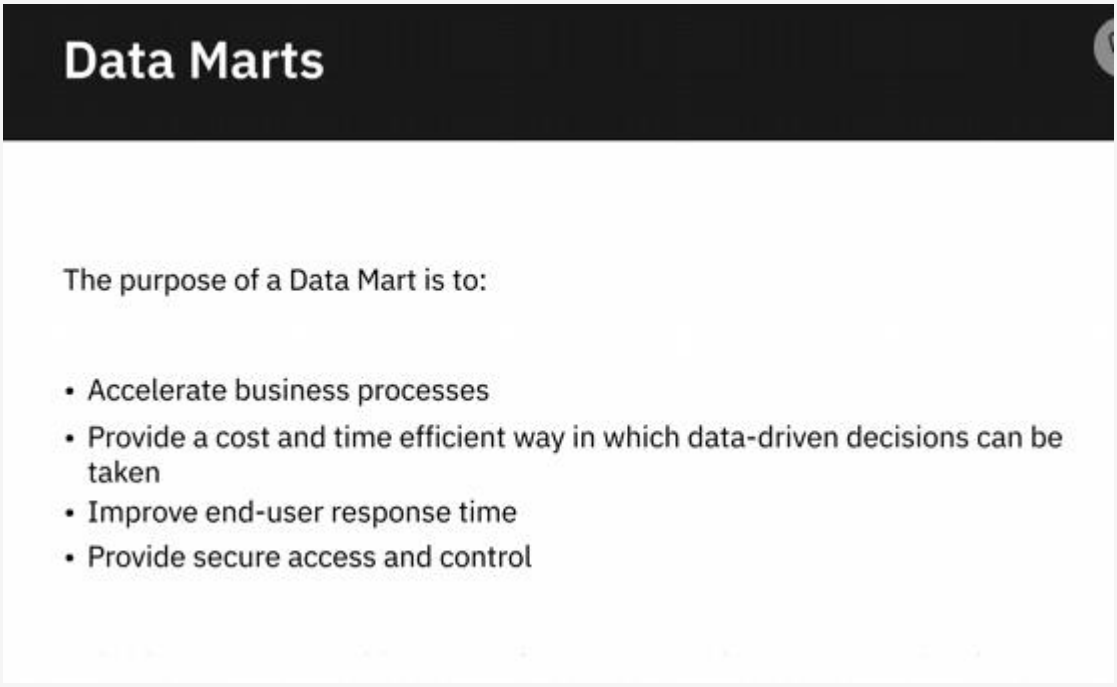


1. **The difference also lies in how data is extracted from the source systems, the transformations**
2. that need to be applied, and how the data is transported into the mart.
3. Dependent data marts, for example, pull data from an enterprise data warehouse, where data
4. has already been cleaned and transformed.
5. Independent data marts need to carry out the transformation process on the source data
6. since it is coming directly from operational systems and external sources.
7. Whatever the type, the purpose of a data mart is to:
8. provide users' data that is most relevant to them when they need it,
9. accelerate business processes by providing efficient response times,
10. provide a cost and time-efficient way in which data-driven decisions can be taken,
11. improve end-user response time; and

12. provide secure access and control.

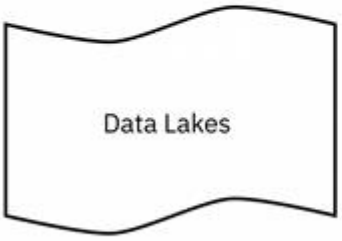


Purpose of Data Mart



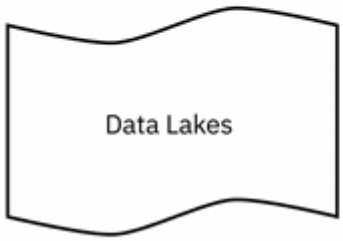
Data Lake

Data Lakes



- Store large amounts of structured, semi-structured, and unstructured data in their native format
- Data can be loaded without defining the structure and schema of data
- Exist as a repository of raw data straight from the source, to be transformed based on the use case
- Data is classified, protected, and governed

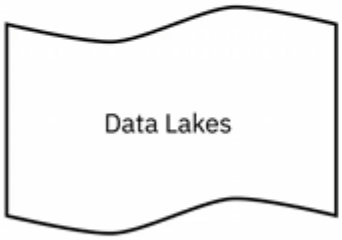
Data Lakes



- A reference architecture that combines multiple technologies
- Can be deployed using
  - > Cloud Object Storage, such as Amazon S3
  - > Large-scale distributed systems such as Apache Hadoop
  - > Relational Database Management Systems, as well as NoSQL data repositories

Benefits

Data Lakes



Benefits:

- Ability to store all types of data (unstructured, semi-structured and structured data)
- Agility to scale based on storage capacity (growing from terabytes to petabytes)
- Saving time in defining structures, schemas, and transformations (data is imported in its original format)
- Ability to repurpose data in several different ways and wide-ranging use cases

Data Lakes

amazon.com cloudera Google IBM

Informatica Microsoft ORACLE EXADATA SAS

snowflake teradata. zaloni

Video: (Optional) Data Lakehouses Explained

Enterprise Data Warehouses

Data Lake: cost-effective way to store tons of data

But without governance and quality assurance, data lake becomes “data swamp”

Not built to perform complex data analysis.

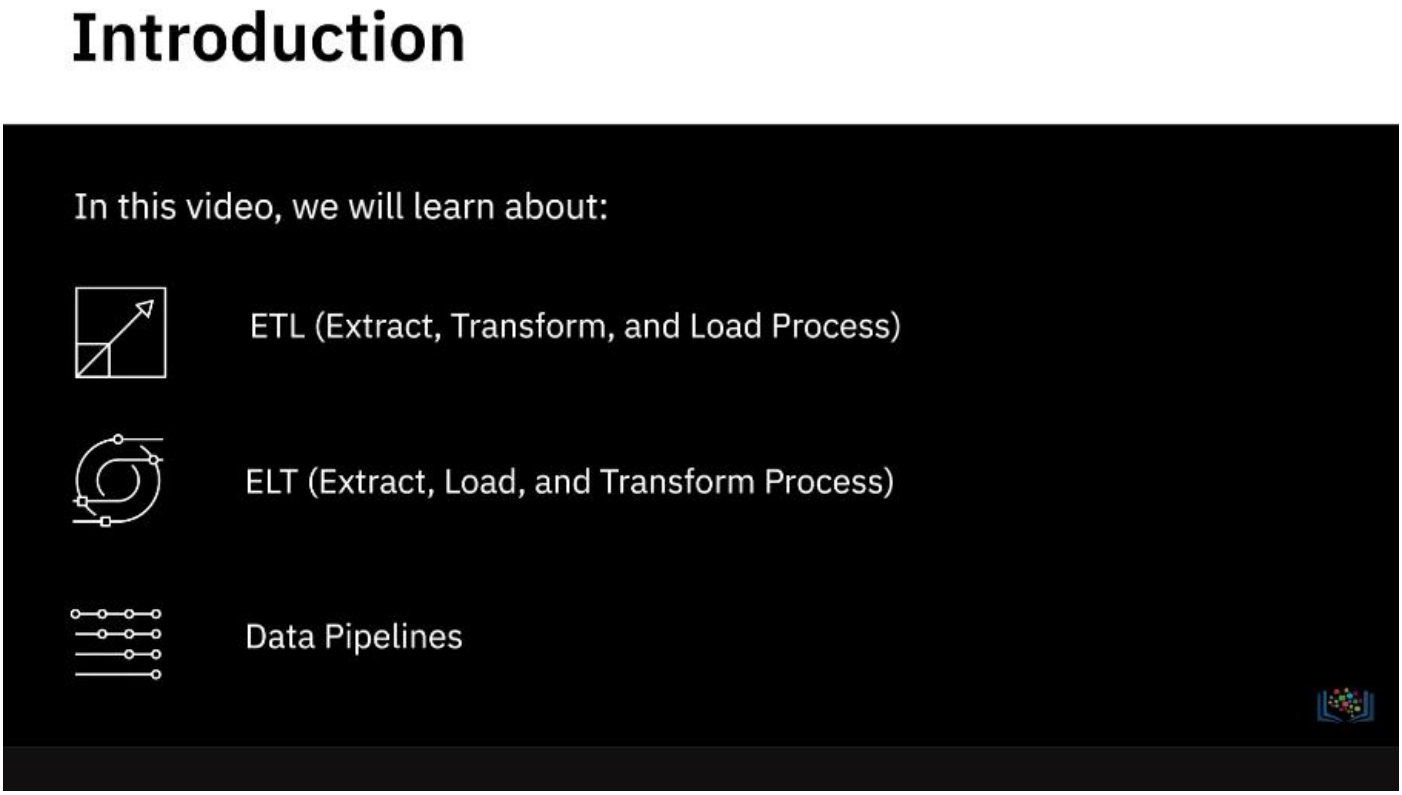
Good at data query,

Data Lakehouse: flexibility + cost effectiveness of data lake

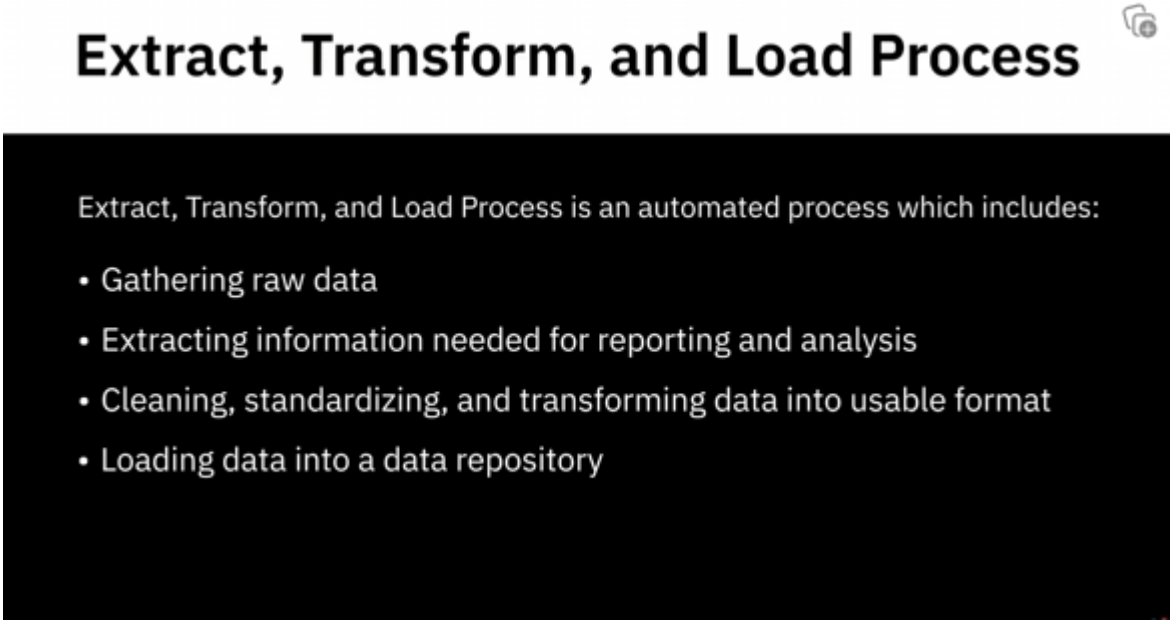
+ We get performance and structure/government from a data warehouse.

Video: Viewpoints: Considerations for Choice of Data Repository (6:25)

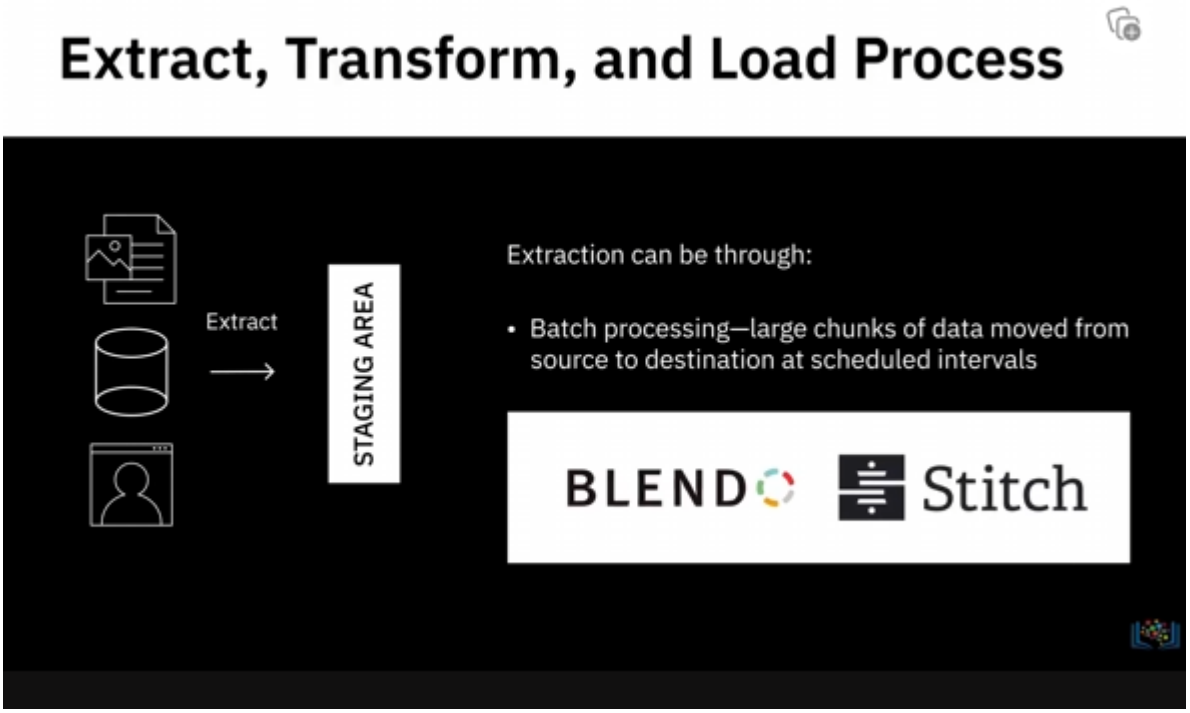
Video: ETL, ELT, and Data Pipelines (6:37)

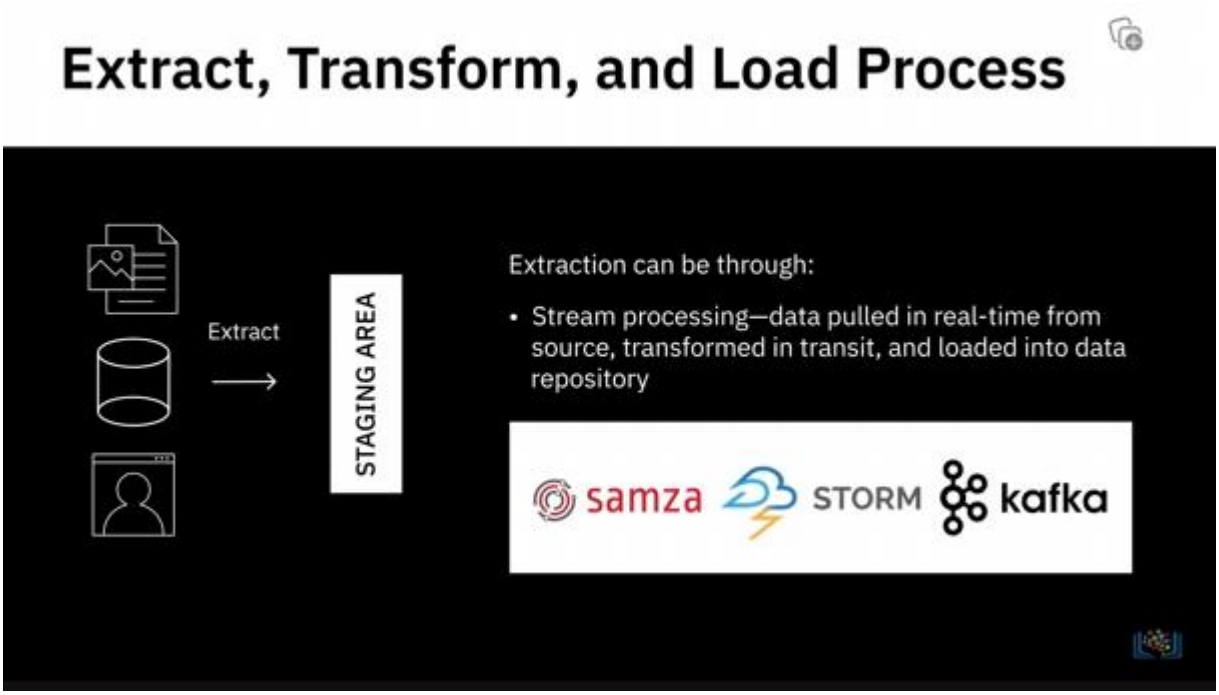


Extract transform and load process (ETL)

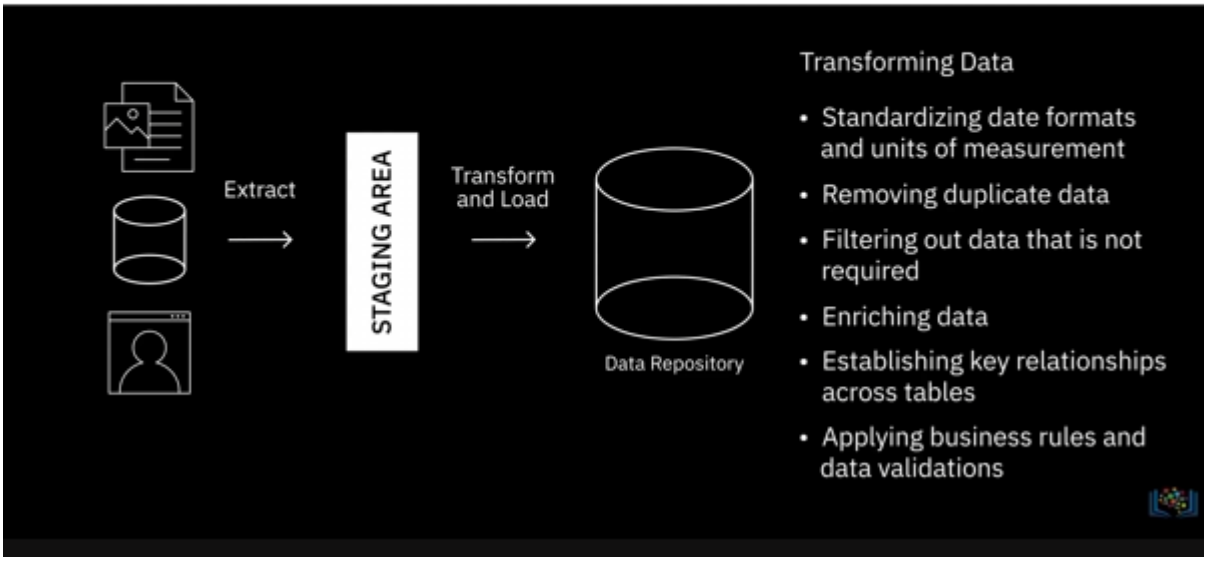


Extract

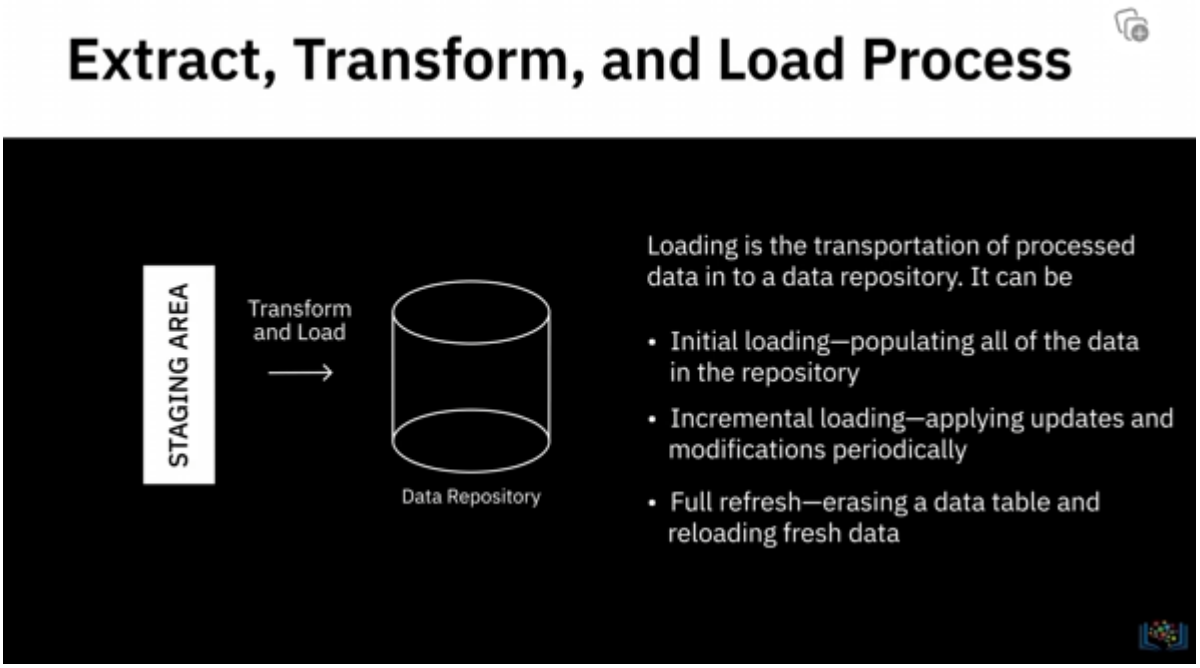

Extract, Transform, and Load Process
Extract
STAGING AREA
Extraction can be through:
• Batch processing—large chunks of data moved from source to destination at scheduled intervals
BLENDO
Stitch


Extract, Transform, and Load Process
Extract
STAGING AREA
Extraction can be through:
• Stream processing—data pulled in real-time from source, transformed in transit, and loaded into data repository
samza
STORM
kafka

Transform

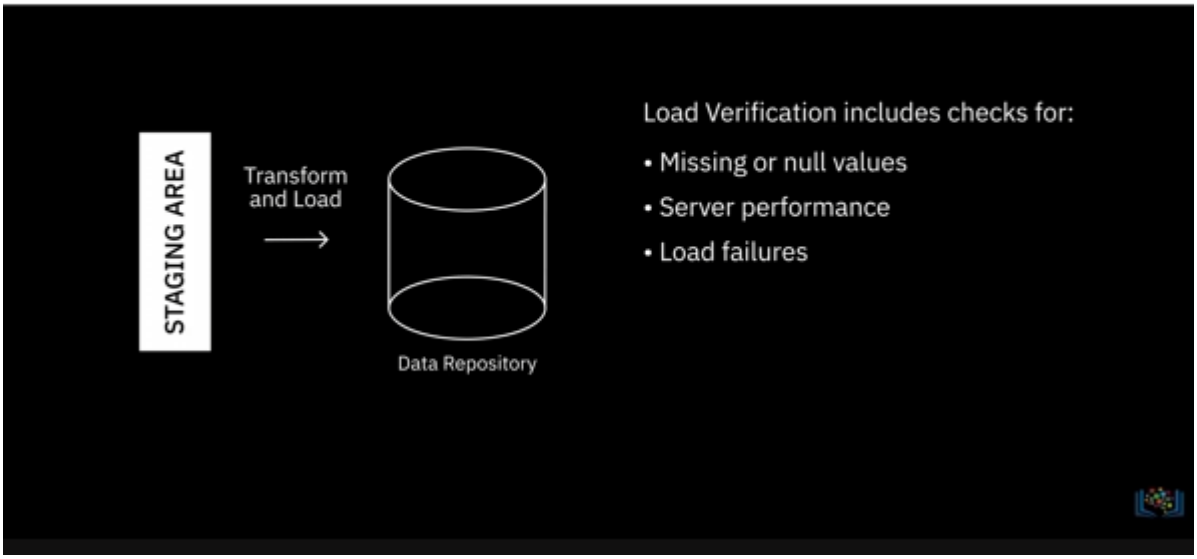


Load

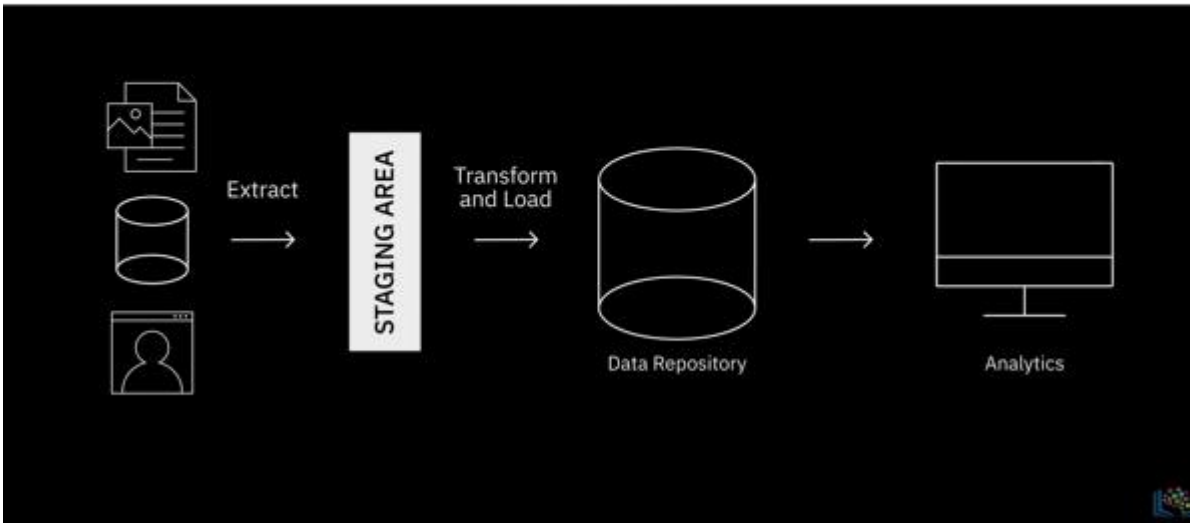


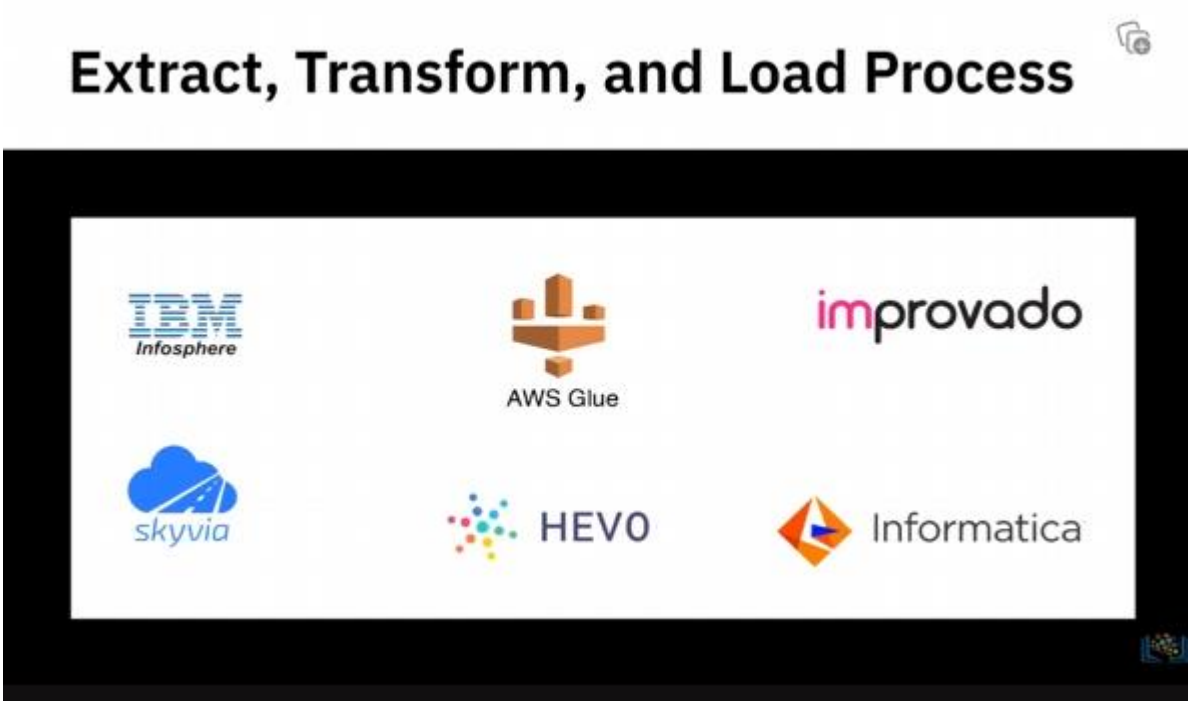
Load Verification

Extract, Transform, and Load Process

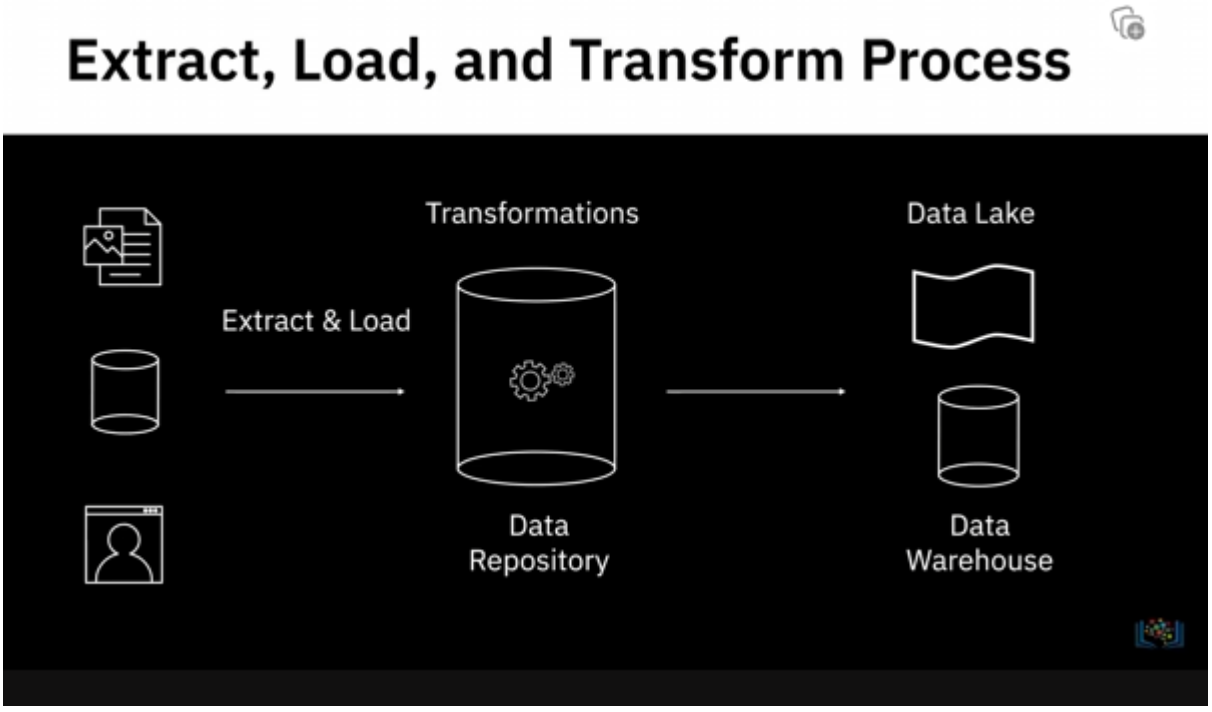


Extract, Transform, and Load Process



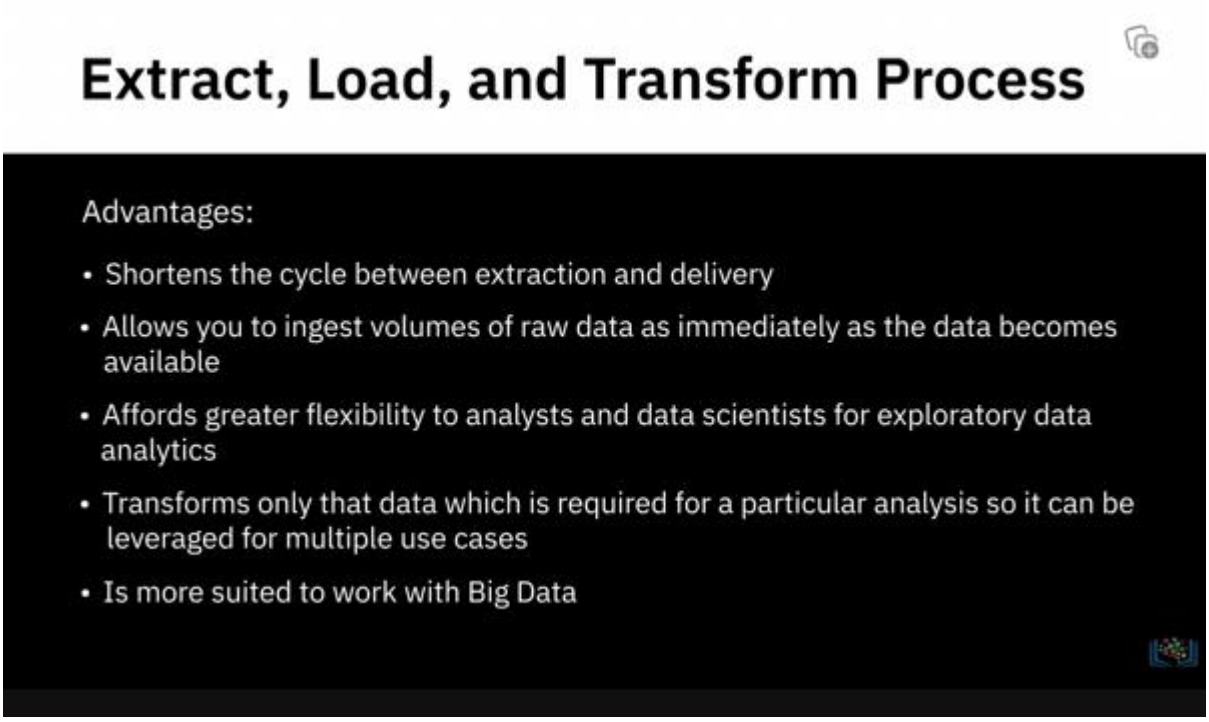


Extract Load and Transform

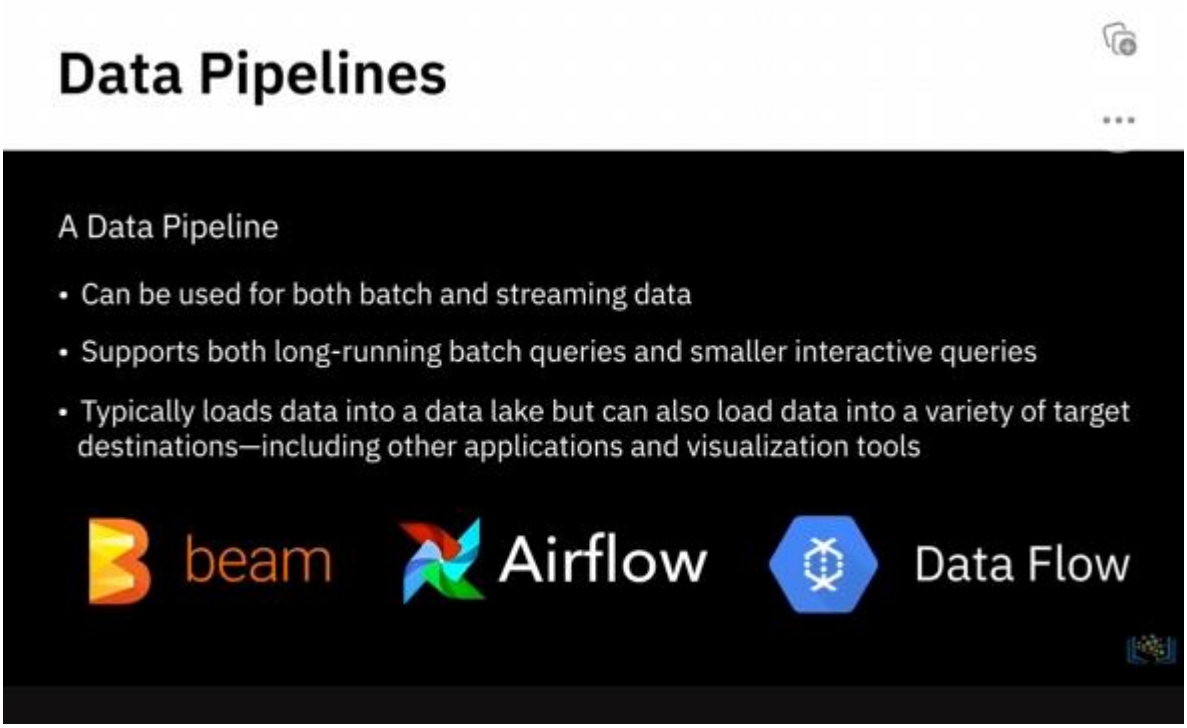


Relatively new

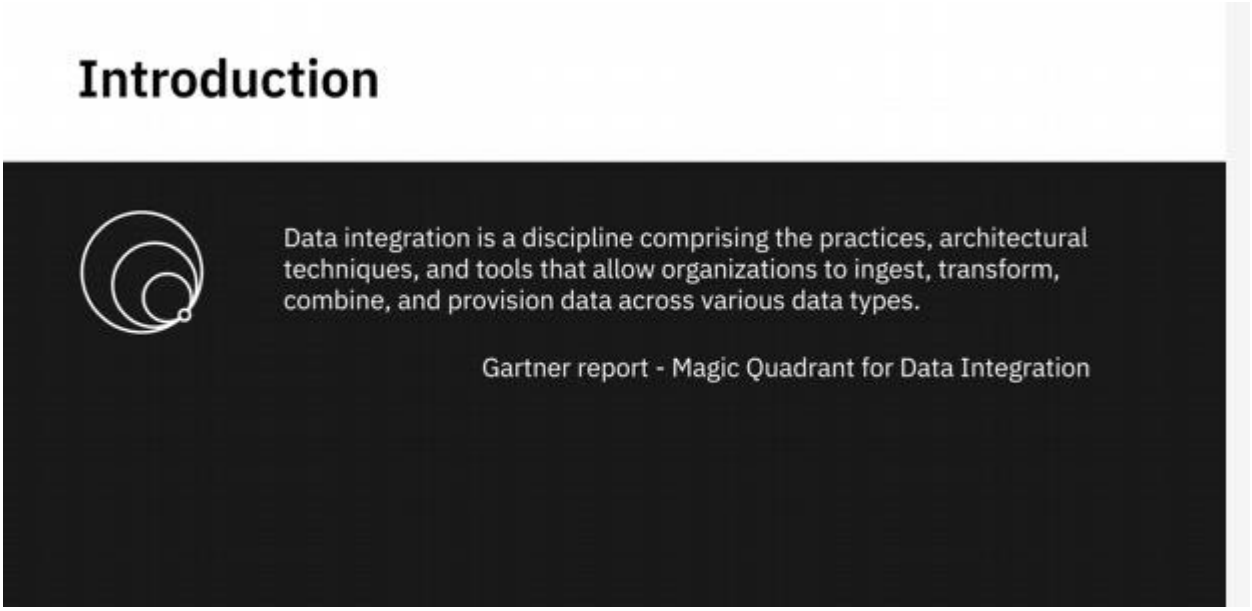
Advantages



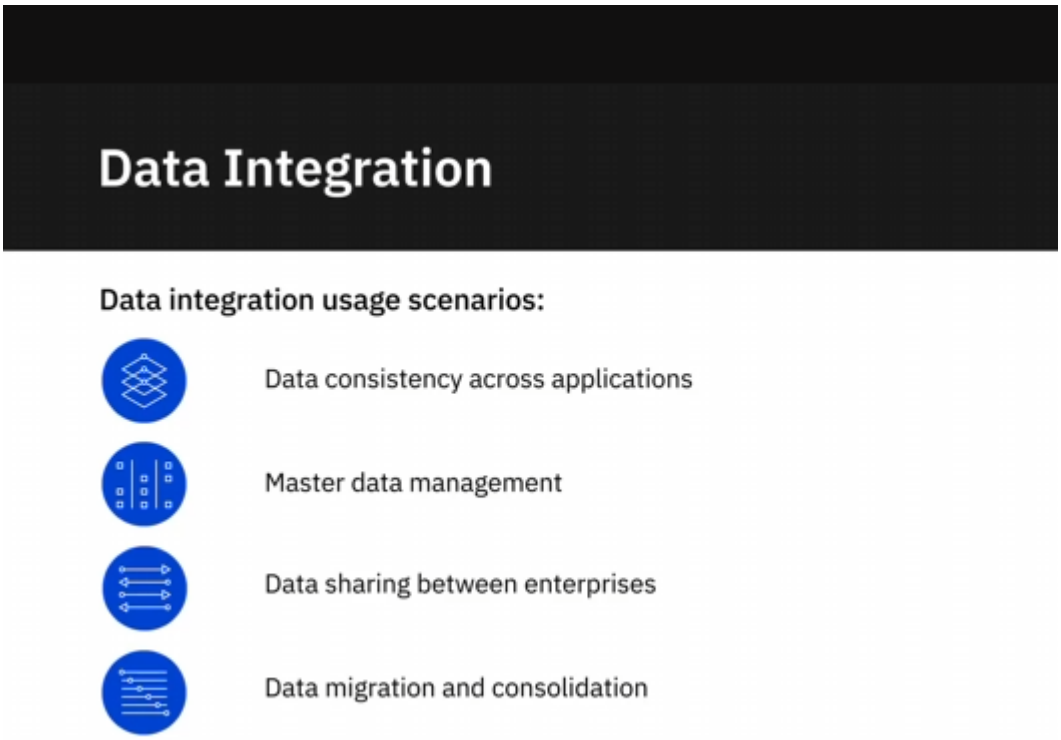
Data Pipelines



Video: Data Integration Platforms (4:48)



Usage Scenarios



Data Integration Inclusions

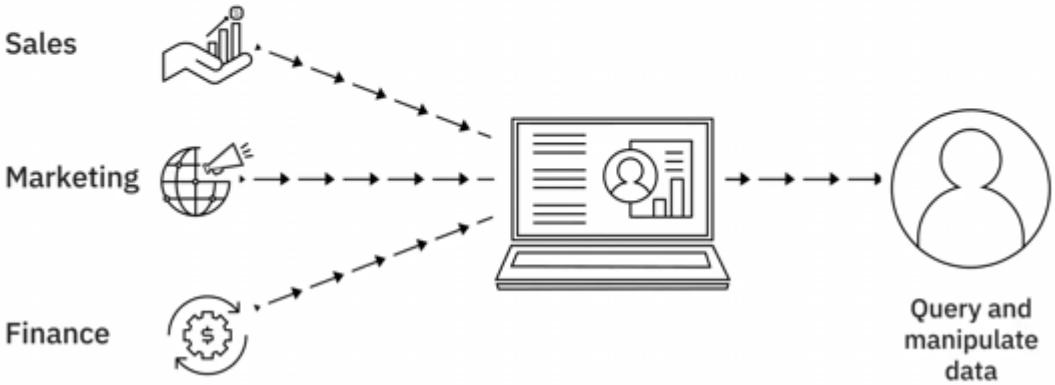
Data Integration

Data Integration includes:

- Accessing, queueing, or extracting data from operational systems
- Transforming and merging extracted data either logically or physically
- Data quality and governance
- Delivering data through an integrated approach for analytics purposes

Data Integration

To make customer data available for analytics:



Capabilities of Data Integration Platform

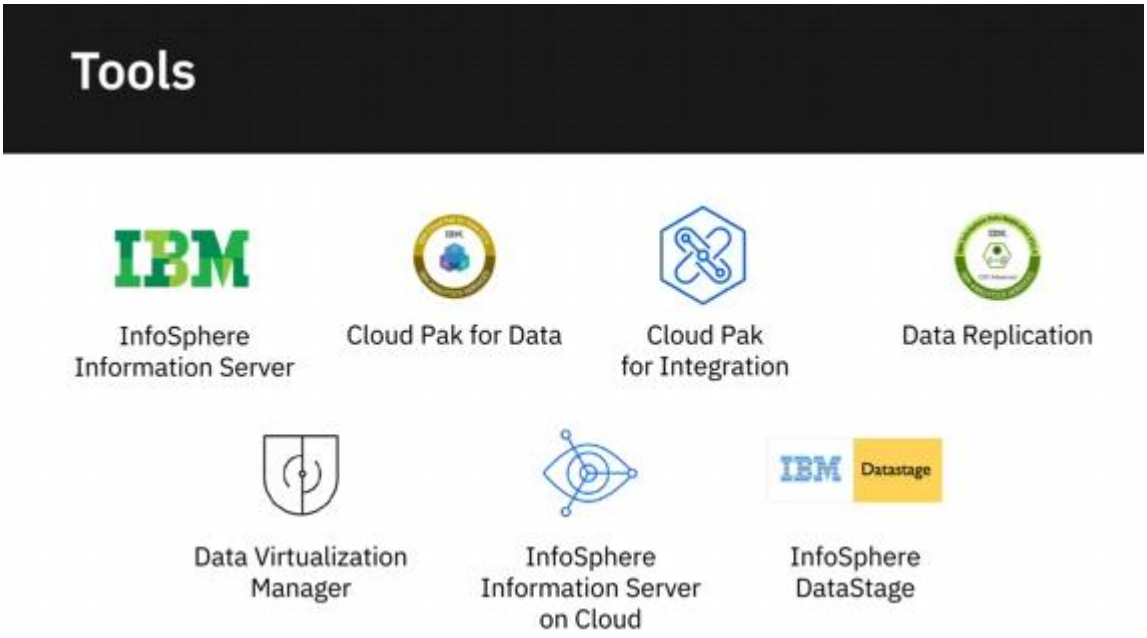
Capabilities of a Data Integration Platform

Capabilities of a modern data integration platform:

- Pre-built connectors and adapters
- Open-source architecture
- Optimization for both batch processing of large-scale data and continuous data streams, or both
- Integration with Big Data sources
- Additional functionalities for data quality and governance, compliance, and security
- Portability between on-premise and different types of cloud environments

Data Integration Tools

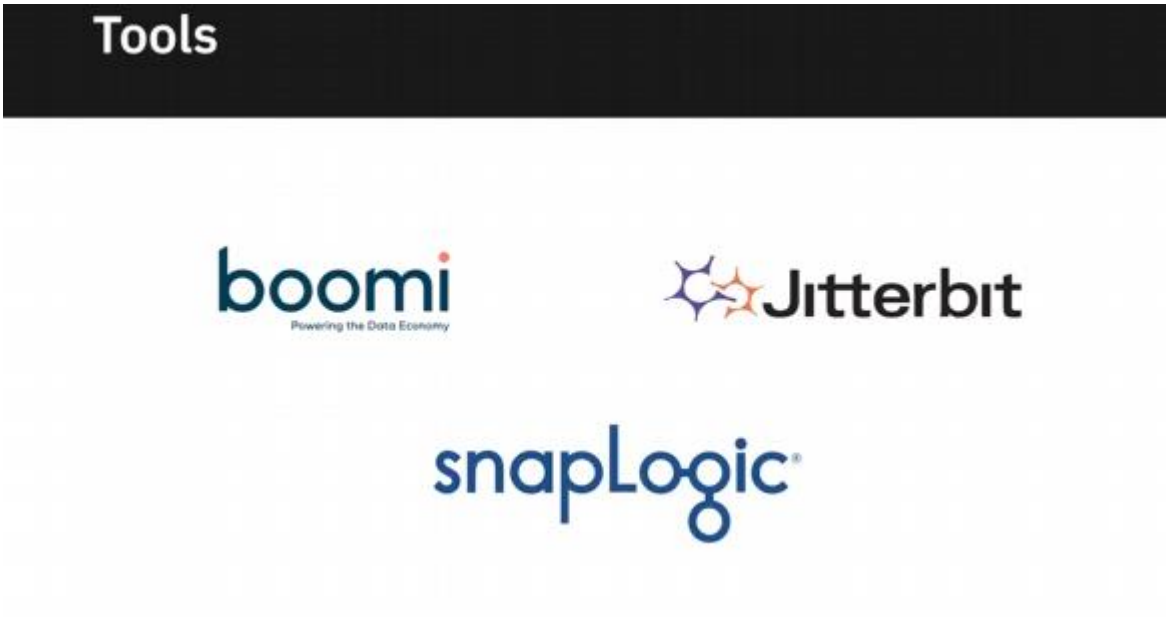
Tools

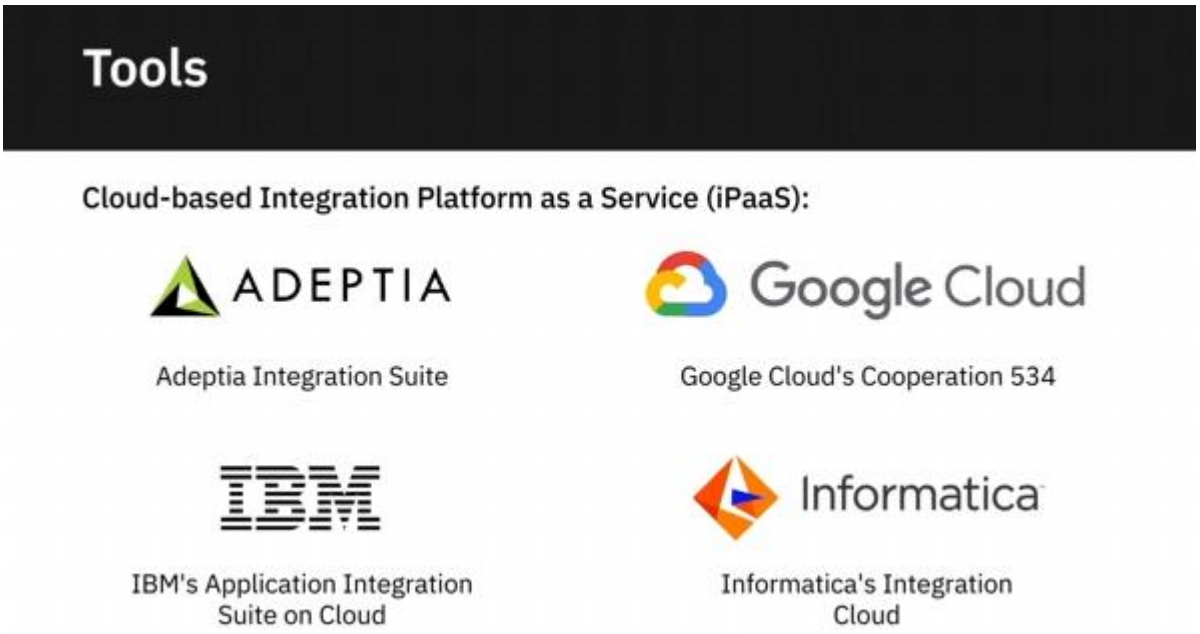


Tools

- Talend Data Fabric
- Talend Cloud
- Talend Data Catalog
- Talend Data Management
- Talend Big Data
- Talend Data Services
- Talend Open Studio

Tools





Video: Viewpoints: Tools, Databases, and Data Repositories of Choice (6:38)

In this video we will listen to data professionals talk about some of the data engineering tools, databases, and data repositories they work with.

The ecosystem of data engineering is

very very vast.
We work especially with, mostly with,
open source tools. So, we work with
RDBMS databases like MySQL. We work with
NoSQL databases like MongoDB, Cassandra,
graph databases like Neo4j.
Python is pretty much indispensable.
So, whatever feature we feel missing in
our data engineering space,
we first try to, you know,
use Python to
get that into a place. And then over a
period of time we see if any product can
take its place. We also work with
tools like
Apache Airflow to create
data pipelines. We work with tools like
Spark for big data processing.
We work with
Kafka to handle streaming data.
We work with Talend for
ETL functionality. We also work with
tools like
Beautiful Soup and Scrappy for
web scraping. And we also look at
a variety of cloud storages for our
archival and then for our daily storage
purposes. Before Coursera I was
in a non-profit where SQL server was the
data repository and we use a SQL server
integration service, as well as Okada
to do
data integration. And in Coursera
AWA's worksheet is our data warehouse.

And we use
AWS A3 as our data lake.
We have internal tool to do...
to build up,
to build our ETL pipeline, and schedule
our ETL pipeline but now we are
moving to use an open source tool
called
Apache Airflow to do
data orchestration. I work with
relational databases such as IBM DB2,
PostgreSQL, and Microsoft SQL Server.
And I also have exposure to working on
NoSQL database such as
Cassandra and MongoDB. And I worked on
streaming technologies
for replication such as WebSphere MQ,
and also for back-end processing
by moving the transactional data
by using the Kafka queues onto the
back office databases.
At the same time, I also worked on
data engineering tools, or
data movement tools,
for moving data from one data source to
another. For example, I worked on
SSIS by building these data movement
packages.
At the same time I also worked on this
cool tool called
NiFi. I think it's a... this NiFi
is now maintained by
Apache Foundation
which is an open source foundation.

Which I highly recommend looking at
some of their projects
because they're all
open source,
the code is open. You can
go through the source code and learn
a lot about some of these products and
even projects. And you can,
if you're an enthusiast, can have a lot of
time on your hands you can even
contribute
to some of these projects.
I digress but
NiFi can be used to
move data between heterogeneous
data sources.
And I also developed my own
scripts such as Shell, and Perl, and
written Java APIs to move
data between different applications
and vendors.
So, most people, myself included,
who've done work in data engineering
have had to work with a number of
databases and tools.
Having spent a good part of my data
career at IBM,
I had to do a lot of work with the IBM
DB2 database
and later became a product manager
for it.
But I've also had to work with other
databases like MySQL
and PostgreSQL. And then as

part of my job evolved, I had to pick up
skills in
Big Data Systems like Hadoop and Spark.
So, it's important that as a data
engineering, a field that continues to
evolve, you need to, you know,
come up with... you need to become
a lifelong learner
and keep picking up skills that are
going to be required
for your job and the problems that
you're trying to solve.
So, as long as you're good with
data fundamentals,
you should be able to quickly pick up
these new skills and technologies.
The primary product that I've worked
with my entire career
is IBM's DB2 on mid-range platforms, so,
on Linux, Unix and Windows.
To me this is a great choice in the
relational database space.
I also work very heavily with
AWS products.
I've been working with databases on RDS,
Microsoft SQL Server. We've done
a little bit of MariaDB there. We also do
MariaDB outside of that
on our own servers. Those are the primary
relational database products that
we work with. There are a ton of other
tools we work with in order to properly
manage those
or to manage workflows around those.

Of course Github or Git.

You have to be familiar with those.

Those are are integral to

any process in a DevOps organization or

in most non-DevOps organizations at this

point. Using Jenkins... we use

Jenkins for a lot of things; to spin up

containers, to do maintenance jobs, to

manage code deploys. There's a

lot of important stuff that

that we do with Jenkins as well.

we use Liquibase.

Liquibase is actually excellent.

We use it with containers

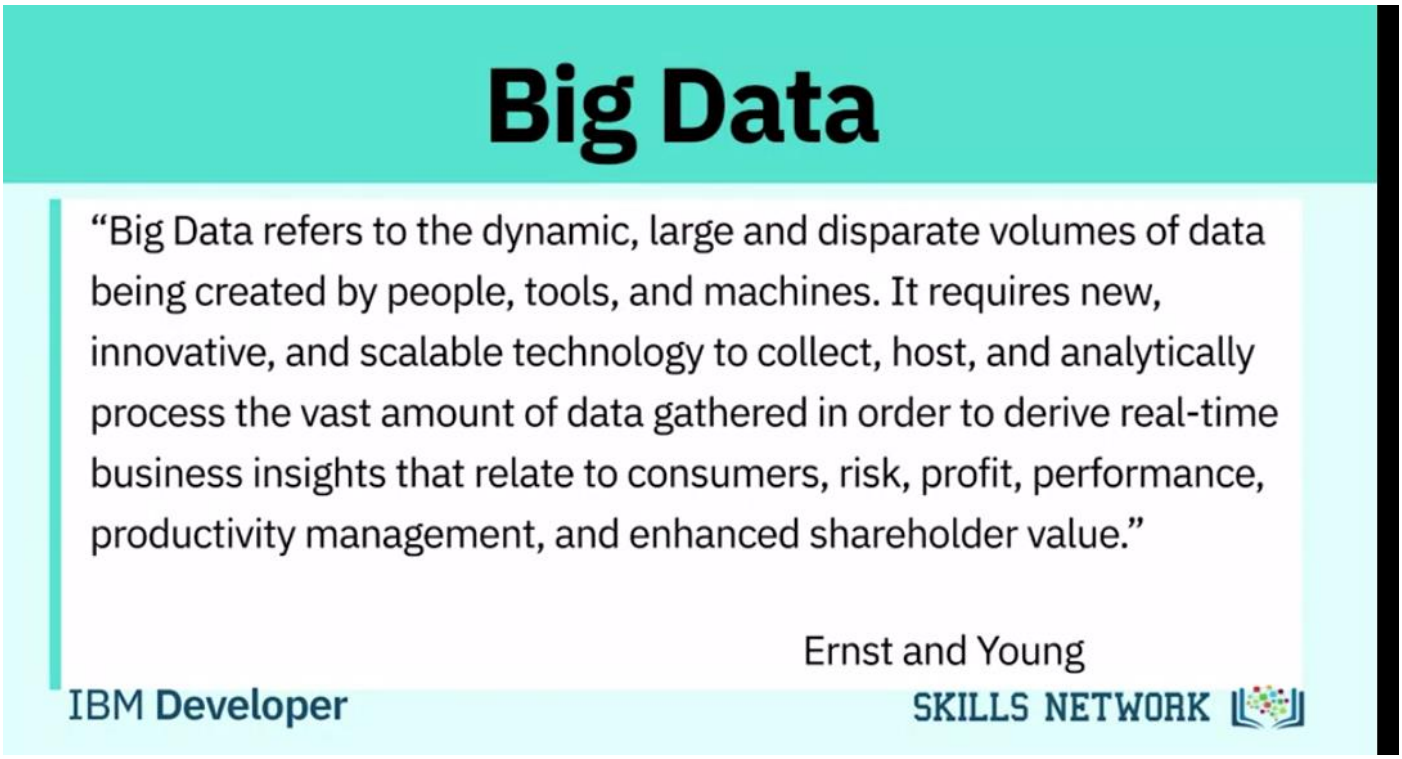
to very easily have some management of

the changes that are made to our

database schemas.

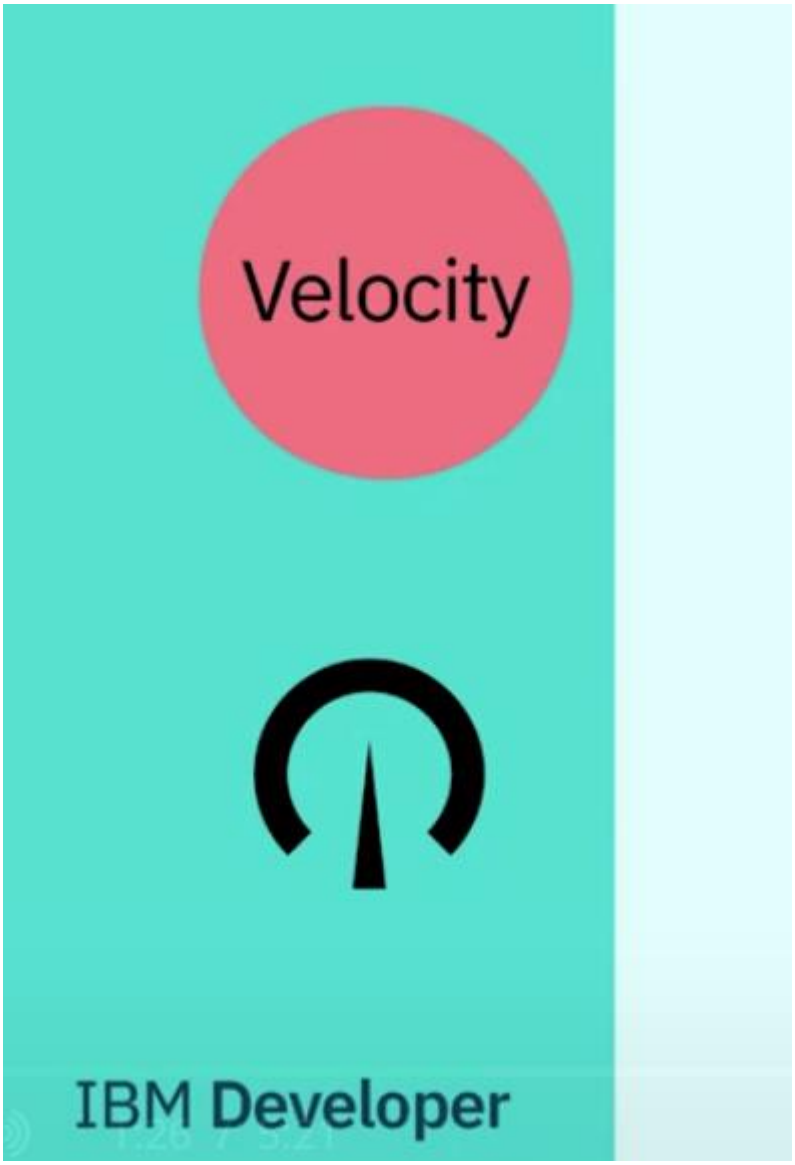
Foundations of Big Data

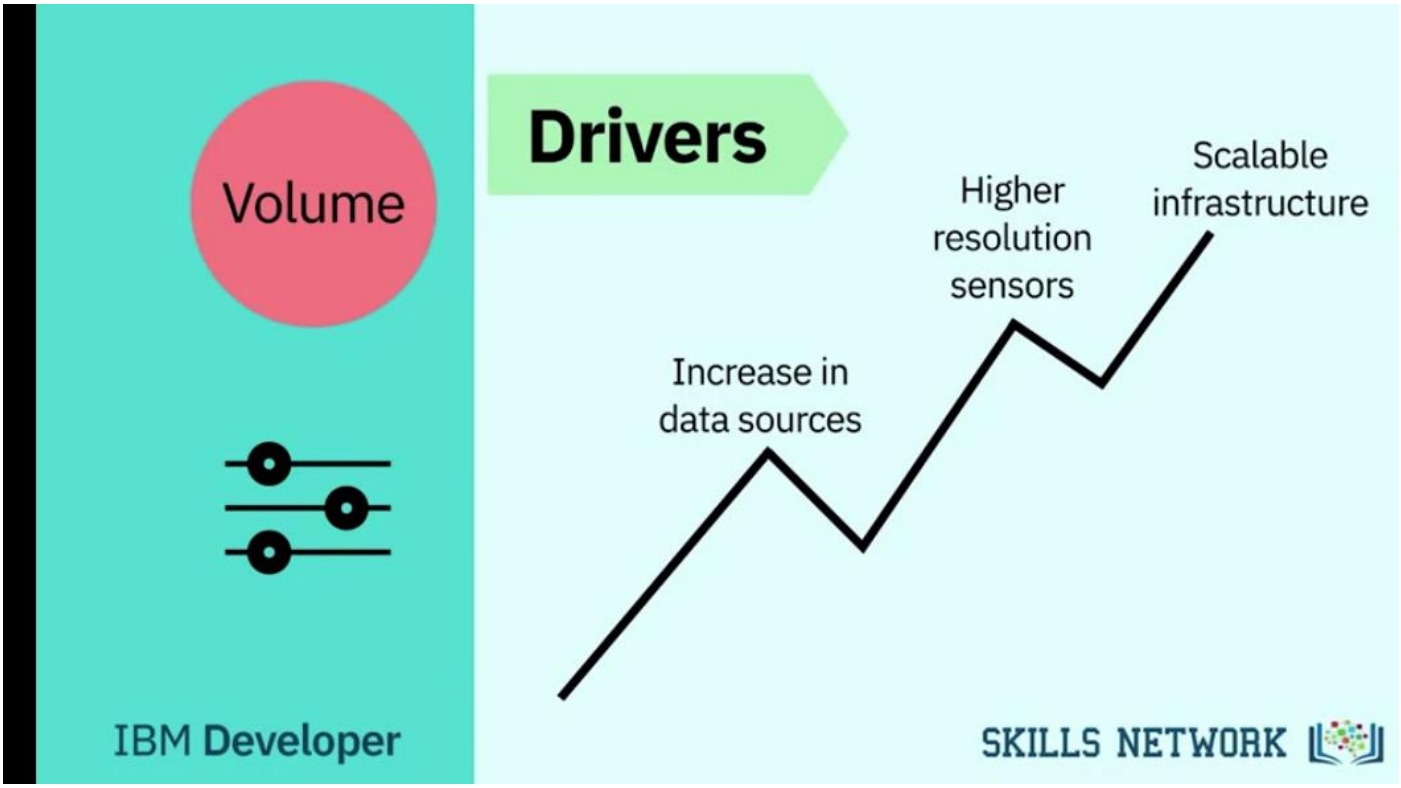
Definition

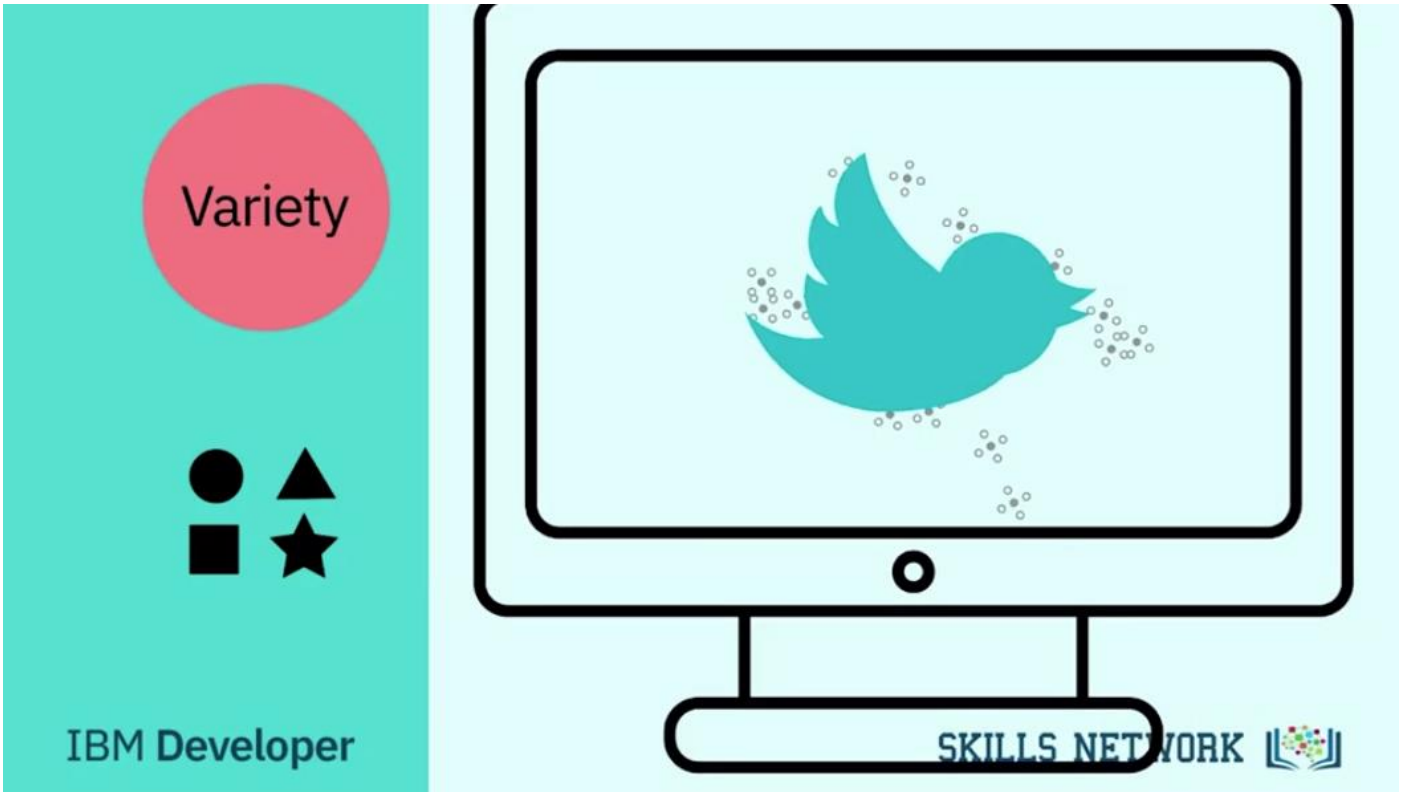


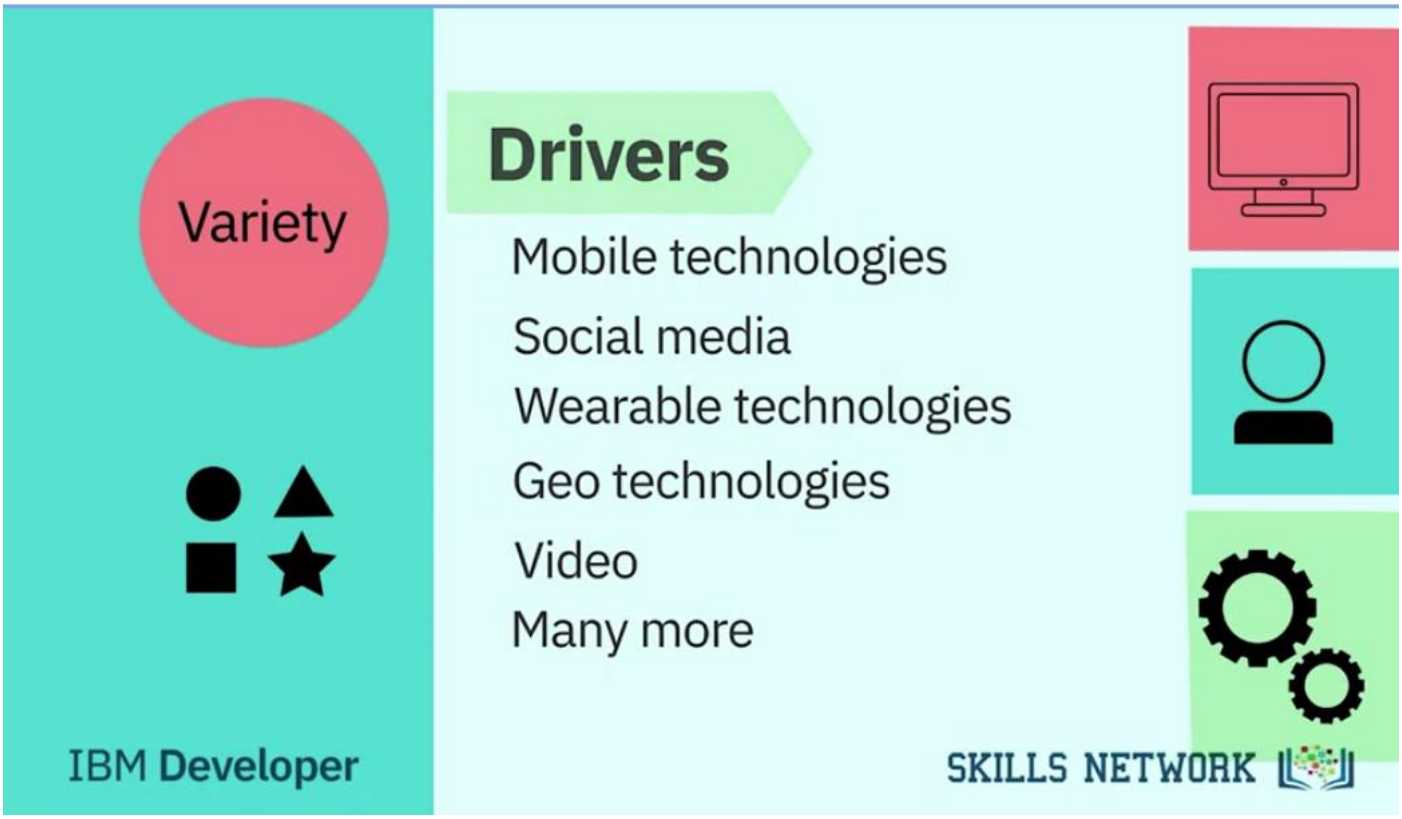
Characteristics

Velocity

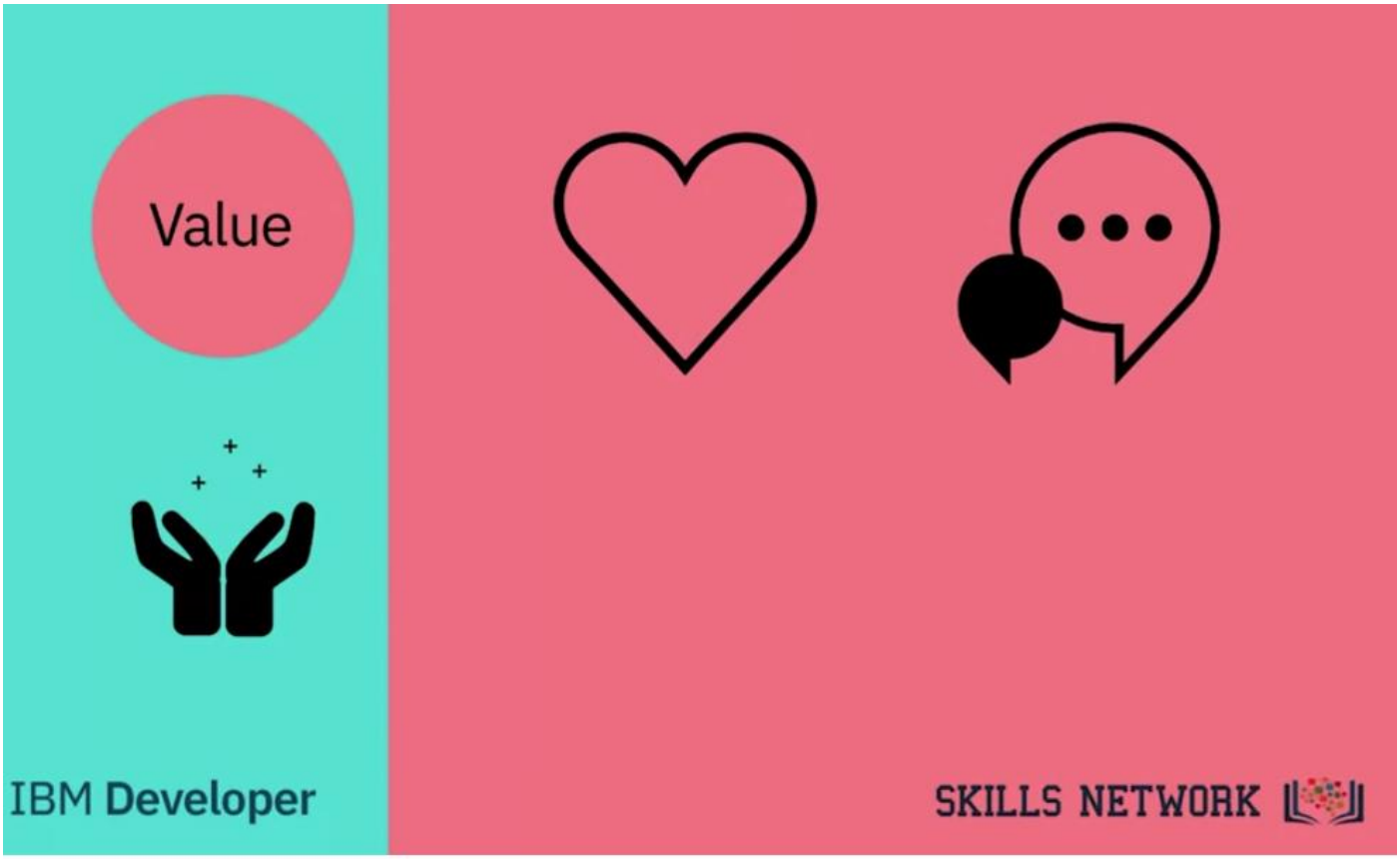



Volume
Drivers
Scalable infrastructure
Higher resolution sensors
Increase in data sources
IBM Developer
SKILLS NETWORK


Variety
IBM Developer
SKILLS NETWORK



Values



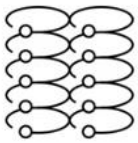
Big Data Processing Tools: Hadoop, HDFS, Hive, and Spark

Big Data Processing Tools

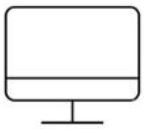
The Big Data processing technologies provide ways to work with large sets of structured, semi-structured, and unstructured data so that value can be derived from big data.

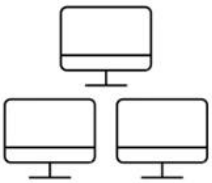
Apache Hadoop
a collection of tools that provides distributed storage and processing of big data

Apache Hive
a data warehouse for data query and analysis



Distributed storage and processing of large datasets
across clusters of computers.


Node


Cluster

Hadoop

Hadoop provides a **reliable, scalable,** and **cost-effective** solution for storing data with no format requirements.

Benefits include:

Better real-time data-driven decisions:
Incorporates emerging data formats not traditionally used in data warehouses

Improved data access and analysis:
Provides real-time, self-service access to stakeholders

Data offload and consolidation:
Optimizes and streamlines costs by consolidating data, including cold data, across the organization

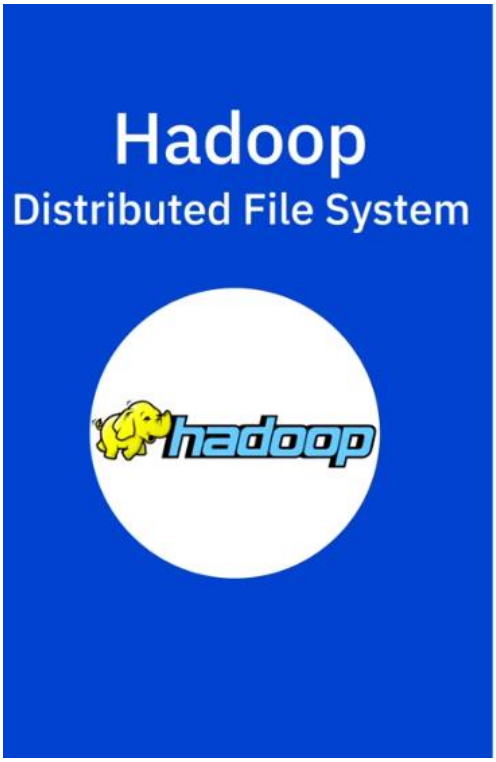
Hadoop
Distributed File System

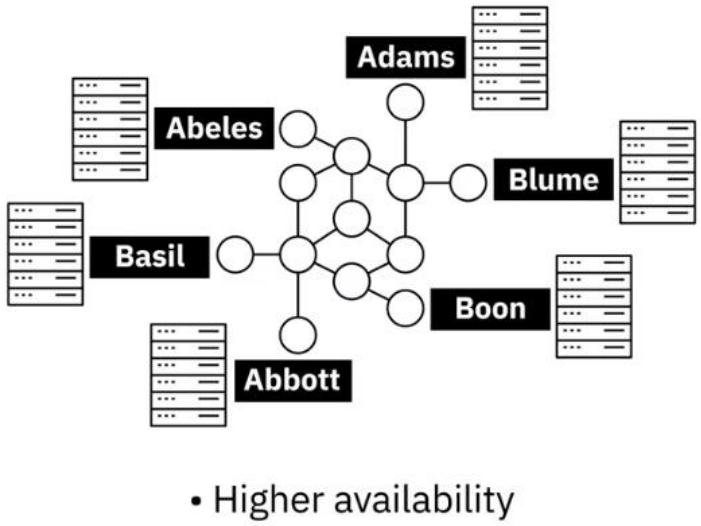
Hadoop Distributed File System, or HDFS, is a storage system for big data that runs on multiple commodity hardware connected through a network.

Provides scalable and reliable big data storage by partitioning files over multiple nodes

Splits large files across multiple computers, allowing parallel access to them

Replicates file blocks on different nodes to prevent data loss

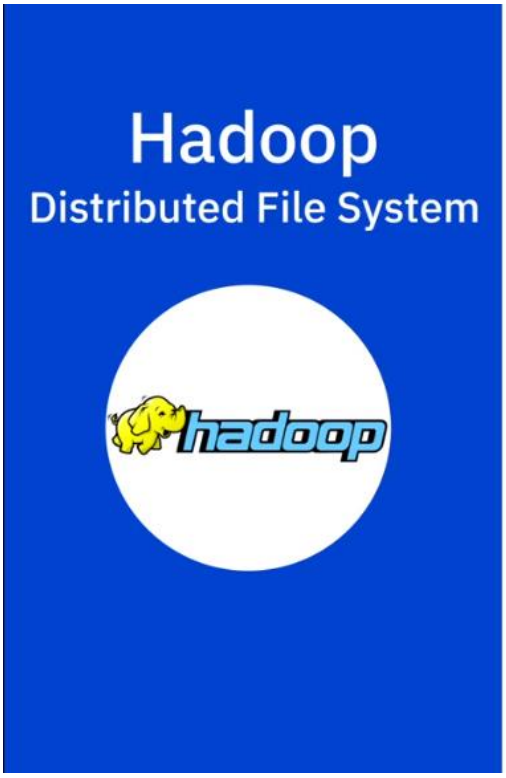




• Higher availability

To reconstruct the entire phonebook, your program would need the blocks from every server in the cluster.
HDFS also replicates these smaller pieces onto two additional servers by default, ensuring availability when a server fails,
In addition to higher availability, this offers multiple benefits.
It allows the Hadoop cluster to break up work into smaller chunks and run those jobs on all servers in the cluster for better scalability.
Finally, you gain the benefit of data locality, which is the process of moving the computation closer to the node on which the data resides.
This is critical when working with large data sets because it minimizes network congestion and increases throughput.

Benefits that com from using HDFS



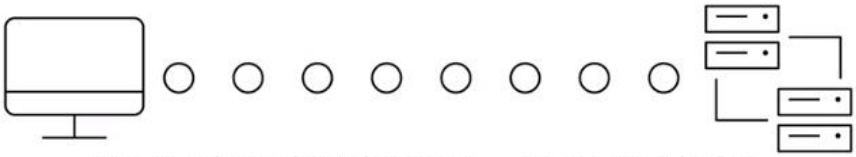
Benefits that come from using HDFS include:

- Fast recovery from hardware failures, because HDFS is built to detect faults and automatically recover.
- Access to streaming data, because HDFS supports high data throughput rates.
- Accommodation of large data sets, because HDFS can scale to hundreds of nodes, or computers, in a single cluster.
- Portability, because HDFS is portable across multiple hardware platforms and compatible with a variety of underlying operating systems.

Hive

Hive

Hive is an open-source data warehouse software for reading, writing, and managing large data set files that are stored directly in either HDFS or other data storage systems such as Apache HBase.



Queries have high latency → Not suitable for applications that need fast response times

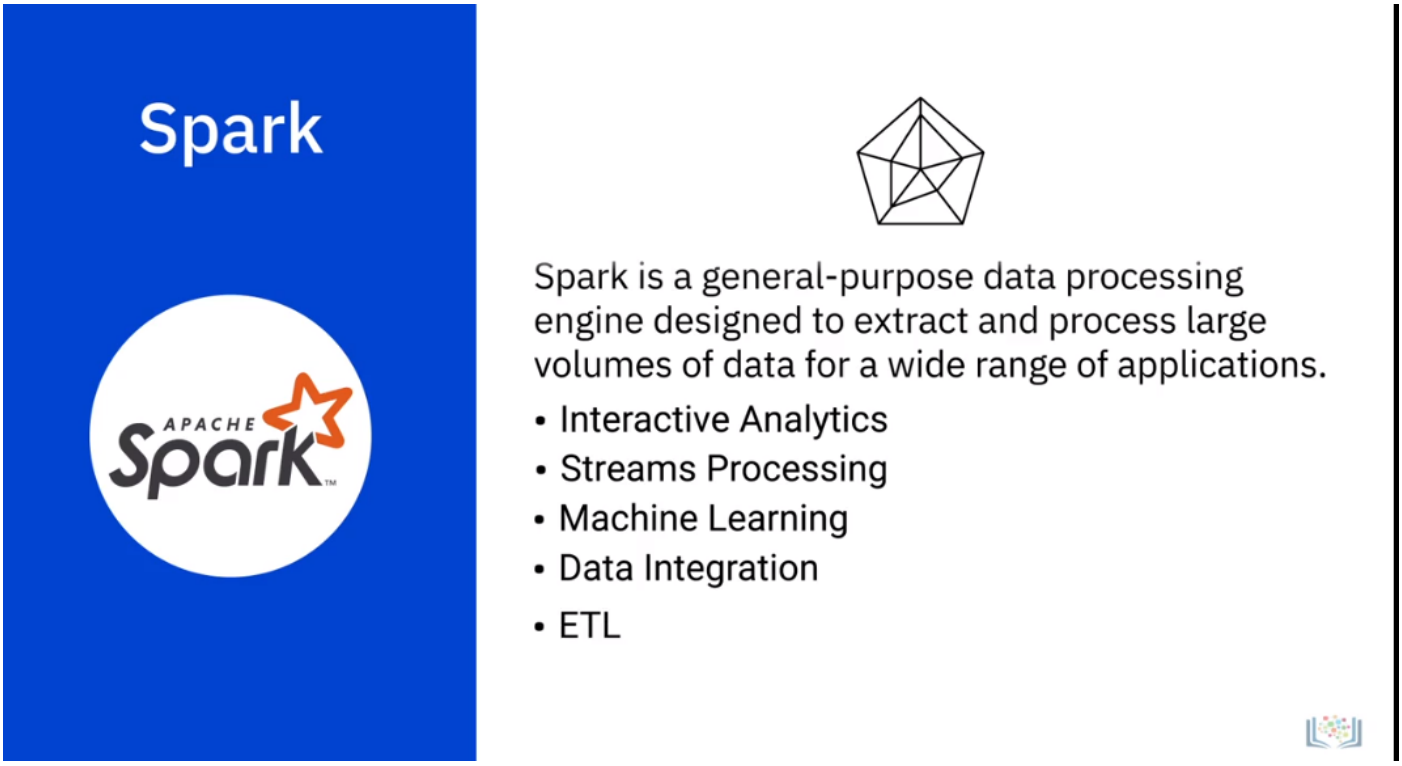
Hive

Read-based → Not suitable for transaction processing that involves a high percentage of write operations.

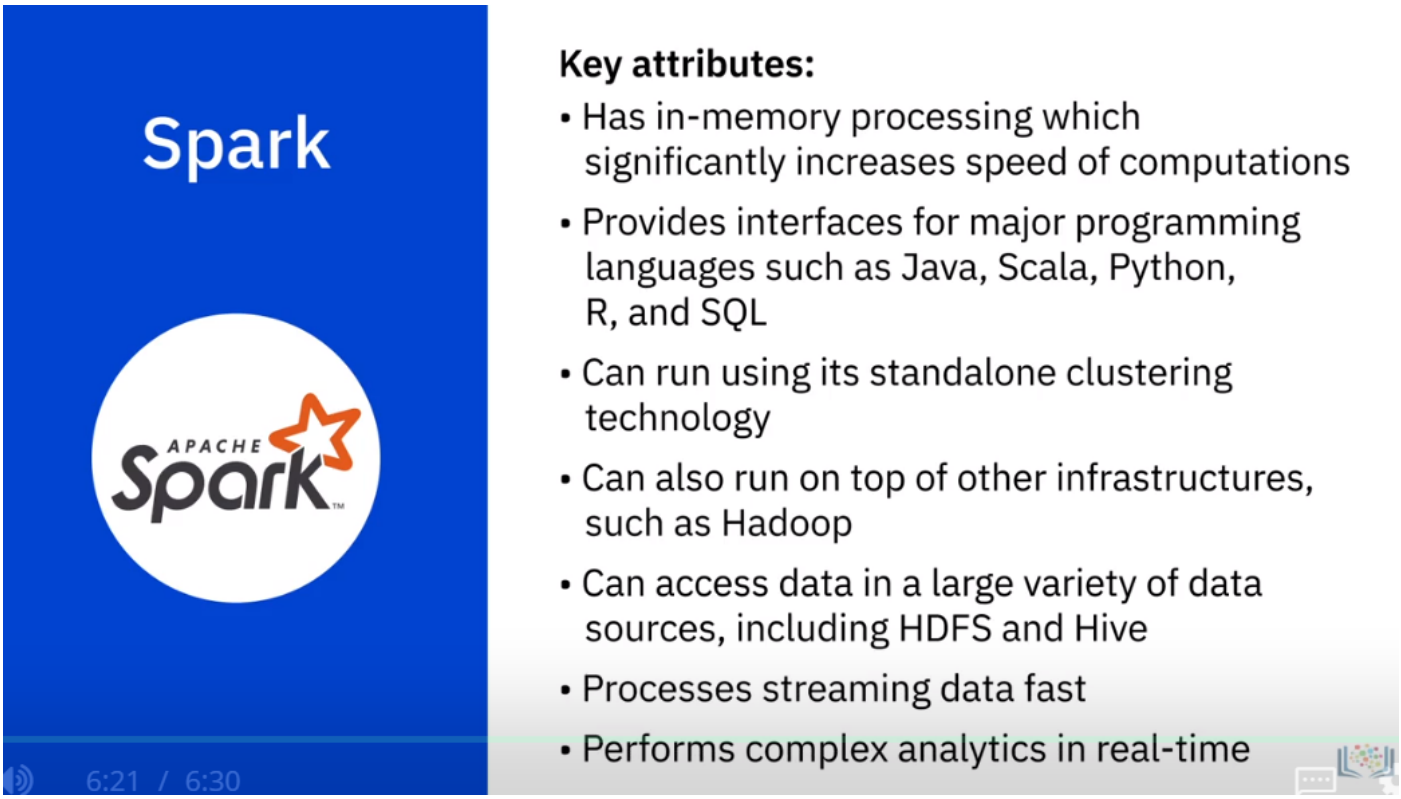
Hive is better suited for →

- Data warehousing tasks such as ETL, reporting, and data analysis

Spark



Spark Key Attributes



Viewpoints: Impact of Big Data on Data Engineering

In this video, we will listen to

several data professionals talk about the impact of Big Data on data engineering.

So how is the Big Data changing and shaping the field

of data engineering. Big Data can be described as the data that

has these four characteristics, also known as four Vs; velocity,

veracity, volume, and variety. Big Data has had a

major impact on data engineering, making it a much

more diverse and a rich field. It's quite natural because

with all the data that the organizations are collecting

these days, being able to make sense of it and derive insights

from it is becoming more and more relevant and critical. And that has opened up a vast

opportunity. Not only have we seen the emergence of new

technologies and products related to working with Big

Data, it's also created a huge demand for professionals who can

effectively work with large volumes and variety of data,

both for building and managing these big data systems, but also

for analyzing data within them. In the pre-IoT on

social media days, the number of ways you can

ingest the data into a database are really small. Like

for example, you have a data entry analyst that's putting data into these

databases at a very slow pace. But in the past decade,

especially with these all new gadgets and devices talking to

each other on these APIs, pushing latest updates to

these different gadgets, the nature of the data itself

changed. As a result of this evolution, the data engineering

envolved as well because your traditional RDBMSes are not a

one-size fits-all solution like a lot of database administrators

and data engineers discovered in a hardware. With these variety of

data sources and volumes, the field of data engineering

itself evolved significantly, and data engineers ended up

inventing these new data sources such as Google BigTable,

Cassandra, Graph-based Databases, Hadoop, MapReduce to analyze these

petabytes of data. So data engineers ended up writing their own tools and also learning and working on new technologies. So there was a big evolution due to Big Data. Big Data has fundamentally changed things in that organizations are much more accepting of storing large amounts of data. Storage for data, actual disk space, is no longer a big issue. We've gotten to the point where we stored a lot more data, and that's what the big data buzzword storm has really brought about. That, and better ways of handling large amounts of data. You know, we can think of Big Data in different ways. There's obviously unstructured data which has become a huge thing and is usually not handled within databases. If it is, it's handled in something like MongoDB. And then there's also just a vast amount more of data that we're dealing with.