**Modern Data Ecosystems**

**Data Sources**

To quote a Forbes 2020 report on data in the coming decade, "The constant increase in data processing speeds and bandwidth, the nonstop invention of new tools for creating, sharing, and consuming data, and the steady addition of new data creators and consumers around the world, ensure that data growth continues unabated.

Data begets more data in a constant virtuous cycle." A modern data ecosystem includes a whole network of interconnected, independent, and continually evolving entities. It includes data that has to be integrated from disparate sources, different types of analysis and skills to generate insights.

Active stakeholders to collaborate and act on insights generated and tools, applications and infrastructure to store, process, and disseminate data as required. Let's start with the data sources.

Data is available in a variety of structured and unstructured datasets, residing in text, images, videos, click streams, user conversations, social media platforms, the Internet of things or IoT devices, real-time events that stream data, legacy databases, and data sourced from professional data providers and agencies.

The sources have never before been so diverse and dynamic. When you're working with so many different sources of data, the first step is to pull a copy of the data from the original sources into a data repository. At this stage, you're only looking at acquiring the data you need working with data formats, sources, and interfaces through which this data can be pulled in. Reliability, security, and integrity of the data being acquired are some of the challenges you work through at this stage

**Enterprise Data Environment**

Once the raw data is in a common place, it needs to get organized, cleaned up, and optimized for access by end users. The data will also need to conform to compliances and standards enforced in the organization.

For example, conforming to guidelines that regulate the storage and use of personal data, such as health, biometrics or household data in the case of IoT devices. Adhering to master data tables within the organization to ensure standardization of master data across all applications and systems of an organization is another example.

The key challenges at this stage could involve data management and working with data repositories that provide high availability, flexibility, accessibility, and security

**Users**

Finally, we have our business stakeholders: applications, programmers, analysts, and data science use cases, all pulling this data from the enterprise data repository. The key challenges at this stage could include the interfaces, APIs, and applications that can get this data to the end users in-line with their specific needs.

For example, data analysts may need the raw data to work with. Business stakeholders may need reports and dashboards. Applications may need custom APIs to pull this data.

**Emerging technologies shaping the modern data ecosystem**

It's important to note the influence of some of the new and emerging technologies that are shaping today's data ecosystem and its possibilities, for example: cloud computing, machine learning, and big data, to name a few.

Thanks to cloud technologies, every enterprise today has access to limitless storage, high-performance computing, open source technologies, machine learning technologies, and the latest tools and libraries. Data scientists are creating predictive models by training machine learning algorithms on past data, also big data.

Today, we're dealing with datasets that are so massive and so varied that traditional tools and analysis methods are no longer adequate, paving the way for new tools and techniques and also new knowledge and insights. We'll learn more about big data and its influence in shaping business decisions further along in this course.

**Key Players in the Data Ecosystem**

**Overview**

Today, organizations that are using data to uncover opportunities and are applying that knowledge to differentiate themselves are the ones leading into the future. Whether looking for patterns in financial transactions to detect fraud, using recommendation engines to drive conversion, mining, social media posts for customer voice or brands personalizing their offers based on customer behavior analysis, business leaders realized that data holds

the key to competitive advantage. To get value from data, you need a vast number of skill sets and people playing different roles.

**Data Professionals**

In this video, we're going to look at the role data engineers, data analysts, data scientists, business analysts, and business intelligence or BI analysts play in helping organizations tap into vast amounts of data and turn them into actionable insights. It all starts with a **data engineer**.

* Data engineers are people who develop and maintain data architectures and make data available for business operations and analysis.
* Data engineers work within the data ecosystem to extract, integrate, and organize data from disparate sources
* Clean transform and prepare data design, store and manage data in data repositories.
* They enabled data to be accessible in formats and systems that the various business applications as well as stakeholders like data analysts and data scientists can utilize.
* A data engineer must have good knowledge of programming, sound knowledge of systems and technology architectures, and in depth understanding of relational databases and non-relational data stores.

Now let's look at the role of a **data analyst.**

* In short, a data analyst translates data and numbers into plain language, so organizations can make decisions, data analysts inspect and clean data for deriving insights, identify correlations, find patterns, and apply statistical methods to.
* Analyze and mined data and visualize data to interpret and present the findings of data analysis.
* Analysts are the people who answer questions such as, Are the users search experiences generally good or bad with the search functionality on our site? or What is the popular perception of people regarding our rebranding initiatives? Or is there a correlation between sales, and one product and another?
* Data analysts require good knowledge of spreadsheets, writing queries, and using statistical tools to create charts and dashboards.
* Modern data analysts also need to have some programming skills.
* They also need strong analytical and storytelling skills.

And now let's look at the role **data scientists** play in this ecosystem.

* Data scientists analyze data for actionable insights and build machine learning or deep learning models that train on past data to create predictive models.
* Data scientists are people who answer questions such as, How many new social media followers am I likely to get next month, or what percentage of my customers am I likely to lose to competition in the next quarter, or is this financial transaction unusual for this customer?
* Data scientists require knowledge of mathematics, statistics, and a fair understanding of programming languages, databases, and building data models.
* They also need to have domain knowledge.

Then we also have **business analysts and BI analysts.**

* Business analysts leverage the work of data analysts and data scientists to look at possible implications for their business and the actions they need to take or recommend.
* BI analysts do the same except. Their focus is on the market forces and external influences that shape their business.
* They provide business intelligent solutions by organizing and monitoring data on different business functions and exploring that data to extract insights and actionables that improve business performance.

**Summery**

To summarize, in simple terms, data engineering converts raw data into usable data. Data analytics uses this data to generate insights. Data scientists use data analytics and data engineering to predict the future using data from the past, business analysts and business intelligence analysts use these insights and predictions to drive decisions that benefit and grow their business. Interestingly, it's not uncommon for data professionals to start their career in one of the data roles and transition to another role within the data ecosystem by supplementing their skills.

**Defining Data Analysis**

Data analysis is the process of gathering, cleaning, analyzing and mining data, interpreting results, and reporting the findings. With data analysis we find patterns within data and correlations between different data points.

And it is through these patterns and correlations that insights are generated, and conclusions are drawn. Data analysis helps businesses understand their past performance and informs their decision-making for future actions. Using data analysis, businesses can validate a course of action before committing to it. Saving valuable time and resources and also ensuring greater success

**Different types of Data Analysis**

We will explore four primary types of data analysis, each with a different goal and place in the data analysis process.

* **Descriptive Analytics** helps answer questions about what happened over a given period of time by summarizing past data and presenting the findings to stakeholders. It helps provide essential insights into past events. For example, tracking past performance based on the organization's key performance indicators or cash flow analysis.
* **Diagnostic analytics** helps answer the question. Why did it happen? It takes the insights from descriptive analytics to dig deeper to find the cause of the outcome. For example, a sudden change in traffic to a website without an obvious cause or an increase in sales in a region where there has been no change in marketing.
* **Predictive analytics** helps answer the question, What will happen next? Historical data and trends are used to predict future outcomes. Some of the areas in which businesses apply predictive analysis are risk assessment and sales forecasts. It's important to note that the purpose of predictive analytics is not. to say what will happen in the future, it's objective is to forecast what might happen in the future. All predictions are probabilistic in nature.
* Prescriptive Analytics helps answer the question, What should be done about it? By analyzing past decisions and events, the likelihood of different outcomes. Is estimated on the basis of which a course of action is decided. Self-driving cars are a good example of Prescriptive Analytics. They analyze the environment to make decisions regarding speed, changing lanes, which route to take, etc. Or airlines automatically adjusting ticket prices based on customer demand. Gas prices, the weather or traffic on connecting routes

**The Data Analysis Process**

Now let's look at some of the key steps in any data analysis process. Understanding the problem and desired result.

* Data analysis begins with understanding the problem that needs to be solved and the desired outcome that needs to be achieved. Where you are and where you want to be needs to be clearly defined before the analysis process can begin.
* Setting a clear metric.
* This stage of the process includes deciding what will be measured. For example, number of product X sold in a region and how it will be measured, for example. In a quarter or during a festival season, gathering data once you know what you're going to measure and how you're going to measure it, you identify the data you require, the data sources you need to pull this data from, and the best tools for the job.
* Cleaning data. Having gathered the data, the next step is to fix quality issues in the data that could affect the accuracy of the analysis. This is a critical step because the accuracy of the analysis can only be ensured if the data is clean. You will clean the data for missing or incomplete values and outliers. For example, a customer demographics data in which the age field has a value of 150 is an outlier. You will also standardize the data coming in from multiple sources.
* Analyzing and mining data. Once the data is clean, you will extract and analyze the data from different perspectives. You may need to manipulate your data in several different ways to understand the trends, identify correlations and find patterns and variations. Interpreting results. After analyzing your data and possibly conducting further research, which can be an iterative loop, it's time to interpret your results. As you interpret your results, you need to evaluate if your analysis is defendable against objections, and if there are any limitations or circumstances under which your analysis may not hold true. Presenting your findings.

Ultimately, the goal of any analysis is to impact decision making. The ability to communicate and present your findings in clear and impactful ways is as important a part of the data analysis process as is the analysis itself. Reports, dashboards, charts, graphs, maps, case studies are just some of the ways in which you can present your data.

**Responsibilities of a Data Analyst**

While the role of a Data Analyst varies depending on the type of organization and the extent to which it has adopted data-driven practices, there are some responsibilities that are typical to a Data Analyst role in today’s organizations. These include:

* Acquiring data from primary and secondary data sources,
* Creating queries to extract required data from databases and other data collection systems,
* Filtering, cleaning, standardizing, and reorganizing data in preparation for data analysis,
* Using statistical tools to interpret data sets,
* Using statistical techniques to identify patterns and correlations in data,
* Analyzing patterns in complex data sets and interpreting trends,
* Preparing reports and charts that effectively communicate trends and patterns,
* Creating appropriate documentation to define and demonstrate the steps of the data analysis process.

**Skills**

Corresponding to these responsibilities, let’s look at some of the skills that are valuable for a Data Analyst. The data analysis process requires a combination of technical, functional, and soft skills. Let’s first look at some of the technical skills that you need in your role as a Data Analyst.

These include:

* Expertise in using spreadsheets such as Microsoft Excel or Google Sheets,
* Proficiency in statistical analysis and visualization tools and software such as IBM Cognos, IBM SPSS, Oracle Visual Analyzer, Microsoft Power BI, SAS, and Tableau
* Proficiency in at least one of the programming languages such as R, Python, and in some cases C++, Java, and MATLAB,
* Good knowledge of SQL, and ability to work with data in relational and NoSQL databases, The ability to access and extract data from data repositories such as data marts, data warehouses, data lakes, and data pipelines,
* Familiarity with Big Data processing tools such as Hadoop, Hive, and Spark.

We will understand more about the features and use cases of some of these programming languages, databases, data repositories, and big data processing tools further along in the course.

**Functions skills**

Now we’ll look at some of the functional skills that you require for the role of Data Analyst.

These include:

* Proficiency in Statistics to help you analyze your data, validate your analysis, and identify fallacies and logical errors.
* Analytical skills that help you research and interpret data, theorize, and make forecasts.
* Problem-solving skills, because ultimately, the end-goal of all data analysis is to solve problems.
* Probing skills that are essential for the discovery process, that is, for understanding a problem from the perspective of varied stakeholders and users—because the data analysis process really begins with a clear articulation of the problem statement and desired outcome.
* Data Visualization skills that help you decide on the techniques and tools that present your findings effectively based on your audience, type of data, context, and end-goal of your analysis.
* Project Management skills to manage the process, people, dependencies, and timelines of the initiative.

That brings us to your soft skills as a Data Analyst. Data Analysis is both a science and an art

**Soft Skills**

You can ace the technical and functional expertise, but one of the key differentiators for your success is going to be soft skills. This includes your ability to:

* Work collaboratively with business and cross-functional teams
* Communicate effectively to report and present your findings
* Tell a compelling and convincing story
* Gather support and buy-in for your work.

Above all, being curious, is at the heart of data analysis. In the course of your work, you will stumble upon patterns, phenomena, and anomalies that may show you a different path.

The ability to allow new questions to surface and challenge your assumptions and hypotheses makes for a great analyst. You will also hear data analysis practitioners talk about intuition as a must-have quality.

It’s essential to note that intuition, in this context, is the ability to have a sense of the future based on pattern recognition and past experiences. In this video, we learned about the responsibilities and skillsets of a Data Analyst.

**Overview of the Data Analyst Ecosystem**

**Data**

quick overview of the ecosystem before going into the details of each of these topics in subsequent videos. Let’s first talk about data. Based on how well-defined the structure of the data is, data can be categorized as structured, semi-structured, or unstructured.

* Data that follows a rigid format and can be organized neatly into rows and columns is structured data. This is the data that you see typically in databases and spreadsheets, for example.
* Semi-structured data is a mix of data that has consistent characteristics and data that doesn’t conform to a rigid structure. For example, emails. An email has a mix of structured data, such as the name of the sender and recipient, but also has the contents of the email, which is unstructured data.
* And then there is unstructured data: Data that is complex, and mostly qualitative information that is impossible to reduce to rows and columns. For example, photos, videos, text files, PDFs, and social media content.

The type of data drives the kind of data repositories that the data can be collected and stored in, and also the tools that can be used to query or process the data. Data also comes in a wide-ranging variety of file formats being collected from a variety of data sources, ranging from relational and non-relational databases, to APIs, web services, data streams, social platforms, and sensor devices

**Data Repositories**

This brings us to data repositories:

A term that includes databases, data warehouses, data marts, data lakes, and big data stores. The type, format, and sources of data influence the type of data repositories that you can use to collect, store, clean, analyze, and mine the data for analysis. If you’re working with big data, for example, you will need big data warehouses, that allow you to store and process large-volume high-velocity data and also frameworks that allow you to perform complex analytics in real-time on big data.

**Languages**

The ecosystem also includes languages that can be classified as query languages, programming languages, and shell and scripting languages. From querying and manipulating data with SQL to developing data applications with Python, and writing shell scripts for repetitive operational tasks, these are important components in a data analyst’s workbench.

Automated tools, frameworks, and processes for all stages of the analytics process are part of the Data Analysts ecosystem. From tools used for gathering, extracting, transforming, and loading data into data repositories, to tools for data wrangling, data cleaning, data mining, analysis, and data visualization — it's a very diverse and rich ecosystem. Spreadsheets, Jupyter Notebooks, and IBM Cognos are just a few examples. We will cover some of the data analytics tools in greater detail in subsequent sections of the course.

**Types of Data**

Data is unorganized information that is processed to make it meaningful. Generally, data comprises of facts, observations, perceptions, numbers, characters, symbols, and images that can be interpreted to derive meaning. One of the ways in which data can be categorized is by its structure.

Data can be:

* Structured
* Semi-structured, or
* Unstructured

**Structured Data**

Structured data has a well-defined structure or adheres to a specified data model, can be stored in well-defined schemas such as databases, and in many cases can be represented in a tabular manner with rows and columns. Structured data is objective facts and numbers that can be collected, exported, stored, and organized in typical databases.

Some of the sources of structured data could include:

* SQL Databases and Online Transaction Processing (or OLTP) Systems that focus on business transactions,
* Spreadsheets such as Excel and Google Spreadsheets, Online forms, Sensors such as Global Positioning Systems (or GPS) and Radio Frequency Identification (or RFID) tags; and
* Network and Web server logs.

You can typically store structured data in relational or SQL databases. You can also easily examine structured data with standard data analysis methods and tools.

**Semi-structured Data**

Semi-structured data is data that has some organizational properties but lacks a fixed or rigid schema. Semi-structured data cannot be stored in the form of rows and columns as in databases. It contains tags and elements, or metadata, which is used to group data and organize it in a hierarchy.

Some of the sources of semi-structured data could include:

* E-mails, XML, and other markup languages,
* Binary executables,
* TCP/IP packets,
* Zipped files,
* Integration of data from different sources.

XML and JSON allow users to define tags and attributes to store data in a hierarchical form and are used widely to store and exchange semi-structured data.

**Unstructured Data**

unstructured data is data that does not have an easily identifiable structure and, therefore, cannot be organized in a mainstream relational database in the form of rows and columns. It does not follow any particular format, sequence, semantics, or rules. Unstructured data can deal with the heterogeneity of sources and has a variety of business intelligence and analytics applications.

Some of the sources of unstructured data could include:

* Web pages,
* Social media feeds,
* Images in varied file formats (such as JPEG, GIF, and PNG),
* Video and audio files,
* Documents and PDF files,
* PowerPoint presentations,
* Media logs and
* Surveys.

Unstructured data can be stored in files and documents (such as a Word doc) for manual analysis or in NoSQL databases that have their own analysis tools for examining this type of data.

**Summery**

To summarize, structured data is data that is well organized in formats that can be stored in databases and lends itself to standard data analysis methods and tools; Semi-structured data is data that is somewhat organized and relies on meta tags for grouping and hierarchy; and Unstructured data is data that is not conventionally organized in the form of rows and columns in a particular format

**Understanding Different Types of File formats**

As a data professional, you will be working with a variety of data file types, and formats. It is important to understand the underlying structure of file formats along with their benefits and limitations.

This understanding will support you to make the right decisions on the formats best suited for your data and performance needs. Some of the standard file formats that we will cover in this video include:

* Delimited text file formats,
* Microsoft Excel Open XML Spreadsheet, or XLSX
* Extensible Markup Language, or XML,
* Portable Document Format, or PDF,
* JavaScript Object Notation, or JSON

**Delimited test files**

Delimited text files are text files used to store data as text in which each line, or row, has values separated by a delimiter, where a delimiter is a sequence of one or more characters for specifying the boundary between independent entities or values.

Any character can be used to separate the values, but most common delimiters are the comma, tab, colon, vertical bar, and space. Comma-separated values (or CSVs) and tab-separated values (or TSVs) are the most commonly used file types in this category.

In CSVs, the delimiter is a comma while in TSVs, the delimiter is a tab. When literal commas are present in text data and therefore cannot be used as delimiters, TSVs serve as an alternative to CSV format.

Tab stops are infrequent in running text. Each row, or horizontal line, in the text file has a set of values separated by the delimiter and represents a record.

The first row works as a column header, where each column can have a different type of data. For example, a column can be of date type, while another can be a string or integer type data.

Delimited files allow field values of any length and are considered a standard format for providing straightforward information schema. They can be processed by almost all existing applications. Delimiters also represent one of various means to specify boundaries in a data stream.

**Microsoft Excel Open XML Spreadsheet, or .XLSX**

Microsoft Excel Open XML Spreadsheet, or XLSX, is a Microsoft Excel Open XML file format that falls under the spreadsheet file format. It is an XML-based file format created by Microsoft. In an .XLSX, also known as a workbook, there can be multiple worksheets.

And each worksheet is organized into rows and columns, at the intersection of which is the cell. Each cell contains data. XLSX uses the open file format, which means it is generally accessible to most other applications. It can use and save all functions available in Excel and is also known to be one of the more secure file formats as it cannot save malicious code

**Extensible Markup Language, or XML**

Extensible Markup Language, or XML, is a markup language with set rules for encoding data. The XML file format is both readable by humans and machines. It is a self-descriptive language designed for sending information over the internet.

XML is similar to HTML in some respects, but also has differences. For example, an .XML does not use predefined tags like .HTML does. XML is platform independent and programming language independent and therefore simplifies data sharing between various systems.

**Portable Document Format, or PDF**

Portable Document Format, or PDF, is a file format developed by Adobe to present documents independent of application software, hardware, and operating systems, which means it can be viewed the same way on any device. This format is frequently used in legal and financial documents and can also be used to fill in data such as for forms

**JavaScript Object Notation, or JSON**

JavaScript Object Notation, or JSON, is a text-based open standard designed for transmitting structured data over the web. The file format is a language-independent data format that can be read in any programming language.

JSON is easy to use, is compatible with a wide range of browsers, and is considered as one of the best tools for sharing data of any size and type, even audio and video. That is one reason, many APIs and Web Services return data as JSON. In this video, we looked at some popular file and data formats. In the next video, we will learn about the different sources of data.

**Sources of Data**

As we touched upon in one of our previous videos, data sources have never been as dynamic and diverse as they are today. In this video, we will look at some common sources such as:

* Relational Databases,
* Flat files and XML Datasets,
* APIs and Web Services,
* Web Scraping,
* Data Streams, and Feeds.

**Relational Databases**

Typically, organizations have internal applications to support them in managing their day to day business activities, customer transactions, human resource activities, and their workflows. These systems use relational databases such as SQL Server, Oracle, MySQL, and IBM DB2, to store data in a structured way.

Data stored in databases and data warehouses can be used as a source for analysis. For example, data from a retail transactions system can be used to analyze sales in different regions, and data from a customer relationship management system can be used for making sales projections.

External to the organization, there are other publicly and privately available datasets. For example, government organizations releasing demographic and economic datasets on an ongoing basis.

Then there are companies that sell specific data, for example, Point-of-Sale data or Financial data, or Weather data, which businesses can use to define strategy, predict demand, and make decisions related to distribution or marketing promotions, among other things. Such data sets are typically made available as flat files, spreadsheet files, or XML documents.

**Flat File and XML Datasets**

Flat files, store data in plain text format, with one record or row per line, and each value separated by delimiters such as commas, semi-colons or tabs. Data in a flat file maps to a single table, unlike relational databases that contain multiple tables.

One of the most common flat file format is CSV in which values are separated by commas. Spreadsheet files are a special type of flat files, that also organize data in a tabular format – rows and columns.

But a spreadsheet can contain multiple worksheets, and each worksheet can map to a different table. Although data in spreadsheets is in plain text, the files can be stored in custom formats and include additional information such as formatting, formulas, etc.

Microsoft Excel, which stores data in .XLS or .XLSX format is probably the most common spreadsheet. Others include Google sheets, Apple Numbers, and LibreOffice.

XML files, contain data values that are identified or marked up using tags. While data in flat files is “flat” or maps to a single table, XML files can support more complex data structures, such as hierarchical.

Some common uses of XML include data from online surveys, bank statements, and other unstructured data sets

**APIs and Web Services**

Many data providers and websites provide APIs, or Application Program Interfaces, and Web Services, which multiple users or applications can interact with and obtain data for processing or analysis.

APIs and Web Services typically listen for incoming requests, which can be in the form of web requests from users or network requests from applications and return data in plain text, XML, HTML, JSON, or media files.

**Popular examples of APIs**

Let’s look at some popular examples of APIs being used as a data source for data analytics:

* The use of Twitter and Facebook APIs to source data from tweets and posts for performing tasks such as opinion mining or sentiment analysis, which is to summarize the amount of appreciation and criticism on a given subject, such as policies of a government, a product, a service, or customer satisfaction in general.
* Stock Market APIs used for pulling data such as share and commodity prices, earnings per share, and historical prices, for trading and analysis.
* Data Lookup and Validation APIs, which can be very useful for Data Analysts for cleaning and preparing data, as well as for co-relating data—for example, to check which city or state a postal or zip code belongs to.
* APIs are also used for pulling data from database sources, within and external to the organization

**Web Scraping**

Web scraping is used to extract relevant data from unstructured sources. Also known as screen scraping, web harvesting, and web data extraction, web scraping makes it possible to download specific data from web pages based on defined parameters.

Web scrapers can, among other things, extract text, contact information, images, videos, product items, and much more from a website.

Some popular uses of web scraping include:

* collecting product details from retailers, manufacturers, and eCommerce websites to provide price comparisons,
* generating sales leads through public data sources,
* extracting data from posts and authors on various forums and communities and collecting training and testing datasets for machine learning models.

Some of the popular web scraping tools include:

* BeautifulSoup
* Scrapy
* Pandas, and
* Selenium.

**Data Steams and feeds**

Data streams are another widely used source for aggregating constant streams of data flowing from sources such as instruments, IoT devices and applications, GPS data from cars, computer programs, websites, and social media posts.

This data is generally timestamped and also geo-tagged for geographical identification. Some of the data streams and ways in which they can be leveraged include:

* stock and market tickers for financial trading,
* retail transaction streams for predicting demand and supply chain management, surveillance and video feeds for threat detection,
* social media feeds for sentiment analysis,
* sensor data feeds for monitoring industrial or farming machinery,

web click feeds for monitoring web performance and improving design, and real-time flight events for rebooking and rescheduling.

Some popular applications used to process data streams include Apache Kafka, Apache Spark Streaming, and Apache Storm.

RSS (or Really Simple Syndication) feeds, are another popular data source. These are typically used for capturing updated data from online forums and news sites where data is refreshed on an ongoing basis. Using a feed reader, which is an interface that converts RSS text files into a stream of updated data, updates are streamed to user devices.

**Languages for Data Professional**

In this video, we will learn about some of the languages relevant to the work of data professionals. These can be categorized as – query languages, programming languages, and shell scripting. Having proficiency in at least one language in each category is essential for any data professional.

Simply stated:

Query languages are designed for accessing and manipulating data in a database; for example, SQL, Programming languages are designed for developing applications and controlling application behavior; for example, Python, R, and Java; and Shell and Scripting languages, such as Unix/Linux Shell, and PowerShell, are ideal for repetitive and time-consuming operational tasks.

**Structured Query Language (SQL)**

SQL, or Structured Query Language, is a querying language designed for accessing and manipulating information from, mostly, though not exclusively, relational databases. Using SQL, we can write a set of instructions to perform operations such as Insert, update, and delete records in a database; Create new databases, tables, and views; and Write stored procedures—which means you can write a set of instructions and call them for later use

Here are some advantages of using SQL:

* SQL is portable and can be used independent of the platform, It can be used for querying data in a wide variety of databases and data repositories, although each vendor may have some variations and special extensions,
* It has a simple syntax that is similar to the English language, Its syntax allows developers to write programs with fewer lines than some of the other programming languages using basic keywords such as select, insert, into, and update, It can retrieve large amounts of data quickly and efficiently,
* It runs on an interpreter system, which means code can be executed as soon as it is written, making prototyping quick and easy.

SQL is one of the most popular querying languages. Due to its large user community and the sheer volume of documentation accumulated over the years, it continues to provide a uniform platform, worldwide, to all its users

**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, as compared to some of the older languages.

Python is perceived as one of the easiest languages to learn and has a large developer community. Because of its focus on simplicity and readability, and a low learning curve, it’s an ideal tool for beginning programmers.

It is great for performing high-computational tasks in vast amounts of data, which can otherwise be extremely time-consuming and cumbersome. Python provides libraries like Numpy and Pandas, which eases this task by the use of parallel processing.

It has inbuilt functions for almost all of the frequently used concepts. Python supports multiple programming paradigms, such as object-oriented, imperative, functional, and procedural, making it suitable for a wide variety of use cases.

Now let’s look at some of the reasons that make Python one of the fastest-growing programming languages in the world today.

* It is easy to learn - With Python, you have the advantage of using fewer lines of code to accomplish tasks compared to other languages.
* It is open-source — Python is free and uses a community-based model for development.
* It runs on Windows and Linux environments and can be ported to multiple platforms.
* It has widespread community support with plenty of useful analytics libraries available.
* It has several open-source libraries for data manipulation, data visualization, statistics, and mathematics, to name just a few.
* 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, Opencv for image processing.

**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 and performing data analytics, it is especially known for its ability to create compelling visualizations, giving it an edge over some of the other languages in this space.

Some of the key benefits of R include the following:

* It is an open-source platform-independent programming language,
* It can be paired with many programming languages, including Python,
* It is highly extensible, which means developers can continue to add functionalities by defining new functions,
* It facilitates the handling of structured as well as unstructured data which means it has a more comprehensive data capability,
* It has libraries such as Ggplot2 and Plotly that offer aesthetic graphical plots to its users, You can make reports with the data and scripts embedded in them; also, interactive web apps that allow users to play with the results and the data,
* It is dominant among other programming languages for developing statistical tools

**Java**

Java is an object-oriented, class-based, and platform-independent programming language originally developed by Sun Microsystems. It is among the top-ranked programming languages used today.

Java is used in a number of processes all through data analytics, including cleaning data, importing and exporting data, statistical analysis, and data visualization. In fact, most of the popular frameworks and tools used for big data are typically written in Java, such as Hadoop, Hive, and Spark. It is perfectly suited for speed-critical projects.

**Unix/Linux shell**

Unix/Linux Shell is a computer program written for the UNIX shell. It is a series of UNIX commands written in a plain text file to accomplish a specific task.

Writing a shell script is fast and easy. It is most useful for repetitive tasks that may be time-consuming to execute by typing one line at a time.

Typical operations performed by shell scripts include:

* file manipulation,
* program execution,
* system administration tasks such as disk backups and evaluating system logs,
* installation scripts for complex programs,
* executing routine backups,
* running batches,

**PowerShell**

PowerShell is a cross-platform automation tool and configuration framework by Microsoft that is optimized for working with structured data formats, such as JSON, CSV, XML, and REST APIs, websites, and office applications.

It consists of a command-line shell and scripting language. PowerShell is object-based, which makes it possible to filter, sort, measure, group, compare, and many more actions on objects as they pass through a data pipeline. It is also a good tool for data mining, building GUIs, and creating charts, dashboards, and interactive reports.

**Overview of Data Repositories**

data repository is a general term used to refer to data that has been collected, organized, and isolated so that it can be used for business operations or mined for reporting and data analysis.

It can be a small or large database infrastructure with one or more databases that collect, manage, and store data sets. In this video, we will provide an overview of the different types of repositories your data might reside in, such as databases, data warehouses, and big data stores, and examine them in greater detail in further videos. Let’s begin with databases.

**Databases**

A database is a collection of data, or information, designed for the input, storage, search and retrieval, and modification of data. And a Database Management System, or DBMS, is a set of programs that creates and maintains the database.

It allows you to store, modify, and extract information from the database using a function called querying. For example, if you want to find customers who have been inactive for six months or more, using the query function, the database management system will retrieve data of all customers from the database that have been inactive for six months and more.

Even though a database and DBMS mean different things the terms are often used interchangeably. There are different types of databases. Several factors influence the choice of database, such as the data type and structure, querying mechanisms, latency requirements, transaction speeds, and intended use of the data.

It’s important to mention two main types of databases here—relational and non-relational databases.

**Relational databases**, also referred to as RDBMSes, build on the organizational principles of flat files, with data organized into a tabular format with rows and columns following a well-defined structure and schema. However, unlike flat files, RDBMSes are optimized for data operations and querying involving many tables and much larger data volumes. Structured Query Language, or SQL, is the standard querying language for relational databases.

Then we have **non-relational databases**, also known **as NoSQL, or “Not Only SQL”.** Non-relational databases emerged in response to the volume, diversity, and speed at which data is being generated today, mainly influenced by advances in cloud computing, the Internet of Things, and social media proliferation.

Built for speed, flexibility, and scale, non-relational databases made it possible to store data in a schema-less or free-form fashion. NoSQL is widely used for processing big data

**Data Warehouse**

A data warehouse works as a central repository that merges information coming from disparate sources and consolidates it through the extract, transform, and load process, also known as the ETL process, into one comprehensive database for analytics and business intelligence. At a very high-level, the ETL process helps you to extract data from different data sources, transform the data into a clean and usable state, and load the data into the enterprise’s data repository.

Related to Data Warehouses are the concepts of Data Marts and Data Lakes, which we will cover later. Data Marts and Data Warehouses have historically been relational, since much of the traditional enterprise data has resided in RDBMSes.

However, with the emergence of NoSQL technologies and new sources of data, non-relational data repositories are also now being used for Data Warehousing.

**Big Data Stores**

Another category of data repositories are Big Data Stores, that include distributed computational and storage infrastructure to store, scale, and process very large data sets Overall, data repositories help to isolate data and make reporting and analytics more efficient and credible while also serving as a data archive.

**Relational Database Management System**

A relational database is a collection of data organized into a table structure, where the tables can be linked, or related, based on data common to each. Tables are made of rows and columns, where rows are the “records”, and the columns the “attributes”.

Let’s take the example of a customer table that maintains data about each customer in a company. The columns, or attributes, in the customer table are the Company ID, Company Name, Company Address, and Company Primary Phone; and Each row is a customer record.

Now let’s understand what we mean by tables being linked, or related, based on data common to each. Along with the customer table, the company also maintains transaction tables that contain data describing multiple individual transactions pertaining to each customer.

The columns for the transaction table might include the Transaction Date, Customer ID, Transaction Amount, and Payment Method. The customer table and the transaction tables can be related based on the common Customer ID field.

You can query the customer table to produce reports such as a customer statement that consolidates all transactions in a given period. This capability of relating tables based on common data enables you to retrieve an entirely new table from data in one or more tables with a single query.

It also allows you to understand the relationships among all available data and gain new insights for making better decisions. Relational databases use structured query language, or SQL, for querying data.

We’ll learn more about SQL later in this course. 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. But this is where the similarity ends.

Relational databases, by design, are ideal for the optimized storage, retrieval, and processing of data for large volumes of data, unlike spreadsheets that have a limited number of rows and columns. Each table in a relational database has a unique set of rows and columns and relationships can be defined between tables, which minimizes data redundancy.

Moreover, you can restrict database fields to specific data types and values, which minimizes irregularities and leads to greater consistency and data integrity. Relational databases use SQL for querying data, which gives you the advantage of processing millions of records and retrieving large amounts of data in a matter of seconds. Moreover, the security architecture of relational databases provides controlled access to data and also ensures that the standards and policies for governing data can be enforced

**Examples of RDBMS**

Relational databases range from small desktop systems to massive cloud-based systems. They can be either:

* open-source and internally supported,
* open-source with commercial support, or
* commercial closed-source systems.
* IBM DB2
* Microsoft SQL Server
* MySQL, Oracle Database, and
* PostgreSQL

are some of the popular relational databases.

Cloud-based relational databases, also referred to as Database-as-a-Service, are gaining wide use as they have access to the limitless compute and storage capabilities offered by the cloud. Some of the popular cloud relational databases include:

* Amazon Relational Database Service (RDS)
* Google Cloud SQL
* IBM DB2 on Cloud
* Oracle Cloud, and
* SQL Azure.

RDBMS is a mature and well-documented technology, making it easy to learn and find qualified talent

**Advantages of the Relational Database Approach**

One of the most significant advantages of the relational database approach is its ability to create meaningful information by joining tables. Some of its other advantages include:

* Flexibility: Using SQL, you can add new columns, add new tables, rename relations, and make other changes while the database is running and queries are happening.
* Reduced redundancy: Relational databases minimize data redundancy. For example, the information of a customer appears in a single entry in the customer table, and the transaction table pertaining to the customer stores a link to the customer table.
* Ease of backup and disaster recovery: Relational databases offer easy export and import options, making backup and restore easy. Exports can happen while the database is running, making restore on failure easy. Cloud-based relational databases do continuous mirroring, which means the loss of data on restore can be measured in seconds or less.
* ACID-compliance: ACID stands for Atomicity, Consistency, Isolation, and Durability. And ACID compliance implies that the data in the database remains accurate and consistent despite failures, and database transactions are processed reliably.

**Use Cases for RDBMS**

Now we’ll look at some use cases for relational databases:

* Online Transaction Processing: OLTP applications are focused on transaction-oriented tasks that run at high rates. Relational databases are well suited for OLTP applications because they can accommodate a large number of users

they support the ability to insert, update, or delete small amounts of data and they also support frequent queries and updates as well as fast response times.

* Data warehouses: In a data warehousing environment, relational databases can be optimized for online analytical processing (or OLAP), where historical data is analyzed for business intelligence.
* IoT solutions: Internet of Things (IoT) solutions require speed as well as the ability to collect and process data from edge devices, which need a lightweight database solution.

**Limitations of RDBMS**

This brings us to the limitations of RDBMS:

RDBMS does not work well with semi-structured and unstructured data and is, therefore, not suitable for extensive analytics on such data. For migration between two RDBMSs, schemas and type of data need to be identical between the source and destination tables.

Relational databases have a limit on the length of data fields, which means if you try to enter more information into a field than it can accommodate, the information will not be stored.

Despite the limitations and the evolution of data in these times of big data, cloud computing, IoT devices, and social media, RDBMS continues to be the predominant technology for working with structured data.

**Not only SQL (NoSQL)**

NoSQL, which stands for “not only SQL,” or sometimes “non SQL” is a non-relational database design that provides flexible schemas for the storage and retrieval of data. NoSQL databases have existed for many years but have only recently become more popular in the era of cloud, big data, and high-volume web and mobile applications.

They are chosen today for their attributes around scale, performance, and ease of use. It's important to emphasize that the "No" in "NoSQL" is an abbreviation for "not only" and not the actual word "No." NoSQL databases are built for specific data models and have flexible schemas that allow programmers to create and manage modern applications.

They do not use a traditional row/column/table database design with fixed schemas, and typically not use the structured query language (or SQL) to query data, although some may support SQL or SQL-like interfaces. NoSQL allows data to be stored in a schema-less or free-form fashion. Any data, be It structured, semi-structured, or unstructured, can be stored in any record.

**Four different types of NoSQL Databases**

Based on the model being used for storing data, there are four common types of NoSQL databases. Key-value store, document-based, column-based, and graph-based.

**Key-value store**

* Data in a key-value database is stored as a collection of key-value pairs.
* The key represents an attribute of the data and is a unique identifier.
* Both keys and values can be anything from simple integers or strings to complex JSON documents.
* Key-value stores are great for storing user session data and user preferences, making real-time recommendations and targeted advertising, and in-memory data caching.
* However, if you want to be able to query the data on specific data value, need relationships between data values, or need to have multiple unique keys, a key-value store may not be the best fit.
  + Redis, Memcached, and
  + DynamoDB

are some well-known examples in this category.

**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.
* Document databases are preferable for eCommerce platforms, medical records storage, CRM platforms, and analytics platforms.
* However, if you’re looking to run complex search queries and multi-operation transactions, a document-based database may not be the best option for you.
  + MongoDB
  + DocumentDB
  + CouchDB, and
  + Cloudant

are some of the popular document-based databases.

**Column-based:**

* Column-based models store data in cells grouped as columns of data instead of rows. A logical grouping of columns, that is, columns that are usually accessed together, is called a column family. For example, a customer’s name and profile information will most likely be accessed together but not their purchase history.
* So, customer name and profile information data can be grouped into a column family. Since column databases store all cells corresponding to a column as a continuous disk entry, accessing and searching the data becomes very fast.
* Column databases can be great for systems that require heavy write requests, storing time-series data, weather data, and IoT data.
* But if you need to use complex queries or change your querying patterns frequently, this may not be the best option for you.

The most popular column databases are

* Cassandra and
* HBase

**Graph-based:**

* Graph-based databases use a graphical model to represent and store data.
* They are particularly useful for visualizing, analyzing, and finding connections between different pieces of data.
* The circles are nodes, and they contain the data. The arrows represent relationships.
* Graph databases are an excellent choice for working with connected data, which is data that contains lots of interconnected relationships.
* Graph databases are great for social networks, real-time product recommendations, network diagrams, fraud detection, and access management.
* But if you want to process high volumes of transactions, it may not be the best choice for you, because graph databases are not optimized for large-volume analytics queries.
* Neo4J and
* CosmosDB

are some of the more popular graph databases.

**Advantages of NoSQL**

NoSQL was created in response to the limitations of traditional relational database technology. The primary advantage of NoSQL is its ability to handle large volumes of structured, semi-structured, and unstructured data.

Some of its other advantages include:

* The ability to run as distributed systems scaled across multiple data centers, which enables them to take advantage of cloud computing infrastructure
* 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 enables you to be more agile, more flexible, and to iterate more quickly

**Key differences between relational and non-relational databases**

|  |  |
| --- | --- |
| Relational Databases | Non-relational Databases |
| RDBMS schemas rigidly define how all data inserted into the database must be typed and composed | NoSQL databases can be schema-agnostic, allowing unstructured and semi-structured data to be stored and manipulated |
| Maintaining high-end, commercial relational database management systems is expensive | NoSQL databases are specifically designed for low-cost commodity hardware. |
| Relational databases, unlike most NoSQL, support ACID-compliance, which ensures reliability of transactions and crash recovery. | NoSQL, which is a relatively newer technology. |
| RDBMS is a mature and well-documented technology, which means the risks are more or less perceivable |  |

Nonetheless, NoSQL databases are here to stay, and are increasingly being used for mission

critical applications.

**Data Marts, Data Lakes, ETL, and Data Pipelines**

Earlier in the course, we examined databases, data warehouses, and big data stores. Now we’ll go a little deeper in our exploration of data warehouses, data marts, and data lakes; and also learn about the ETL process and data pipelines.

**Data Warehouse**

A data warehouse works like a multi-purpose storage for different use cases. By the time the data comes into the warehouse, it has already been modeled and structured for a specific purpose, meaning it is analysis ready. As an organization, you would opt for a data warehouse when you have massive amounts of data from your operational systems that needs to be readily available for reporting and analysis.

Data warehouses serve as the single source of truth—storing current and historical data that has been cleansed, conformed, and categorized. A data warehouse is a multi-purpose enabler of operational and performance analytics.

**Data Marts**

A data mart is a sub-section of the data warehouse, built specifically for a particular business function, purpose, or community of users. The idea is to provide stakeholders data that is most relevant to them, when they need it.

For example, the sales or finance teams accessing data for their quarterly reporting and projections. Since a data mart offers analytical capabilities for a restricted area of the data warehouse, it offers isolated security and isolated performance.

The most important role of a data mart is:

* business-specific reporting and
* analytics.

**Data Lake**

A Data Lake is a storage repository that can store large amounts of structured, semi-structured, and unstructured data in their native format, classified and tagged with metadata. So, while a data warehouse stores data processed for a specific need, a data lake is a pool of raw data where each data element is given a unique identifier and is tagged with metatags for further use.

You would opt for a data lake if you generate, or have access to, large volumes of data on an ongoing basis, but don’t want to be restricted to specific or pre-defined use cases. Unlike data warehouses, a data lake would retain all source data, without any exclusions. And the data could include all types of data sources and types.

Data lakes are sometimes also used as a staging area of a data warehouse. The most important role of a data lake is in predictive and advanced analytics.

**Extract, Transform, and Load Process (ETL)**

ow we come to the process that is at the heart of gaining value from data—the Extract, Transform, and Load process, or ETL.

ETL is how raw data is converted into analysis-ready data. It is an automated process in which you gather raw data from identified sources, extract the information that aligns with your reporting and analysis needs, clean, standardize, and transform that data into a format that is usable in the context of your organization; and load it into a data repository. While ETL is a generic process, the actual job can be very different in usage, utility, and complexity.

**Extract** is the step where data from source locations is collected for transformation. Data extraction could be through:

* Batch processing, meaning source data, is moved in large chunks from the source to the target system at scheduled intervals.
* Tools for batch processing include Stitch and Blendo.
* Stream processing, which means source data is pulled in real-time from the source and transformed while it is in transit and before it is loaded into the data repository.
* Tools for stream processing include Apache Samza, Apache Storm, and Apache Kafka.

**Transform** involves the execution of rules and functions that converts raw data into data that can be used for analysis. For example:

* making date formats and units of measurement consistent across all sourced data
* removing duplicate data
* filtering out data that you do not need
* enriching data, for example:
  + splitting full name to first, middle, and last names
* establishing key relationships across tables
* applying business rules and data validations.

**Load** is the step where processed data is transported to a destination system or data repository. It could be:

* Initial loading, that is, populating all the data in the repository, Incremental loading, that is, applying ongoing updates and modifications as needed periodically, or
* Full refresh, that is, erasing contents of one or more tables and reloading with fresh data.
* Load verification, which includes data checks for missing or null values, server performance, and monitoring load failures, are important parts of this process step.
* It is vital to keep an eye on load failures and ensure the right recovery mechanisms are in place.
* ETL has historically been used for batch workloads on a large scale.
* However, with the emergence of streaming ETL tools, they are increasingly being used for real-time streaming event data as well

**Data Pipeline**

It’s common to see the terms ETL and data pipelines used interchangeably. And although both move data from source to destination, data pipeline is a broader term that encompasses the entire journey of moving data from one system to another, of which ETL is a subset.

Data pipelines can be architected for batch processing, for streaming data, and a combination of batch and streaming data. In the case of streaming data, data processing or transformation, happens in a continuous flow.

This is particularly useful for data that needs constant updating, such as data from a sensor monitoring traffic. A data pipeline is a high performing system that supports both long-running batch queries and smaller interactive queries. The destination for a data pipeline is typically a data lake, although the data may also be loaded to different target destinations, such as another application or a visualization tool.

There are a number of data pipeline solutions available, most popular among them being Apache Beam and DataFlow.

**Foundations of Big Data**

In this digital world, everyone leaves a trace. From our travel habits to our workouts and entertainment, the increasing number of internet connected devices that we interact with on a daily basis record vast amounts of data about us there's even a name for it Big Data.

Ernst and Young offers the following definition: big data refers to the dynamic, large, and disparate volumes of data being created by people, tools, and machines. It requires new, innovative and scalable technology to collect, host, and analytically process the vast amount of data gathered in order to drive real-time business insights that relate to consumers, risk, profit, performance, productivity management, and enhanced shareholder value

**5 v’s of Big Data**

There is no one definition of big data but there are certain elements that are common across the different definitions, such as velocity, volume, variety, veracity, and value. These are the V's of big data

* **Velocity** is the speed at which data accumulates. Data is being generated extremely fast in a process that never stops. Near or real-time streaming, local, and cloud-based technologies can process information very quickly.
* **Volume** is the scale of the data or the increase in the amount of data stored. Drivers of volume are the increase in data sources, higher resolution sensors, and scalable infrastructure.
* **Variety** is the diversity of the data. Structured data fits neatly into rows and columns in relational databases, while unstructured data is not organized in a predefined way like tweets, blog posts, pictures, numbers, and video. Variety also reflects that data comes from different sources; machines, people, and processes, both internal and external to organizations.

Drivers are mobile technologies social media, wearable technologies, geo technologies video, and many, many more.

* **Veracity** is the quality and origin of data and its conformity to facts and accuracy. Attributes include consistency, completeness, integrity, and ambiguity. Drivers include cost and the need for traceability. With the large amount of data available, the debate rages on about the accuracy of data in the digital age. Is the information real or is it false?
* **Value** is our ability and need to turn data into value. Value isn't just profit. It may have medical or social benefits, as well as customer, employee or personal satisfaction. The main reason that people invest time to understand big data

is to derive value from it.

**V’s in action**

Let's look at some examples of the V's in action.

* Velocity. Every 60 seconds, hours of footage are uploaded to YouTube, which is generating data. Think about how quickly data accumulates over hours, days, and years.
* Volume. The world population is approximately 7 billion people and the vast majority are now using digital devices. Mobile phones, desktop and laptop computers, wearable devices, and so on. These devices all generate, capture, and store data approximately 2.5 quintillion bytes every day. That's the equivalent of 10 million blu-ray DVDs.
* Variety. Let's think about the different types of data. Text, pictures, film, sound, health data from wearable devices, and many different types of data from devices connected to the internet of things.
* Veracity. Eighty percent of data is considered to be unstructured and we must devise ways to produce reliable and accurate insights. The data must be categorized, analyzed, and visualized. Data scientists, today, derive insights from big data and cope with the challenges that these massive data sets present. The scale of the data being collected means that it's not feasible to use conventional data analysis tools, however, alternative tools that leverage distributed computing power can overcome this problem. Tools such as Apache Spark, Hadoop, and its ecosystem provides ways to extract, load, analyze, and process the data across distributed compute resources, providing new insights and knowledge. This gives organizations more ways to connect with their customers and enrich the services they offer.

So next time you strap on your smartwatch, unlock your smartphone, or track your workout, remember your data is starting a journey that might take it all the way around the world, through big data analysis and back to you.

**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. In some of the other videos, we discussed Big Data technologies such as NoSQL databases and Data Lakes.

In this video, we are going to talk about three open source technologies and the role they play in big data analytics—Apache Hadoop, Apache Hive, and Apache Spark. Hadoop is a collection of tools that provides distributed storage and processing of big data.

Hive is a data warehouse for data query and analysis built on top of Hadoop. Spark is a distributed data analytics framework designed to perform complex data analytics in real-time

**Apache Hadoop**

Hadoop, a java-based open-source framework, allows distributed storage and processing of large datasets across clusters of computers. In Hadoop distributed system, a node is a single computer, and a collection of nodes forms a cluster.

Hadoop can scale up from a single node to any number of nodes, each offering local storage and computation. Hadoop provides a reliable, scalable, and cost-effective solution for storing data with no format requirements.

Using Hadoop, you can:

* Incorporate emerging data formats, such as streaming audio, video, social media sentiment, and clickstream data, along with structured, semi-structured, and unstructured data not traditionally used in a data warehouse.
* Provide real-time, self-service access for all stakeholders.
* Optimize and streamline costs in your enterprise data warehouse by consolidating data across the organization and moving “cold” data, that is, data that is not in frequent use, to a Hadoop-based system.

One of the four main components of Hadoop is Hadoop Distributed File System, or HDFS, which is a storage system for big data that runs on multiple commodity hardware connected through a network.

* HDFS provides scalable and reliable big data storage by partitioning files over multiple nodes.
* It splits large files across multiple computers, allowing parallel access to them.
* Computations can, therefore, run in parallel on each node where data is stored.
* It also replicates file blocks on different nodes to prevent data loss, making it fault-tolerant.

Let’s understand this through an example.

Consider a file that includes phone numbers for everyone in the United States; the numbers for people with last name starting with A might be stored on server 1, B on server 2, and so on.

With Hadoop, pieces of this phonebook would be stored across the cluster. 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. Some of the other 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 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.

Hadoop is intended for long sequential scans and, because Hive is based on Hadoop, queries have very high latency—which means Hive is less appropriate for applications that need very fast response times.

Hive is not suitable for transaction processing that typically involves a high percentage of write operations. Hive is better suited for data warehousing tasks such as ETL, reporting, and data analysis

and includes tools that enable easy access to data via SQL.

**Apache Spark**

This brings us to Spark, a general-purpose data processing engine designed to extract and process large volumes of data for a wide range of applications, including Interactive Analytics, Streams Processing, Machine Learning, Data Integration, and ETL.

It takes advantage of in-memory processing to significantly increase the speed of computations and spilling to disk only when memory is constrained. Spark has interfaces for major programming languages, including Java, Scala, Python, R, and SQL.

It can run using its standalone clustering technology as well as on top of other infrastructures such as Hadoop. And it can access data in a large variety of data sources, including HDFS and Hive, making it highly versatile. The ability to process streaming data fast and perform complex analytics in real-time is the key use case for Apache Spark.

**Identifying Data for Analysis**

At this stage, you have an understanding of the problem and the desired outcome—you know “Where you are” and “Where you want to be.“ You also have a well-defined metric—you know “What will be measured,” and “How it will be measured.”

The next step is for you is to identify the data you need for your use case. The process of identifying data begins by determining the information you want to collect. In this step, you make decisions regarding (a) the specific information you need; and (b) the possible sources for this data. Your goals determine the answers to these questions.

**Process for identifying data**

Let’s take the example of a product company that wants to create targeted marketing campaigns based on the age group that buys their products the most. Their goal is to design reach-outs that appeal most to this segment and encourages them to further influence their friends and peers into buying these products.

First step:

* Based on this use case, some of the obvious information that you will identify includes the customer profile, purchase history, location, age, education, profession, income, and marital status, for example. To ensure you gain even greater insights into this segment, you may also decide to collect the customer complaint data for this segment to understand the kind of issues they face because this could discourage them from recommending your products.

To know how satisfied they were with the resolution of their issues, you could collect the ratings from the customer service surveys. Taking this a step forward, you may want to understand how these customers talk about your products on social media and how many of their connections engage with them in these discussions, for example, the likes, shares, and comments their posts receive.

The next

* step in the process is to define a plan for collecting data. You need to establish a timeframe for collecting the data you have identified. Some of the data you need may be required on an ongoing basis and some over a defined period of time. For collecting website visitor data, for example, you may need to have the numbers refreshed in real-time. But if you’re tracking data for a specific event, you have a definite beginning and end date for collecting the data.

In this step, you can also define how much data would be sufficient for you to reach a credible analysis. Is the volume defined by the segment, for example, all customers within the age range of 21 to 30 years; or a dataset of a hundred thousand customers within the age range of 21 to 30. You can also use this step to define the dependencies, risks, mitigation plan, and several other such factors that are relevant to your initiative. The purpose of the plan should be to establish the clarity you need for execution.

The third step

* in the process is for you to determine your data collection methods. In this step, you will identify the methods for collecting the data you need. You will define how you will collect the data from the data sources you have identified, such as internal systems, social media sites, or third-party data providers. Your methods will depend on the type of data, the timeframe over which you need the data, and the volume of data.

Once your plan and data collection methods are finalized, you can implement your data collection strategy and start collecting data. You will be making updates to your plan as you go along because conditions evolve as you implement the plan on the ground.

**Key considerations**

The data you identify, the source of that data, and the practices you employ for gathering the data have implications for quality, security, and privacy. None of these are one-time considerations but are relevant through the life cycle of the data analysis process.

Working with data from disparate sources without considering how it measures against the quality metric can lead to failure. In order to be reliable, data needs to be free of errors, accurate, complete, relevant, and accessible

**Data Quality**

You need to define the quality traits, the metric, and the checkpoints in order to ensure that your analysis is going to be based on quality data. You also need to watch out for issues pertaining to data governance, such as, security, regulation, and compliances

**Data Governance**

Data Governance policies and procedures relate to the usability, integrity, and availability of data. Penalties for non-compliance can run into millions of dollars and can hurt the credibility of not just your findings, but also your organization. Another important consideration is data privacy.

**Data Privacy**

Data you collect needs to check the boxes for confidentiality, license for use, and compliance to mandated regulations. Checks, validations, and an auditable trail needs to be planned. Loss of trust in the data used for analysis can compromise the process, result in suspect findings, and invite penalties.

Identifying the right data is a very important step of the data analysis process. Done right, it will ensure that you are able to look at a problem from multiple perspectives and your findings are credible and reliable.

**Data Sources**

Data sources can be internal or external to the organization, and they can be primary, secondary or third party sources of data. Let's look at a couple of examples to understand what we mean by primary, secondary and 3rd party sources of data.

**Primary Data**

The term primary data refers to information obtained directly by you from the source. This could be from internal sources such as data from the organization, CRM, HR or workflow applications. It could also include data you gather directly through surveys, interviews, discussions, observations and focus groups.

**Secondary Data**

Secondary data refers to information retrieved from existing sources, such as external databases, research articles, publications, training material and Internet searches, or financial records available as public data. This could also include data collected through externally conducted surveys, interviews, discussions, observations and focus groups

**Third-party Data**

Third party data is data you purchased from aggregators who collect data from various sources and combine it into comprehensive datasets purely for the purpose of selling the data. Now will look at some of the different sources from which you could be gathering data.

Databases can be a source of primary, secondary and 3rd party data. Most organizations have internal applications for managing their processes, workflows and customers. External databases are available on a subscription basis or for purchase.

**Sources For Gathering Data**

A significant number of businesses have or are currently moving to the cloud, which is increasingly becoming a source for accessing real time information and on demand insights.

* **The Web** is a source of publicly available data that is available to companies. And individuals for free or commercial use. The Web is a rich source of data available in the public domain. These could include textbooks, government records, papers, and articles that are for public consumption, social media sites, and interactive platforms such as Facebook, Twitter, Google, YouTube.

An Instagram are increasingly being used to source user data and opinions. Businesses are using these data sources for quantitative and qualitative insights. An existing and potential customer.

* Sensor data produced by wearable devices, smart buildings, smart cities, smart phones, medical devices, even household appliances is a widely used source of data.
* Data exchange is a source of 3rd party data that involves the voluntary sharing of data between data providers and data consumers, individuals, organizations and governments could be both data providers and data consumers. The data that is exchanged could include data coming from business applications, sensor devices, social media activity, location data, or consumer behavior data.
* Surveys gather information through questionnaires distributed to a select group of people. For example, gauging the interest of existing customers in spending on an updated version of a product. Surveys can be web or paper based.
* Census data is also a commonly used source for gathering household data, such as wealth and income or population data, for example.
* Interviews are source for gathering qualitative data, such as the participants opinions and experiences. For example, an interview conducted to understand the day-to-day challenges faced by a customer service executive. Interviews could be telephonic over the Web or face to face observation.
* Studies include monitoring participants in a specific environment or while performing a particular task. For example, observing users navigate an E Commerce site to assess them.

Ease with which they are able to find products and make a purchase data from surveys, interviews, an observation. Studies could be available as primary, secondary and 3rd party data. Data sources have never been as dynamic and diverse as they are today. They are also evolving continuously.

Supplementing your primary data with secondary and 3rd party data sources can help you explore problems and solutions in new and meaningful ways.

**How to Gather and Import Data**

In this video, we will learn about the different methods and tools available for gathering data from the data sources discussed earlier in the course—such as databases, the web, sensor data, data exchanges, and several other sources leveraged for specific data needs. We will also learn about importing data into different types of data repositories.

**Using Queries to Extract data from databases**

SQL, or Structured Query Language, is a querying language used for extracting information from relational databases. SQL offers simple commands to specify what is to be retrieved from the database, the table from which it needs to be extracted, grouping records with matching values, dictating the sequence in which the query results are displayed, and limiting the number of results that can be returned by the query, amongst a host of other features and functionalities. Non-relational databases can be queried using SQL or SQL-like query tools. Some non-relational databases come with their own querying tools such as CQL for Cassandra and GraphQL for Neo4J.

**Using APIs**

Application Programming Interfaces (or APIs) are also popularly used for extracting data from a variety of data sources. APIs are invoked from applications that require the data and access an end-point containing the data.

End-points can include databases, web services, and data marketplaces. APIs are also used for data validation. For example, a data analyst may utilize an API to validate postal addresses and zip codes.

**Web Scrapping**

Web scraping, also known as screen scraping or web harvesting, is used for downloading specific data from web pages based on defined parameters. Among other things, web scraping is used to extract data such as text, contact information, images, videos, podcasts, and product items from a web property.

RSS feeds are another source typically used for capturing updated data from online forums and news sites where data is refreshed on an ongoing basis.

**Sensor data**

Data streams are a popular source for aggregating constant streams of data flowing from sources such as instruments, IoT devices and applications, and GPS data from cars. Data streams and feeds are also used for extracting data from social media sites and interactive platforms.

Data Exchange platforms allow the exchange of data between data providers and data consumers.

**Data Exchange Platforms**

Data Exchanges have a set of well-defined exchange standards, protocols, and formats relevant for exchanging data. These platforms not only facilitate the exchange of data, they also ensure that security and governance are maintained.

They provide data licensing workflows, de-identification and protection of personal information, legal frameworks, and a quarantined analytics environment. Examples of popular data exchange platforms include AWS Data Exchange, Crunchbase, Lotame, and Snowflake

**Other sources**

Numerous other data sources can be tapped into for specific data needs. For marketing trends and ad spending, for example, research firms like Forrester and Business Insider are known to provide reliable data.

Research and advisory firms such as Gartner and Forrester are widely trusted sources for strategic and operational guidance. Similarly, there are many trusted names in the areas of user behavior data, mobile and web usage, market surveys, and demographic studies

**Imported Data**

Data that has been identified and gathered from the various data sources now needs to be loaded or imported into a data repository before it can be wrangled, mined, and analyzed. The importing process involves combining data from different sources to provide a combined view and a single interface using which you can query and manipulate the data.

**Data types and destinations repositories**

Depending on the data type, the volume of data, and the type of destination repository, you may need varying tools and methods. Specific data repositories are optimized for certain types of data.

Relational databases store structured data with a well-defined schema. If you’re using a relational database as the destination system, you will only be able to store structured data, such as data from OLTP systems, spreadsheets, online forms, sensors, network and web logs.

Structured data can also be stored in NoSQL. Semi-structured data is data that has some organizational properties but not a rigid schema, such as, data from emails, XML, zipped files, binary executables, and TCP/IP protocols. Semi-structured can be stored in NoSQL clusters.

XML and JSON are commonly used for storing and exchanging semi-structured data. JSON is also the preferred data type for web services. Unstructured data is data that does not have a structure and cannot be organized into a schema, such as data from web pages, social media feeds, images, videos, documents, media logs, and surveys.

NoSQL databases and Data Lakes provide a good option to store and manipulate large volumes of unstructured data. Data lakes can accommodate all data types and schema. ETL tools and data pipelines provide automated functions that facilitate the process of importing data. Tools such as Talend and Informatica, and programming languages such as Python and R, and their libraries, are widely used for importing data.

**What is Data Wrangling?**

Data wrangling, also known as data munging, is an iterative process that involves data exploration, transformation, validation, and making it available for a credible and meaningful analysis. It includes a range of tasks involved in preparing raw data for a clearly defined purpose, where raw data at this stage is data that has been collated through various data sources in a data repository.

Data wrangling captures a range of tasks involved in preparing data for analysis. Typically, it is a 4-step process that involves—Discovery, Transformation, Validation, and Publishing.

**The Discovery (Exploration Phase)**

The Discovery phase, also known as the Exploration phase, is about understanding your data better with respect to your use case. The objective is to figure out specifically how best you can clean, structure, organize, and map the data you have for your use case.

The next phase, which is the Transformation phase, forms the bulk of the data wrangling process. It involves the tasks you undertake to transform the data, such as structuring, normalizing, denormalizing, cleaning, and enriching the data.

**Transformation Phase**

Let’s begin with the first transformation task:

* **Structuring.** This task includes actions that change the form and schema of your data. The incoming data can be in varied formats. You might, for example, have some data coming from a relational database and some data from Web APIs.

In order to merge them, you will need to change the form or schema of your data. This change may be as simple as changing the order of fields within a record or dataset or as complex as combining fields into complex structures.

**Joins and Unions** are the most common structural transformations used to combine data from one or more tables.

How they combine the data is different. Joins combine columns.

When two tables are joined together, columns from the first source table are combined with columns from the second source table—in the same row.

So, each row in the resultant table contains columns from both tables.

Unions combine rows. Rows of data from the first source table are combined with rows of data from the second source table into a single table.

Each row in the resultant table is from one source table or another.

Transformation can also include

* normalization and denormalization of data.

**Normalization** focuses on cleaning the database of unused data and reducing redundancy and inconsistency. Data coming from transactional systems, for example, where a number of insert, update, and delete operations are performed on an ongoing basis, are highly normalized.

**Denormalization** is used to combine data from multiple tables into a single table so that it can be queried faster. For example, normalized data coming from transactional systems is typically denormalized before running queries for reporting and analysis.

Another transformation type is Cleaning. Cleaning tasks are actions that fix irregularities in data in order to produce a credible and accurate analysis. Data that is inaccurate, missing, or incomplete can skew the results of your analysis and need to be considered.

It could also be that the data is biased, or has null values in relevant fields, or have outliers. For example, you may want to find out the demographic information on the sale of a certain product, but the data you have received does not capture the gender. You either need to source this data point and merge it with your existing dataset, or you may need to remove, and not consider the records with this field missing. We will explore many more examples of data cleaning further on in the course.

* **Enriching the data**—is the fourth type of transformation. When you consider the data you have, to look at additional data points that could make your analysis more meaningful, you are looking at enriching your data. For example, in a large organization with information fragmented across systems, you may need to enrich the dataset provided by one system with information available in other systems, or even public datasets.

Consider a scenario where you sell IT peripherals to businesses and want to analyze the buying patterns of your customers over the last five years. You have the customer master and transaction tables from where you’ve captured the customer information and purchase history.

Supplementing your dataset with the performance data of these businesses, possibly available as a public dataset, could be valuable for you to understand factors influencing their purchase decisions.

**Inserting metadata** also enriches data. For example, computing a sentiment score from a customer feedback log, collecting geo-based weather data from a resorts location to analyze occupancy trends, or capturing published time and tags for a blog post.

**Validation**

After transformation, the next phase in Data Wrangling is Validation. This is where you check the quality of the data post structuring, normalizing, cleaning, and enriching.

**Validation rules** refer to repetitive programming steps used to verify the consistency, quality, and security of the data you have. This brings us to Publishing—the fourth phase of the data wrangling process.

**Publishing** involves delivering the output of the wrangled data for downstream project needs. What is published is the transformed and validated version of the input dataset along with the metadata about the dataset

**Publishing**

Lastly, it is important to note the criticality of documenting the steps and considerations you have taken to convert the raw data to analysis-ready data. All phases of data wrangling are iterative in nature. In order to replicate the steps and to revisit your considerations for performing these steps, it is vital that you document all considerations and actions.

**Tools For Data Wrangling**

In this video, we will look at some of the popularly used data wrangling software and tools, such as:

* Excel Power Query / Spreadsheets,
* OpenRefine
* Google DataPrep
* Watson Studio Refinery
* Trifacta Wrangler
* Python
* R

**Excel Power Query (Spreadsheet)**

Let’s begin with the most basic software used for manual wrangling—Spreadsheets. Spreadsheets such as Microsoft Excel and Google Sheets have a host of features and in-built formulae that can help you identify issues, clean, and transform data.

Add-ins are available that allow you to import data from several different types of sources and clean and transform data as needed—such as Microsoft Power Query for Excel and Google Sheets Query function for Google Sheets.

**OpenRefine**

OpenRefine is an open-source tool that allows you to import and export data in a wide variety of formats, such as TSV, CSV, XLS, XML, and JSON. Using OpenRefine, you can clean data, transform it from one format to another, and extend data with web services and external data. OpenRefine is easy to learn and easy to use. It offers menu-based operations, which means you don’t need to memorize commands or syntax.

**Google DataPrep**

Google DataPrep is an intelligent cloud data service that allows you to visually explore, clean, and prepare both structured and unstructured data for analysis.

* It is a fully managed service, which means you don’t need to install or manage the software or the infrastructure.
* DataPrep is extremely easy to use. With every action that you take, you get suggestions on what your ideal next step should be.
* DataPrep can automatically detect schemas, data types, and anomalies.

**Watson Studio Refinery**

Watson Studio Refinery, available via IBM Watson Studio, allows you to discover, cleanse, and transform data with built-in operations.

* It transforms large amounts of raw data into consumable, quality information that’s ready for analytics.
* Data Refinery offers the flexibility of exploring data residing in a spectrum of data sources.
* It detects data types and classifications automatically and also enforces applicable data governance policies automatically.

**Trifacta Wrangler**

Trifacta Wrangler is an interactive cloud-based service for cleaning and transforming data.

* It takes messy, real-world data and cleans and rearranges it into data tables, which can then be exported to Excel, Tableau, and R. It is known for its collaboration features, allowing multiple team members to work simultaneously.

**Python**

Python has a huge library and set of packages that offer powerful data manipulation capabilities.

Let’s look at a few of these libraries and packages.

* Jupyter Notebook is an open-source web application widely used for data cleaning and transformation, statistical modeling, also data visualization.
* Numpy, or Numerical Python, is the most basic package that Python offers.
* It is fast, versatile, interoperable, and easy to use.
* It provides support for large, multi-dimensional arrays and matrices, and high-level mathematical functions to operate on these arrays.
* Pandas is designed for fast and easy data analysis operations.
* It allows complex operations such as merging, joining, and transforming huge chunks of data, performed using simple, single-line commands.
* Using Pandas, you can prevent common errors that result from misaligned data coming in from different sources

**R – Programming**

R, also offers a series of libraries and packages that are explicitly created for wrangling messy data—such as Dplyr, Data.table, and Jsonlite.

* Using these libraries, you can investigate, manipulate, and analyze data.
* Dplyr is a powerful library for data wrangling.
* It has a precise and straightforward syntax.
* Data.table helps to aggregate large data sets quickly.
* Jsonlite is a robust JSON parsing tool, great for interacting with web APIs.

Tools for data wrangling come with varying capabilities and dimensions. Your decision regarding the best tool for your needs will depend on factors that are specific to your use case, infrastructure, and teams—such as supported data size, data structures, cleaning and transformation capabilities, infrastructure needs, ease of use, and learnability.

**Data Cleaning**

**Quality of Data**

According to a Gartner report on data quality:

* poor quality data weakens an organization's competitive standing and undermines critical business objectives.
* Missing, inconsistent, or incorrect data can lead to false conclusions and therefore ineffective decisions. And in the business world, that can be costly.
* Data sets picked up from disparate sources could have a number of issues, including missing values, inaccuracies, duplicates, incorrect or missing delimiters, inconsistent records, and insufficient parameters.
* In some cases, data can be corrected manually or automatically with the help of data wrangling tools and scripts, but if it cannot be repaired, it must be removed from the dataset.

**Data Wrangling Process**

Although the terms Data Cleaning and Data Wrangling are sometimes used interchangeably, it is important to keep in mind that data cleaning is only a subset of the entire Data Wrangling process.

Data Cleaning forms a very significant and integral part of the Transformation phase in a data wrangling workflow

**Data Cleaning Workflow**

A typical data cleaning workflow includes:

* Inspection
* Cleaning, and
* Verification.

The first step in the data cleaning workflow is to detect the different types of issues and errors that your dataset may have.

* You can use scripts and tools that allow you to define specific rules and constraints and
* validate your data against these rules and constraints.
* You can also use data profiling and data visualization tools for inspection.
* Data profiling helps you to inspect the source data to understand the structure, content, and interrelationships in your data.
* It uncovers anomalies and data quality issues. For example, blank or null values, duplicate data, or whether the value of a field falls within the expected range.
* Visualizing the data using statistical methods can help you to spot outliers. For example, plotting the average income in a demographic dataset can help you spot outliers.

That brings us to the actual cleaning of the data. The techniques you apply for cleaning your dataset will depend on your use case and the type of issues you encounter. Let’s look at some of the more common data issues.

Let’s start with **missing values**.

* Missing values are very important to deal with as they can cause unexpected or biased results.
  + You can choose to filter out the records with missing values or
  + find a way to source that information in case it is intrinsic to your use case. For example, missing age data from a demographics study.
  + A third option is a method known as imputation, which calculates the missing value based on statistical values.

Your decision on the course of action you choose needs to be anchored in what’s best for your use case.

You may also come across **duplicate data**, data points that are repeated in your dataset. These need to be removed.

Another type of issue you may encounter is that of irrelevant data. Data that does not fit within the context of your use case can be considered irrelevant data. For example, if you are analyzing data about the general health of a segment of the population, their contact numbers may not be relevant for you.

Cleaning can involve **data type conversion** as well.

* This is needed to ensure that values in a field are stored as the data type of that field—for example, numbers stored as numerical data type or date stored as a date data type.
* You may also need to clean your data in order to standardize it. For example, for strings, you may want all values to be in lower case.
* Similarly, date formats and units of measurement need to be standardized.

Then there are **syntax errors**.

* For example, white spaces, or extra spaces at the beginning or end of a string is a syntax error that needs to be rectified.
* This can also include fixing typos or format, for example, the state name being entered as a full form such as New York versus an abbreviated form such as NY in some records.

Data can also have **outlier**s, or values that are vastly different from other observations in the dataset. Outliers may, or may not, be incorrect. For example, when an age field in a voters database has the value 5, you know it is incorrect data and needs to be corrected.

* Now let’s consider a group of people where the annual income is in the range of one hundred thousand to two hundred thousand dollars—except for that one person who earns a million dollars a year. While this data point is not incorrect, it is an outlier, and needs to be looked at.

Depending on your use case, you may need to decide if including this data will skew the results in a way that does not serve your use case.

This brings us to the next step in the data cleaning workflow—**Verification**.

* In this step, you inspect the results to establish effectiveness and accuracy achieved as a result of the data cleaning operation.
* You need to re-inspect the data to make sure the rules and constraints applicable on the data still hold after the corrections you made.

And in the end, it is important to note that all changes undertaken as part of the data cleaning operation need to be documented. Not just the changes, but also the reasons behind making those changes, and the quality of the currently stored data. Reporting how healthy the data is, is a very crucial step.

**Overview of Statistical Analysis**

Statistical Analysis

Descriptive Statistics

Inferential Statistics