1. Introduction to Data Engineering

- NoSQL
  - Key-Value Store
  - Document-based
  - Column-based
  - Graph-based
- Data Warehouses, data marts and data lakes
  - Data warehouse
  - Data mart
  - Data lake
- ETL, ELT, and Data Pipelines
  - ETL: Extract Transform and Load
  - ELT: Extract Load and Transform
  - Data pipeline
- Data integration platforms
- LAB: Provision an instance of IBM Db2 Lite plan
- Foundations of Big Data
  - The 5 V's:
- Big data processing tools
  - Apache Hadoop
  - Apache Hive
  - Apache Spark

Semana 3

- Architecting the Data Platform (5 layers)
  - Data ingestion or data collection layer
  - Data storage and integration layer
  - Data processing layer
  - Analysis and user interface layer
  - Data pipeline layer
- Factors for selecting and designing data stores
  - **A repository can be:**
  - **Intended use of data:**
  - **Storage considerations:**
  - Privacy, security and governance
- Security
  - The CIA Triad
  - Physical Infrastructure Security
  - Network Security
  - Application Security
  - Data Security
  - Monitoring and Intelligence
- How to gather and import data
  - SQL
  - APIs
  - Web scraping
  - Data streams
  - Data exchange
  - Importing data
- Data wrangling or munging
  - Transformation
  - Tools for data wrangling
- Lab: Load data into the IBM Db2 database from a CSV file
- Querying and analyzing data → SQL
  - Count
  - Aggregation
  - Extreme value identification
  - Slicing data
  - Sorting data
  - Filtering patterns
  - Grouping data
- Performance tuning and troubleshooting
  - Data pipelines
  - Data base optimization for performance
  - Monitoring systems

Semana 1

Key players in the data ecosystem

Data Engineer

- Develop and mantain data architectures and provide data to business analysts
- Extract, integrate and organize data from disparate sources
- Clean, transform and prepare data
- Design, store and manage data in data repositories

Data Analyst

- Translates data and numbers into plain language so organizations can make desitions
- Inspect and clean data
- Identify correlations, find patterns, apply statistical methods
- Visualize data to interpret and present
- They answer the questions that can be answered from the data

Data Scientist

- Analyze data for actionable insights
- Create ML and Deep learning models
- They answer predictive questions, like how many followers am i likely to get next month?

Business Analyst and BI Analyst

- Thet look at possible applications for their business
- Organize and monitor data
- Explore data to extract insight

What is Data Engineering?

WHAT'S DONE:

- Collecting source data
- Proccesing data
- Storing data
- Making data available to users securely

— — —

We dont play as much time playing with the data as organizing it. Data Analysts and Data Scientist use the data that comes from data engineering.
The goal of Data Engineering is to make quality data available for analytics and decision-making. And it does this by collecting raw source data, processing data so it becomes usable, storing data, and making quality data available to users securely.

Responsibilities and skillsets of a Data Engineer

Data is analytics-ready when it is:

- accurate
- reliable
- complies to regulations
- accesible to consumers when they need it

Technical skills:

- operating systems
- infrastructure components: virtual machines, networking, application services, cloud-based services
- database and data warehouses: RDBMS, NoSQL, Data Warehouses
    - Datawarehouses: Oracle Exdata, IBM Db2 Warehouse on Cloud, IBM Netezza Performance Server, Amazon RedSchif

Data pipelines:

- apache beam
- airflow
- dataflow

ETL tools (Extract, Transform, Load):

- IBM Infosphere
- AWS
- improvado

Languages:

- query languages
- programming languages
- shell and scripting languages

Big Data processing tools:

- hadoop
- hive
- apache spark

Functional skills:

- convert business requirements into technical specifications
- work with the complete software development lifecycle: **ideation** → **architecture** → **design** → **prototyping** → **testing** → **deployment** → **monitoring**
- understand risks of poor data management

Cuenta en IBM Cloud

—

Semana 2

Overview of the data engineering ecosystem

Infrastrucuture, tools, frameworks and processes.

Data

- Structured
  - Follows a rigid format and can be organized in rows and columns
- Semi-structure
  - Mix of data, example: E-mails. Doesnt respond to a rigid structure
- Unstructured
  - Complex, qualitive information, imposible to reduce to rows and columns. Example, photos, social media posts

Data can be collected from:

- relational database
- non-relational database
- apis
- web services
- data sterams
- social platforms
- sensor devices

Data Repositories

- transactional or OLTP (online transactional processing system)
  - design to store high volume day-to-day operational data
    - ex: banking transactions, airline bookings
    - can be relational or non-relational
- analytical or OLAP (online analytical processing system)
  - optimized for conducting complex data analytics
    - include relational and non-relational databases, data warehouses, data marts, data lakes and big data stores

Data Integration

Collated → Processed → Cleansed → Integrated → Access for users

Data Pipelines

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

Uses ETL or ELT process.

Languages

- query
  - example: SQL for querying and manipulating data
- programming
  - example: python, for developing data applications
- shell and scripting
  - for repetitive operational tasks

BI and Reporting Tools

Collect data from multiple data sources and present them in a visual format, such as interactive dashboards

Visualize data in real-time

Drag and drop products that do not requiere prog. knowledge

Types of data

- facts, observations, perceptions
- numbers, characters, symbols
- images

—

- structured data
  - facts and numbers
  - collected → exported → stored → organized
  - sources: SQL databases, OLTP Systems, spreadsheets, online forms, sensors, network and web server logs
- semi-structured data
  - contains tags or metadata which are used to group data
    - ex: e-mails, xml, binary executables, TCP/IP packets, zipped files, integration of data
    - xml and json allow users to define tags and attributes
- unstructured data
  - no rules
  - sources: web pages, social media feeds, imagines in varied file formats, video and audio files, documents and pdf files, pppt, media logs, surveys
  - stored in files and docs for manual analysis or in NoSQL with analysis tools

Different types of File Formats

Standard file formats

1. Delimited text file formats, or CSV
2. Excel Open .XML Spreadsheet or XLSX
3. Extensible markup language or XML
4. Portable document format or PDF
5. Javascript object notaion or JSON

CSV, TSV

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.

XLSX

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, 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.

PDF

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 forms.

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.**

Sources of data

Relational databases

- can be used as source for analysis

APIS

Examples:

- Twitter and facebook APIs for customer sentiment analysis
- Stock market APIs for trading and analysis
- Data lookup and validation APIs for cleaning and co-relating data

Web scraping

- extract relevant data from unstructured sources
- sinonimos: screen scraping, web harvesting, web data extraction
- downloads specific data based on defined parameters
- can extract text, conttact information, images ...
- popular tools:
    - beautiful soup
    - scrapy
    - pandas
    - selenium

Data streams and feeds

- agreggating streams of data flowing from instruments, iot devices, gps data from cars ...
- usually time stamped and with geolocation

Languages for data professionals

Query languages

→ designed for accesing and manipulating data in a database (sql)

SQL

- advantages:
  - portable and platform independent
  - can be used for querying in a wide variety of databases and repositories
  - simple syntax similar to english
  - its syntax allows developers to write programs with fewer lines of code using basic keywords
  - retrieve large amounts of data quickly and efficiently
  - runs on interpreter system

Programming languages

→ designed for deveoping applications and controlling application behavior (python, R, java)

Python

- express concepts in few lines of code
- easy to learn and large comunity
- great for perfoming high-computational tasks and large volumes of data
- object oriented, imperative, functional, procedural
- multi platform
- libraries:
  - pandas for data cleansing and analysis
  - numpy and scipy, for statistical analysis
  - beautifulSoup and scrapy for web scraping
  - matplotlib and seaborn to visually represent data
  - opency for image processing

R

- open-source programming language and enviroment for data analysis, data visualization, machine learning and statistics
- known for creating compeling visualizations
- platform independent
- highly extensible
- structure and unstructured data
- ggplot2 and plotly for aesthetic graphical plots
- allows data and script embedded in reports
- allows creation interactive web apps

Java

- platform independent
- data analysis, cleaning importing and exporting data, statistical analysis, data visualization
- used in the development of big data frameworks like hadoop

- well-suited for speed-crititcal projects

Shell scripting

→ ideal for repetitive and time-consuming operational tasks (unix/linux shell, power shell)

Unix/Linux shell

- series of unix commanfs written in a plain text file to accomplish a specific task
- file maniupulation
- program execution
- system admin tasks
- installation scripts
- executing routine backups
- running batches

PoweShell

- microsoft
- optimized for structured data formats such as json, csv, xml and rest apis, websites and office apps
- cross-platform automation tool and configuration framework
- object-based → filter, sort, measure, group and compare objects as they pass through a data pipeline
- data mining, building GUIs, creating charts, dashboards and interactive reports

Overview of data repositories

→ A 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.

Types of data repositories include:

- databases
- data warehouses
- big data stores

Data bases

→ Collection of data for input, storage, search, retrieval and modification of data

- relational
  - flat files
  - data organized into tabular format
  - well-defined structure and schema
  - optimized for data operations and querying
  - sql
- non-relational
  - nosql
  - volume diversity
  - speed, flexibility and scale
  - store data in a schema-less form
  - big data

Data warehoouse

→ 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.

- data marts
  - relational
- data lakes

Big data stores

→ Distributed computational and storage infrastructure to store, scale and process very large data sets

RDBMS

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”.

(Como en genexus :D)

Relationship between tables minimizes data redundancy

Create meaningful information by **joining** tables

ACID compliant: Atomicity, Consistency, Isolation, and Durability

Use cases:

- oltp applications
- data warehouses (olap)
- iot solutions

Limitaitions:

- semi-structured and unstructured
- migration between two RDBMS (source and destination must have identical schemas)
- entering a value greater than the defined length

NoSQL

→ non-relational database design that provides flexible schemas for the storage and retrieval of data

- scale, performance, ease of use
- NO = Not Only SQL
- allows data to be stored in a schema-less or free-form fashion
- handle large volumes
- ability to run as distributed system scaled accross multiple data center
- efficient and cost-effective scale-out architecture that provides additional capacity and performance with the addition of new nodes
- NOT ACID compliant

Four types of NoSQL databases:

- key-value store
- document based
- column based
- graph based

Key-Value Store

- the key represents an attribute of the data and is a unique indentifier
- both keys and values can be anything from int to json
- great for storing user session data, real-time recommendations...

- NOT GREAT:
  - query data on specific data value
  - need relationships between data values
  - need multiple unique keys
- Examples:
  - redis
  - memcached
  - dynamoDB

Document-based

- store each record within a single document
- flexible indexing, powerful ad hoc queries, analytics
- great for eCommerce platforms, medical records storage, CRM platforms, analytics platforms
- NOT GREAT;
  - complex search queries
  - multi-operation transactions
- Examples:
  - mongoDB
  - DocumentDB
  - CouchDB
  - Cloudant

Column-based

- stored in cells grouped as columns of data instead of rows
- logical grouping of columns = columns family
- all cells corresponding to a column are saved as a continuos disk entry, making access and search easier and faster
- great for systems that requiere heavy write requests, storing time-series data, weather data and iot data
- NOT GREAT:
  - complex queries
  - change querying patterns frequently
- Examples:
  - cassandra
  - hbase

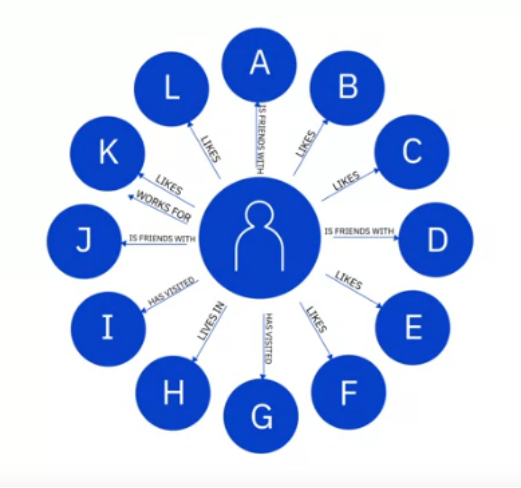
Graph-based

- use a graphical model to represent and store data
- useful for visualizing, analyzing and finding connections between different pieces of data

Nodes = data

Arrows = relationships

- great for social networks, product recommendations, network diagrams, fraud detection, access management
- NOT GREAT:



- process high volumes of transactions
- Examples:
  - neo4j
  - cosmosDB

Data Warehouses, data marts and data lakes

Data warehouse

→ Central repository of data integrated from multiple sources

- storing current and historical data
- when data gets loaded it is already modeled, structured and analysis-ready
- relational data from transactional systems and operational databases
  - crm
  - hr
  - erp
- non-relational
- 3 tier architecture
  - Client front-end layer (querying, reporting and analyzing data)
  - OLAP Server (process and analyze information coming from database servers)
  - Database Servers (extracting data from different sources)
- Benefits of cloud-based warehouses:
  - lower costs
  - limitless storage and compute capabilities
  - scale on a pay-as-you-go basis
  - faster disaster recovery
- examples:
  - teradata
  - oracle exadata
  - ibm db2
  - netezza
  - amazon redshift
  - google bigquery
  - cloudera
  - snowflake

Data mart

→ sub-section of the data warehouse, built specifically for a particular business function, purpose or community of users

- Three types:
  - Dependent
    - subsection of an enterprise data warehouse where data has already been cleaned and transformed
    - offers analytical capabilities for a restricted area of a data warehouse
    - isloated security and performance
  - Independent
    - created from sources other than an enterprise data warehouse, such as internal operational systems and external systems
    - need to carry out the transformation on the source data
  - Hybrid
    - combine inputs from data warehouses, operational systems and external systems

Data lake

→ store large amounts of structured, semi-structured and unstructured data in their native format

- data can be loaded without defining structure and schema
- exists as a repository of raw data
- data is classified
- reference arquitecture independent of technology
- can be deployed using cloud object storage or large-scale distributed systems and relational and non-relational database systems
- ability to store all types of data
- agility to scale based on storage capacity
- ability to repurpose data in several different ways and wide-ranging use cases
- Examples:
  - amazon
  - cloudera
  - google
  - ibm
  - informatica
  - microsoft
  - oracle

ETL, ELT, and Data Pipelines

Tools and processes that work to move data from source to destintation systems

ETL: Extract Transform and Load

- raw data is converted to analysis-ready data
- Steps:
  - gathering raw data
  - extracting information needed for reporting and analysis
  - cleaning, standardizing, and transforming data into usable format
  - loading data into a data repository

Extract (staging area) → transform and load (data repository) → analytics

Extraction can be through:

- batch processing: large chunks of data moved from source to destination at scheduled intervals
  - tools: stitch, blendo
- stream processing: data pulled in real-time from source, transformed in transit and loaded into data repository
  - tools: apache samza, apache storm, apache kafka

Transforming data:

- standarizing date formats and units of measurements
- removing duplicate data
- filtering out data that is not required
- enriching data
- establishing key relationship across tables
- applying business rules and data validations

Loading data:

- transportation of processed data in to a data repository
- Inital loading: populating all of the data in the repository
- Incremental loading: applying updates and modifications periodically
- Full-refresh: erasing a data table and reloading with fresh table
- Load verification:
  - missing or null values
  - server performance
  - load failures

ELT: Extract Load and Transform

- In the ELT process, extracted data is first loaded into the target system, and transformations are applied in the target system.
- The destination system for an ELT pipeline is most likely a data lake, though it can also be a data warehouse.
- ELT is a relatively new technology powered by cloud technologies.
- ELT is useful for processing large sets of unstructured and non-relational data. It is ideal for data lakes where transformations on the data are applied once the raw data is loaded into the data lake.
- Advantages:
  - shortens the cycle between extraction and delivery
  - ingest volumes of raw data when it becomes available
  - greater flexibility for exploratory data nalytics
  - transforms only that data whiich is required for a particular analysis
  - more suited to work with big data

Data pipeline

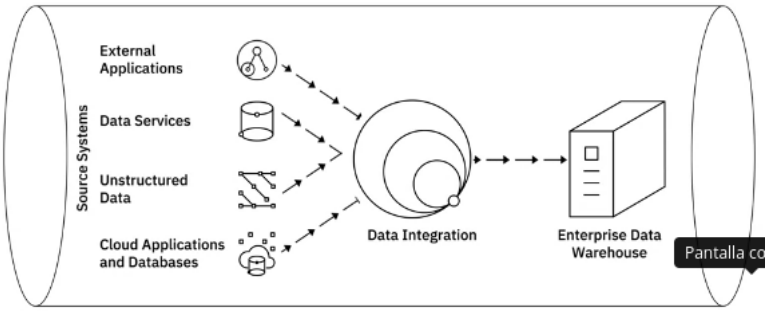
- Encompasses the entire journey of moving data from one system to another, including the ETL process
- can be used for batch and streaming data
- typically loads into a data lake

Data integration platforms

Gartner defines data integration as a discipline comprising the practices, architectural techniques, and tools that allow organizations to ingest, transform, combine, and provision data across various data types.

In the field of data science:

- accessing, queueing or extracting data from operational systems
- transforming and merging extracted data either logically or phisically
- data quality and governance
- delivering data through and integrated approach for analytics purposes



TOOLS:

IBM offers a host of data integration tools targeting a range of enterprise integration scenarios:

- Information Server for IBM,
- Cloud Pak for Data,
- IBM Cloud Pak for Integration,
- IBM Data Replication,
- IBM Data Virtualization Manager,
- IBM InfoSphere Information Server on Cloud, and
- IBM InfoSphere DataStage all target a range of enterprise data integration scenarios.

iPaaS = Cloud-based Integration Platform as a Service

LAB: Provision an instance of IBM Db2 Lite plan

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/7f635a7e-3fdd-422a-8744-4f0e26240b21/1.2.2.9_Hands-on_Lab_Provision_an_instance_of_IBM_Db2_Lite_plan.md.html

Foundations of 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 derive real-time business insights that relate to consumers, risk, profit, performance, productivity management, and enhanced shareholder value.” 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.

The 5 V's:

Velocity at which data accumulates

Volume is the scale of the data

Variety is the diversity of the data

Veracity is the quality and origin of the data

Value is our ability to turn data into Value.

Big data processing tools

These technologies are open source

Apache Hadoop

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

- storage and processing of large datasets across clusters of computers
- node = 1 computer
- cluster = a collection of computers
- reliable, scalable and cost-effective solution for storing data with no format requirements
- better real-time data-driven decisions
- improved data access and analysis
- data offload and consolidation (move the computation closer to the node in which the data resides)

HDFS:

- Hadoop Distributed File System: is a storage system for big data that runs on multiple commodity hardware connected through a network.
- Partitioning files over multiple nodes
- Splits large files accross multiple computers, allowing parallel acces to them
- Replicates file blocks on different nodes to prevent file loss
- Higher availability
- Better scalability
- Data locality
- Fast recovery from hardware failures
- Access to straming data
- Accomodation of large data sets
- Portability

Apache Hive

is a data warehouse for data query and analysis built on top og Hadoop. This software is used for reading, writing, and managing large data set files that are stored directly in either HDFS or other data storage systems such as Apache HBase

- Queries have high latency → less appropiate for apps that need very fast response times
- Read-based → not suitable for transaction processing that involves write operations
- Better used for: ETL, reporting and data analysis
- Easy access via SQL

Apache Spark

is a distributed analytics framework for complex, real-time data analytics. This general-purpose data processing engine is designed to extract and process large volumes of data for a wide range of applications

- Applications include:

    - interactive analysis
    - streams processing
    - machine learning
    - data integration
    - ETL
- in-memory processing → increases speed of computations
- interfaces for major programming languages such as java, python, R and SQL
- runs using its standalone clustering tech
- it can run on top of other infrastructures, such as hadoop
- can access data in a large varierty of data sources, including HDFS and Hive
- process streaming data fast
- **performs complex analytics in real-time**

Semana 3

Architecting the Data Platform (5 layers)

Data ingestion or data collection layer

Responsible for connecting to the source systems and bringing the data from there systems into the data platform.

- Connect to data sources
- Transport data (batch and streaming mode)
- Mantain information about the data collected in the metadata repository

Tools:

- Data flow, IBM Streams, IBM Streaming Analytics on Cloud, amazon Kinesis, Kafka

Data storage and integration layer

- Store data for processing and long-term use
- Transform and merge extracted data, either logically or phisically
- Make data available for processing in both streaming and batch modes
- Needs to be:
    - reliable, scalable, high-performing, and also cost-efficient

Tools (databases):

- relational databases: Db2 IBM, Microsoft SQL Server, MySQL, Oracle, PostgreSQL
- also database as a service (cloud)
- non-relational databases like mongoDB and casssandra

Integration tools:

- IBM's Cloud Park for Data
- IBM's Cloud Park for Integration
- Talend Data Fabric
- Open Studio
- SnapLogic

Data processing layer

- read data in batch or streaming modes from storage and apply transformations

- support popular querying tools and programming languages
- scale to meet the processing demands of a growing dataset
- provide a way for analysts and data scientists to work with data in the data platform

Transformation tasks:

- Structuring
- Normalization: cleaning the database of unused data and reducing redundancy and inconsistency
- Denormalization: combining data from multiple tables into a single table so that it can be queried more efficiently
- Data cleaning

Tools for transformations:

- spreadsheets, openrefine, google dataprep, watson studio refinery, trifacta, python, R

Storage and processing may not always be performed in separate layers

→ In relational databases thet can occur in the same layer

→ In big data systems, data can first be stored in Hadoop and then processed in a data processing engine like Spark

→ Data processing can also preceed the processing layer

Analysis and user interface layer

- processed data delivery for consumers
- consumers = BI analysts, business stakeholders, data scientists, data analyists, other apps and services
- querying tools and programming languages, apis, dashboards, jupyter notebooks, microsoft power bi

Data pipeline layer

- extract, transform and load tools
- implement and maintain a continuosly flowing data pipeline

Factors for selecting and designing data stores

A data store or data repository, refers to data that has been collected, organized and isolated so that it can be:

- used for business operation
- mined for reporting or data analysis

A repository can be:

- database
  - input
  - storage
  - search and retrieval
  - modification
  - two types
    - RDBMS
    - NoSQL
- data warehouse
- data mart
- big data store
  - high-volume
  - high-velocity
  - diverse types

    - fast analytics
    - split large files across multiple computers. Computations run in parallel on each node where data is stored
- data lake
    - large volumes of raw data in its native format
    - both relational and non-relational

Intended use of data:

- transactional systems: used for capturing high-volume transactions, need to be designed for high-speed read, write, and update operations
- analytical systems: need complex queries to be applied to large amounts of historical data aggregated from transacional systems. They need faster response times to complex queries
- schema design, indexing, and partitioning strategies have a big role to play in performance of systems based on how data is getting used
- scalability
- normalization

Storage considerations:

- performance: throughput and latency
- availability: access data when u need it, no downtime
- integrity: data must be safe from corruption, loss, and outside attack
- recoverability: storage solution should ensure you can recover your data in the event of failures and natural disasters

Privacy, security and governance

A secure data strategy is a layered approach. It includes:

- access control
- multizone encryption
- data management
- monitoring systems

USA:

- general data protecion regulation (GDPR)
- california consumer privacy art (CCPA)
- health insurance portability accountability act (HIPAA)

Security

Levels:

- pyisical infrastructure
- network
- application
- data

The CIA Triad

- **Confidentiality** through controlling unauthorized access
- **Integrity** through validating that your resources are trustworthy and have not been tampered with
- **Availability** by ensuring authorized users have access to resources when they need it

Applicable to all levels of security

Physical Infrastructure Security

- access to the perimeter of the facility based on auth.
- multiple power feeds
- heating and cooling mechanisms
- factoring enviromental threats

Network Security

- firewalls
- network access control
- network segmentation to create silos or virual local area networks
- security protocols
- intrution detection and prevention

Application Security

- needs to be built in the foundation of the app
- threat modeling to identify relative weakness
- secure design
- secure coding guides
- security testing

Data Security

Data is either at rest in storage or in transit.

- authentication systems
- data at rest:
  - stored physically,
  - can be protected by encryption
- data in transit:
  - protected using encryption (https, ssl, tls)

Monitoring and Intelligence

- create an audit history for triage and compiance purposes
- provide reports and alerts that help enterprises react to security violations in time

How to gather and import data

SQL

SQL is a querying language used for extracting information from relational databases

- commands to specify:
  - what to retrieve from the DB
  - table from which it needs to be extracted
  - grouping records with matchiing values
  - dictating the sequence in which the query results are displayed
  - limiting the number of results that can be returned by the query
- non-relational DB 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

APIs

- they are invoked from apps that require the data and access an endpoint containing the data. Endpoints can include databases, web services, and data marketplaces
- used for data validation

Web scraping

- downloading specific data from web pages based on defined parameters
- extract data such as text, contact info, images...
- RSS feeds are also used for capture data

Data streams

- instruments, iot devices, sensors, GPS data from cars, apps
- data streams and feeds are also used for extracting data from social media sites and interactive platforms
- data streams are a popular source for aggregating constant streams of data flowing from sources

Data exchange

- allow the exchange of data between data providers and data consumers
- set of well-defined exchange standards, protocols and formats
- facilitate the exchange
- ensure that security and governance are maintained
- data licensing workflows
- examples: AWS dataexvhange, crunchbase, lotame, snowflake

Importing data

- structured data
  - relational databases
  - NoSQL
- semi structured data
  - NoSQL
- unstructured data
  - NoSQL
  - data lakes

Data wrangling or munging

Data weangling or munging is an iterative process that involves:

- data exploration
- transformation
- validation
- making data available for credible and meaningful analysis

Transformation

- actions that change the the form and schema of your data
- joins: combined columns in the same row
- unions: combined rows
- normalization and denormalization of data
  - normalization: cleaning unused data, reduce redundancy, reduce inconsistency

    - denormalization: combine data from multiple tables so it can be queried faster
- cleaning: fix irregularities in data
    - detect issues and errors with scripts and tools
    - validate against rules and constraints
    - profiling data to inspect source data
        - check structures etc
    - visualization tools
        - useing statistical methods
        - plot the average and look for abnormal values
    - missing values
        - filter out these records
        - source missing info
        - imputation: calculates the missing value based on statistical values
    - duplicate data
        - need to be removed
    - irrelevant data
    - data type conversion to ensure values in a field are stored as the data type of that field
    - standarizing data (all with lowcase for ex)
    - syntax errors such as white spaces, extra spaces, typos, formats
    - outliers are values that are very different from the others and needs to be examined

Tools for data wrangling

- excel power query / spreadsheets
    - in-built formulae
    - add-ins allow importing from different sources
- openRefine
    - open-source
    - import and export in a wide variety of formats: TSV, CSV, XLS, XML, JSON
    - clean data, transform its format, extend data with web services and external data
    - easy to learn and to use
    - menu-based operations (no need to memorize syntax)
- google dataPrep
    - intelligent cloud data service
    - visually explore, clean and prepare both structured and unstructured data for analysis
    - fully managed service: no need to install or manage infrastructure
    - extremly easy to use
    - offers suggestions for next steps
    - automatically detect schemas, data types and anomalies
- watson studio refinery
    - IBM watson studio
    - discover, cleanse and transform data with built-in operations
    - transforms large amounts of raw data into consumable, quality information that is ready for analytics

    - flexibility of exploring data residing in a spectrum of data sources
    - detects data type and classifications automatically
    - enforces applicable data governance policies automatically
- trifacta wrangler
    - interactive cloud-based service for cleaning and transforming data
    - takes messy, real-wordl data and cleans and rearranges it into data tables
    - can export tables to Excel, Tableau and R
    - collaboration features
- python
    - libraries and packages
    - jupyter notebook: data cleaning and transformation, statistically modeling and data visualization
    - numpy: fast, versatile, easy to use. Large multi-dimensional arrays and matrices, high-level math functions
    - pandas: fast and easy data analysis. Complex operations such as merging, joinging and transforming huge chunks of data. Helps prevent common errors that result from misaligned data coming from different sources
- R
    - libraries and packages
    - invesigtate manipulate and analyse data
    - dplyr: powerful library for data wrangling with a precise and straightforward syntax
    - Data.table: helps aggregate large data sets quickly
    - jsonlite: a robust json parsing tools, great for interacting with web APIs

Lab: Load data into the IBM Db2 database from a CSV file

Congratulations! You have successfully loaded data from a CSV file into the IBM Db2 instance you created in the previous lab.

Now that you have learned about the process of importing data into a data repository from varied sources, you will load data from a CSV file into the IBM Db2 database instance you created in the previous lab.

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0100EN-SkillsNetwork/labs/1.3.2.5_Hands-on_Lab_Load_data_into_Db2_Database3_from_CSV_file.md.html?origin=www.coursera.org

Congratulations! You have successfully loaded data from a CSV file into the IBM Db2 instance you created in the previous lab.

Querying and analyzing data → SQL

Count

```
select count(*) CarSaleDetails; /*count number of rows or records*/

select distinct Dealer from CarSaleDetails; /*it shows only the Dealer column*/

select count(distinct Dealer) from CarSaleDetails; /*total number of unique, or distinct, car dealers*/
```

Aggregation

```
select sum(Sale_Amount) as sum_all from CarSaleDetails;  /* calculates the sum of a numeric column */

select avg(Sale_Amount) as avg_all from CarSaleDetails;

select stddev(Sale_Amount) as std_all from CarSaleDetails;  /* standard deviation to see how spread out the cost of a used car is
 */
```

Extreme value identification

```
select max(Sale_Amount) as max_price from CarSaleDetails;  /* max value in a column */

select min(Sale_Amount) as min_price from CarSaleDetails;
```

Slicing data

```
select Dealer from CarSaleDetails where Dealer City in ("Albuquerque", "Texline");
```

Sorting data

```
select * from CarSaleDetails order by Date_of_Purchase;
```

Filtering patterns

```
select * from CarSaleDetails where Pin like "871%";  /* returns records that match a data value PARTIALLY. This can return 8710, 87
11, ... */
```

Grouping data

```
select sum(Sale_Amount) as area_sum, Pin from CarSaleDetails group by Pin;  /* Total amount spent by customers, pincode-wise */
```

Performance tuning and troubleshooting

Data pipelines

- typically runs with a combination of complex tools and can face several different types of performance threats
- scalability in the face of increasing data sets and workloads
- app failures
- scheduled jobs not functioning
- tool incompatibilities

Performance metrics:

- latency
- failures
- resource utilization and utilization patterns
- traffic

Troubleshooting:

- collect information
- check if we're working with all the right versions of software and source codes
- check logs and metrics
- reproduce the issue in a test enviroment

Data base optimization for performance

Performance metrics:

- system outages
- capacity utilization
- application slowdown
- performance of queries

- conflicting activities and queries being executed simultaneously
- batch activities causing resource constraints

Best practices:

- capacity planning: determining the optimal hardware and software resources required for perfomance
- database indexing: locating data without searching each row in a database resulting in faster querying
- database partitioning: dividing large tables into smaller, individual tables, improving performance and data manageablitiy. Queries run faster.
- database normalization

Monitoring systems

- help collect quantitative data about our systems and apps in real time
- visibility to the performance of data pipelines, data platforms, databases, apps, tools, queries, scheduled jobs, and more
- **Data base monitoring tools:** frequent snapshots of the performance indicators of a databse
  - track when and how a problem started to ocurr
  - isolate and get to the root of the issue
- **Application performance management tools**: measure and monitor the performance of apps and amount of resources utilized by each process
  - helps in proactive allocation of resources to improve performance
- **Tools for monitoring query performance**: gather statistics about query throughput, exeecution, performance, resource utilization and utilization patterns for better planning and allocation of resources
- **Job-level runtime monitoring**: breaks up the job into a series of logical steps, so that errors can be detected before completion
- **Monitoring amount of data being processed:** through a data pipeline helps to asses if size of workload is slowing down the system

Maintenance schedules

Preventive maintenance routines generate data that we can use to identify systems and procedures responsible for faults and low availability.

These routines can be:

- Time-based. That is, they could be planned as scheduled activities at pre-fixed time intervals.
- Condition-based, which means they are performed when there is a specific issue or when a decrease in performance has been noted or flagged.

Lab: Explore your dataset using SQL queries

Congratulations! You have successfully run some SQL queries to help you explore and understand your dataset.

Now that you have learned how querying techniques can help you to explore and analyze your data, you will run some basic SQL queries on the data you loaded into your database instance in the previous lab. For this, you will use the in-built SQL editor available in your Lite account.

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0100EN-SkillsNetwork/labs/1.3.3.3_Hands-on_Lab_Explore_your_dataset_using_SQL_queries.md.html?origin=www.coursera.org

Governance and compliance

Governance

Data **Governance** is a collection of principles, practices, and processes to maintain the security, privacy, and integrity of data through its lifecycle. A data governance framework encompasses every part of an organization's data management process—including the technologies, databases, and data models.

A data governance framework encompasses every part of an organization's data management process—including the technologies, databases, and data models.

What are the types of data these regulations seek to protect from misuse?

- Primarily, personal and sensitive data. (PI)
  - That is data that can be traced back to an individual,
  - can be used to identify an individual,
  - or contains information that can be used to cause harm to the person
  - GDPR (eruopean union)
  - CCPA (california)
  - industry-specific regulations
    - HIPAA
    - PCI DSS (retail)
    - SOX (finance)

Compliance

Compliance covers the processes and procedures through which an organization adheres to regulations and conducts its operations in a legal and ethical manner. Organizations need to establish controls and checks in order to comply with regulations

Data Lifecycle

- **Data acquisition stage**
  - identify data that needs to be collected and the legal basis for procuring the data
  - establish the intended use of data, published as a privacy policy
  - identify the amount of data you need to meed your defined purposes
- **Data processing stage**
  - flesh out details of how exactly you are going to process personal data
  - establish your legal basis for the processing of personal data
- **Data storage stage**
  - define where you will store data
  - establish specific measures you will take to prevent internal and external security breaches
- **Data sharing stage**
  - identify third-party vendors in your supply chain that will have access to the collected data
  - establish how you will hold third-party vendors contractually accountable to regulations
- **Data retention and disposal stage**
  - define policies and processes you will follow
  - define how you will ensure deleted data is removed from all locations, including third-party systems

Technology as an enabler

- authentication and access control
- encryption and data masking for data at rest and in transit
  - the process of anonymization abstracts the presentation layer without changing the data in the database physically
- hosting options
- monitoring and alerting functionalities
- data erasure → is a software-based method of permanently clearing data from a system by overwriting. This is different from a simple deletion of data since deleted data can still be retrieved.

Overview of the DataOps Methodology

Gartner defines DataOps as a collaborative data management practice focused on improving the communication, integration, and automation of data flows between data managers and consumers across an organization. DataOps aims to create predictable delivery and change management of
data, data models, and related artifacts. DataOps uses technology to automate data delivery with the appropriate levels of security, quality, and metadata to improve the use and value of data in a dynamic environment.”

(Source: https://blogs.gartner.com/nick-heudecker/hyping-dataops/)

A small team working on a simpler or limited number of use cases can meet business requirements efficiently. As data pipelines and data infrastructures get more complex, and data teams and consumers grow in size, you need development processes and efficient collaboration between teams to govern the data and analytics lifecycle. From data ingestion and data processing to analytics and reporting, you need to reduce data defects, ensure shorter cycle times, and ensure 360-degree access to quality data for all stakeholders.

DataOps helps you achieve this through metadata management, workflow and test automation, code repositories, collaboration tools, and orchestration to help manage complex tasks and workflows. Using the DataOps methodology ensures all activities occur in the right order the
right security permissions. It helps set in a continual process that allows you to cut wastages, streamline steps, automate processes, increase throughput, and improve continually.

Several DataOps Platforms are available in the market, some of the popular ones being IBM DataOps, Nexla, Switchboard, Streamsets, and Infoworks.

DataOps Methodology

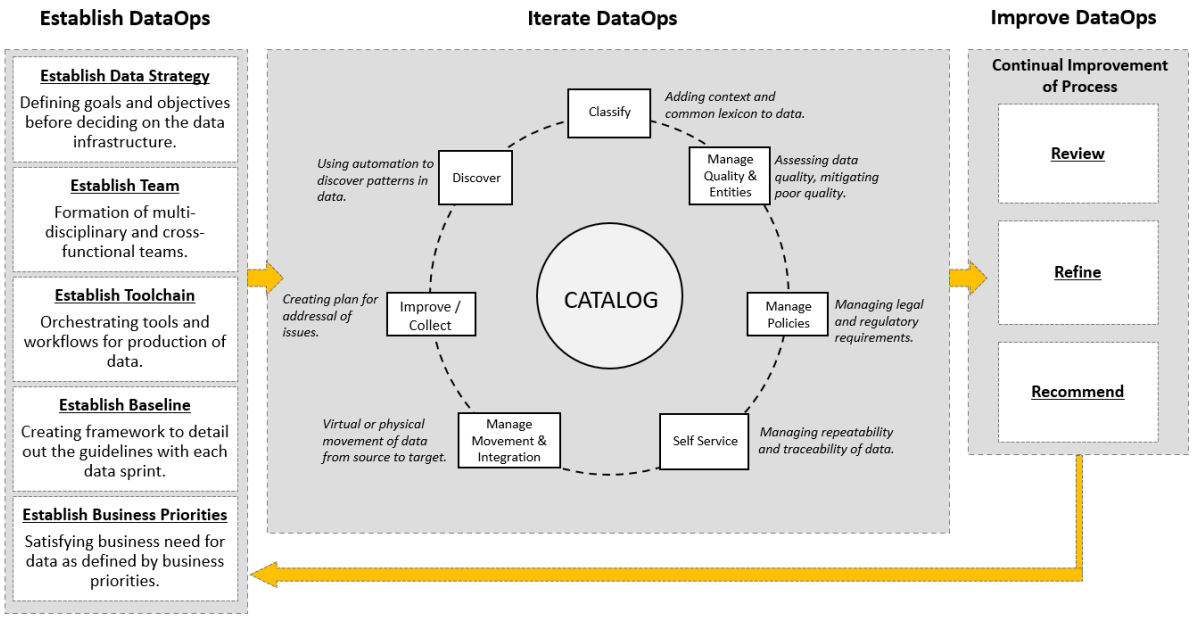
The purpose of the DataOps Methodology is to enable an organization to utilize a repeatable process to build and deploy analytics and data pipelines. Successful implementation of this methodology allows an organization to know, trust, and use data to drive value.

It ensures that the data used in problem-solving and decision making is relevant, reliable, and traceable and improves the probability of achieving desired business outcomes. And it does so by tackling the challenges associated with inefficiencies in accessing, preparing, integrating, and making data available.

In a nutshell, the DataOps Methodology consists of three main phases:

- The **Establish DataOps Phase** provides guidance on how to set up the organization for success in managing data.
- The **Iterate DataOps Phase** delivers the data for one defined sprint.
- The **Improve DataOps Phase** ensures learnings from each sprint is channeled back to continually improve the DataOps process.

The figure below presents a high-level overview of these phases and the key activities within each of these phases.



Benefits of using the DataOps methodology

Adopting the DataOps methodology helps organizations to organize their data and make it more trusted and secure. Using the DataOps methodology, organizations can:

- Automate metadata management and catalog data assets, making them easy to access.
- Trace data lineage to establish its credibility and for compliance and audit purposes.
- Automate workflows and jobs in the data lifecycle to ensure data integrity, relevancy, and security.
- Streamline the workflow and processes to ensure data access and delivery needs can be met at optimal speed.
- Ensure a business-ready data pipeline that is always available for all data consumers and business stakeholders.
- Build a data-driven culture in the organization through automation, data quality, and governance.

As a data practitioner, using the methodology can help you reduce development time, cut wastages and duplication of effort, increase your productivity and throughput, and ensure that your actions produce the best possible quality of data.

With DataOps, data professionals, consumers, and stakeholders can collaborate more effectively towards the shared goal of creating valuable insights for business. While implementing the methodology will require systemic change, time, and resources, but in the end, it makes data and analytics more efficient and reliable.

Interestingly, it also opens up additional career opportunities for you as a data engineer. **DataOps Engineers** are technical professionals that focus on the development and deployment lifecycle rather than the product itself. And as you grow in experience, you can move into more specialist roles within DataOps, contributing to defining the data strategy, developing and deploying business processes, establishing performance metrics, and measuring performance.