Chapter 1
Introduction to Data Engineering

Contents:

- DE Introduction
- Types of data, Big Data
- Data wrangling & characteristics
- ETL/ELT models and differences
- On Data Warehouses
- Fundamentals of Data Pipelines & different types of DPs
- Design of a Data Pipeline – Detailed approach with an example

Data Engineering – Introduction

Who is a Data Engineer (DE)?

- A DE is associated with data - their extraction, delivery, storage, wrangling, and pre-processing;
- A DE is expected to provide a reliable compute infrastructure for data;

Who is a Data Scientist (DS)?

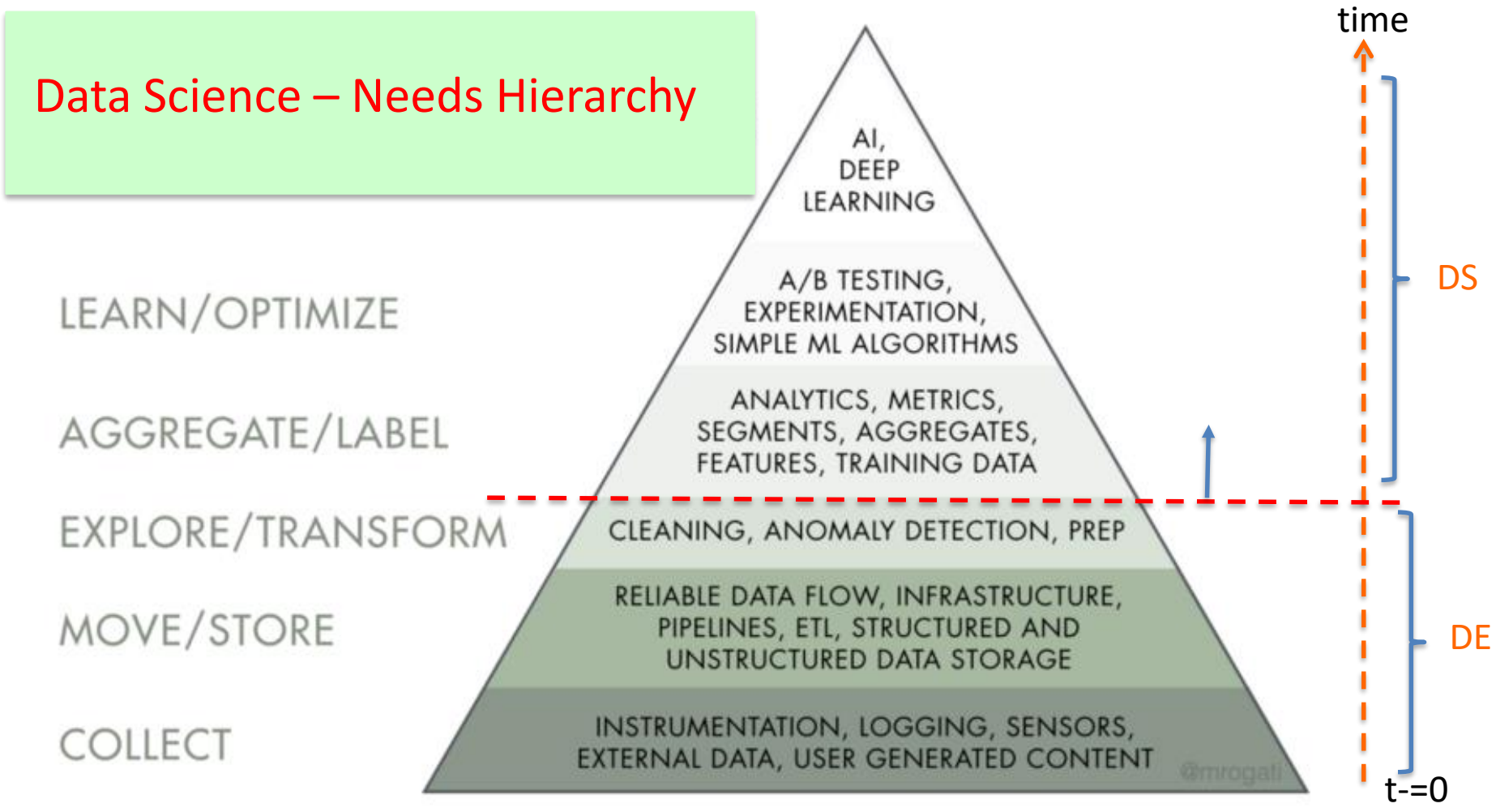
- Uses sophisticated analytics programs and machine learning and statistical methods on the wrangled data to make complete sense of data and to interpret behaviors/trends;
- Performs modeling, predictive anlaytics, etc

Data sources & sizes – Examples of some real-life applications

- Social media data - 500++ terabytes everyday added to the DB (Ex: FB, Twitter, etc)
- Jet Engine – ~ every 30 mins generates, 10-12 terabyes of data per engine! Total data then scales easily into Peta/Exa-scales!!
- Weather forecast – on a conservative scale, 1.5 terabytes of meteorological data collected each day by certain countries specific to certain areas;

Data Engineering… (Cont'd)

Data Science – Needs Hierarchy

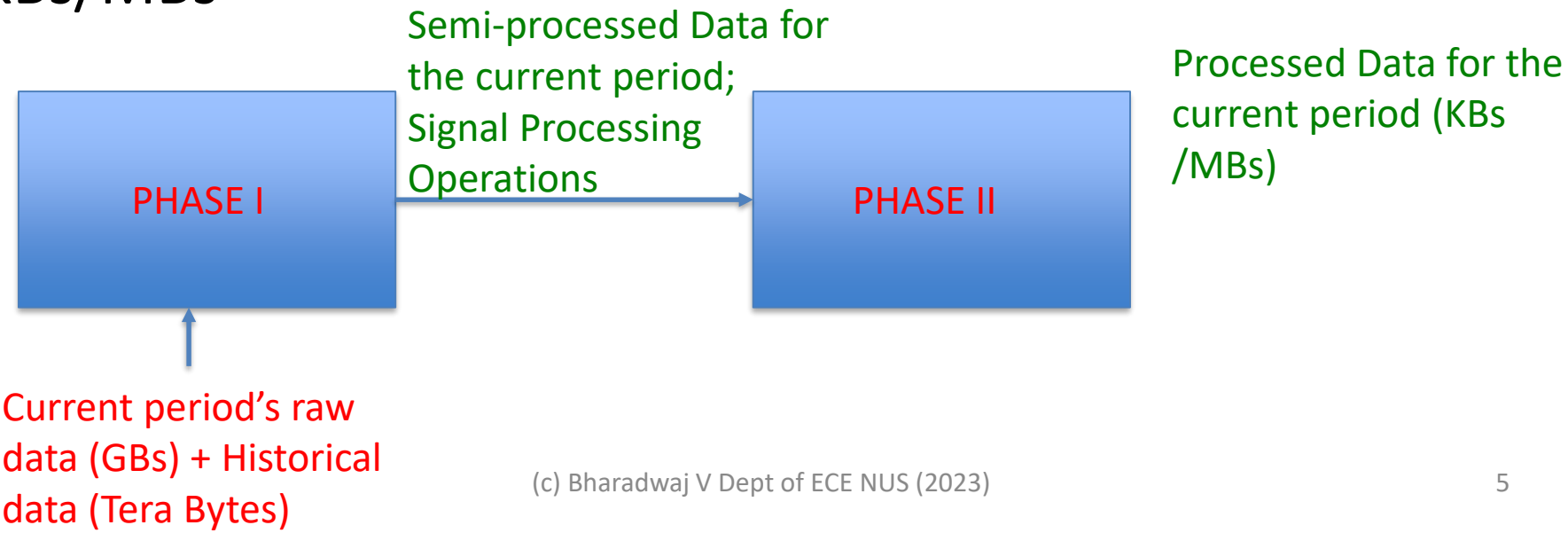


Observe the width of the pyramid above; Larger the width implies more the amount of work to be done in the respective phases;

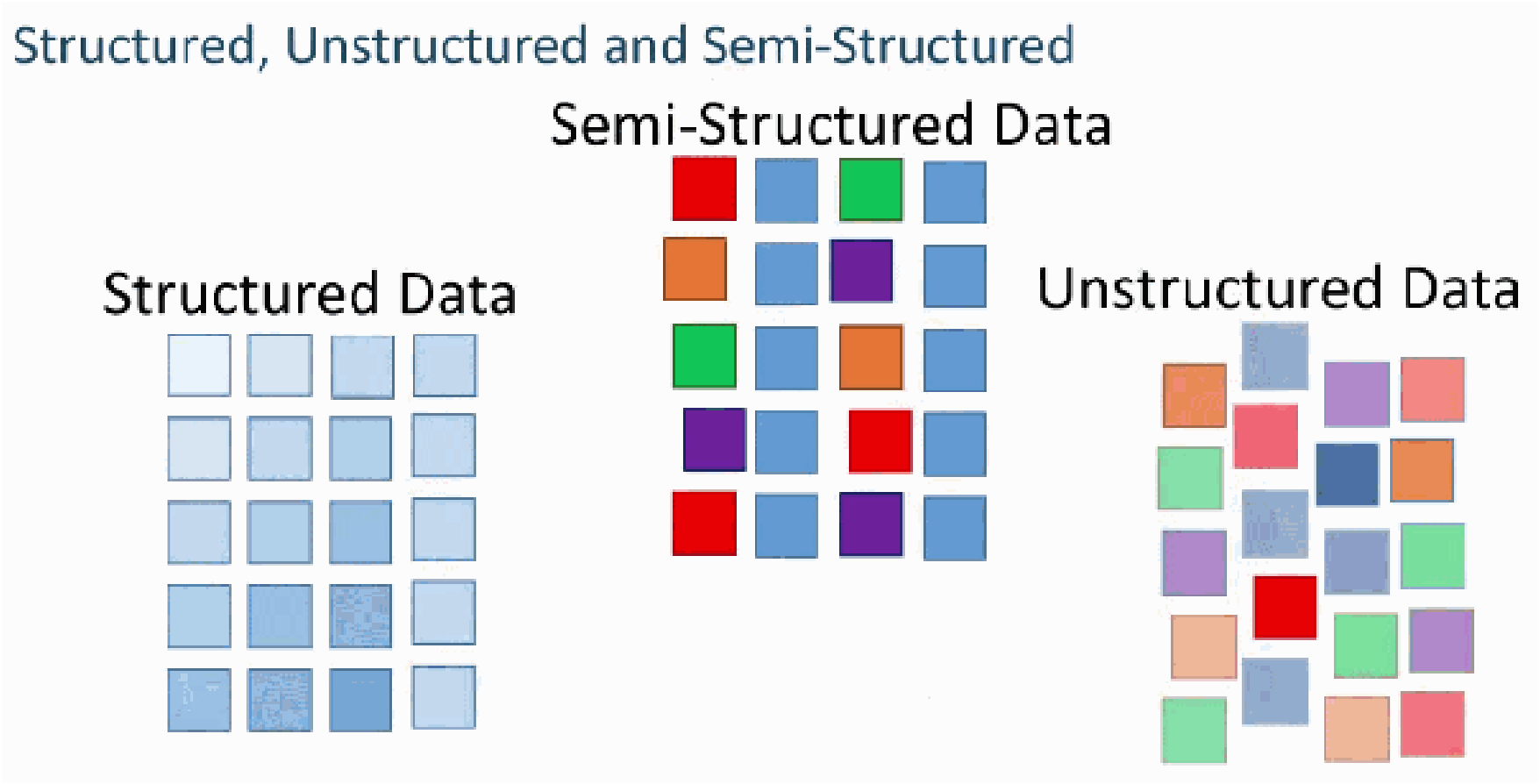
Types of data

- It is possible that the data may "expand" or "shrink" during processing!

Weather prediction data processing: Two phases: Phase I (during processing)- raw data + historical data; Phase II (tail end processing) – Processed data (useful output) in KBs/MBs



Types of data



Unstructured data

- Any data with unknown form or the format is classified as unstructured data.
- Challenges – Data processing and inference through deriving any meaningful value associated with a query.

Example: A typical example of unstructured data is a heterogeneous data source containing a combination of simple text files, images, videos etc.

Output of your Google search is a good example for this category!

Structured data

- Data that can be stored, accessed and processed in the form of fixed format is termed as a 'structured' data;

Example:

First Name	Gender	Start Date	Salary	Bonus %	Team
Douglas	Male	8/6/93	97308	6.945	Marketing
Thomas	Male	3/31/1996	61933		
Maria	Female	4/23/1993	130590	11.858	Finance
Jerry	Male	3/4/05	138705	9.34	Finance
Larry	Male		101004	1.389	Client Services
Dennis	Male	4/18/1987	115163	10.125	Legal
Ruby	Female	8/17/1987	65476	10.012	Product
	Female	7/20/2015	45906	11.598	Finance

Semi-structured data

- Semi-structured data can contain both the forms of data.
- We can see semi-structured data as a structured data in a specific form but may not be actually defined using a table definition; Plus, attributes could be different;
- Example - Data represented in an XML file

<rec><name>Susan</name><gender>Female</gender></rec>

<rec><name>Seema</name><gender>Female</gender><age>41</age></rec>

<rec><name>Aaron</name><gender>Male</gender><age>29</age><Addr>35 BB West, Sg</Addr></rec>

Big data Characteristics

- V V V V (4 Vs)

Volume – Refers to size which is enormous. Size of data plays a very crucial role in determining value out of data;

Variety- Refers to heterogeneous sources and the nature of data, both structured and unstructured; Ex: Spreadsheets, databases, emails, photos, videos, monitoring devices, PDFs, audio, etc;

Velocity – Refers to the speed/rate of generation of data; This contributes to volume of the data;

Variability – Refers to any inconsistency which can be shown by the data at times; Inconsistency hampers the process and management of the data effectively; In stats, this also means a range of coverage.

Raw Data Characteristics

Raw data refers to data that are directly acquired from different data sources, sensors (in the case of IoT based systems) and *need not confirm to any specific format* that users may expect during the processing;

Raw data usually has some "gaps" or "missing data" in some of the sample observations; This is a very common problem in any raw data;

First Name	Gender	Start Date	Salary	Bonus %	Team
Douglas	Male	8/6/93	101106	6.945	Finance
Thomas	Male	3/31/1996			Finance
Maria	Female	4/23/1993	130590	11.858	Finance
Jerry	Male	3/4/05	138705	9.34	Finance
Larry	Male		101004	1.389	Finance
Dennis	Male	4/18/1987	105163	10.125	Finance
Ruby	Female	8/17/1987	165476	10.012	Finance
	Female	7/20/2015	140000	11.598	Finance
	Female			10.012	Finance

Data Wrangling – Six imperative steps

As a DE, your first job is to "***wrangle***" the raw data that is passed to you!

Data wrangling – Refers to the *process of cleaning, structuring and enriching raw data* into a desired format for quick and efficient processing in the analytics stage of the entire process!

Following are 6 mandatory steps involved in data wrangling operations:

Data Discovery	Data Structuring	Data Cleaning
Data Enrichment	Data Validation	Data Publishing

Data Wrangling

Six imperative steps involved!

Data Discovery

Understanding what is in your data - which will inform *how you want to clean, position it and analyze it*. How you wrangle data, for example, may be informed by where they are located, where they bought data, about data sources, etc.

Data Structuring

Organizing the data, which is necessary because raw data comes in many different shapes and sizes;

Data Wrangling… (Cont'd)

Data Cleaning

What happens when errors and outliers skew your data? You clean the data. What happens when data is entered as SG/Sg or Singapore or S'pore? Question that can be asked:
How to deal with Nulls? What if I replace a missing data with some statistical quantity?

Data Enrichment

Taking stock in your data and strategize about how other additional data might augment it. Questions to be asked during this step includes: What new types of data can I derive from what I already have or what other information would better influence my decision making about this current data? What type of queries I can entertain?

* Example – Feature selection

Data Wrangling… (Cont'd)

Data Validation

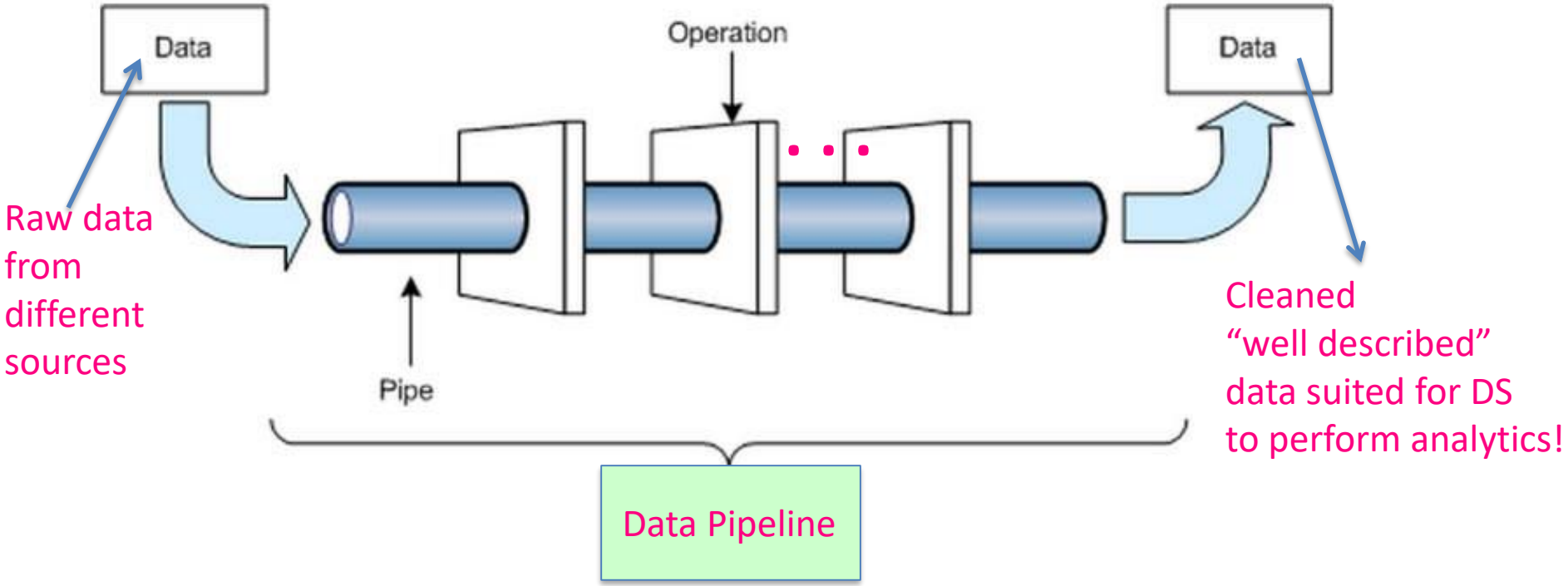
Validation rules are repetitive programming sequences that verify data consistency, quality, and security. Examples of validation include: Ensuring a uniform distribution of attributes that should be distributed normally (e.g. birth dates) or confirming accuracy of fields through a check across data.

Data Publishing

DE prepares the wrangled data for use in a downstream analysis – whether by a particular user or software and documents any particular steps taken or logic used to wrangle said data. After this step, data is ready for performing analytics by a DS!

Data Engineering… (Cont'd)

As a DE, you will go through all the above mentioned stages in data wrangling. DE spends most of the time in setting up a process flow, referred to as a "*data pipeline*".



Data Engineering… (Cont'd)

A DE will go through the following three stage process:

- Extraction
- Transform
- Load

Data Wrangling is categorized into 3 major categories as E-T-L

The above three seems imperative in the entire data processing pipeline (*we will see an example of such a pipeline later*)

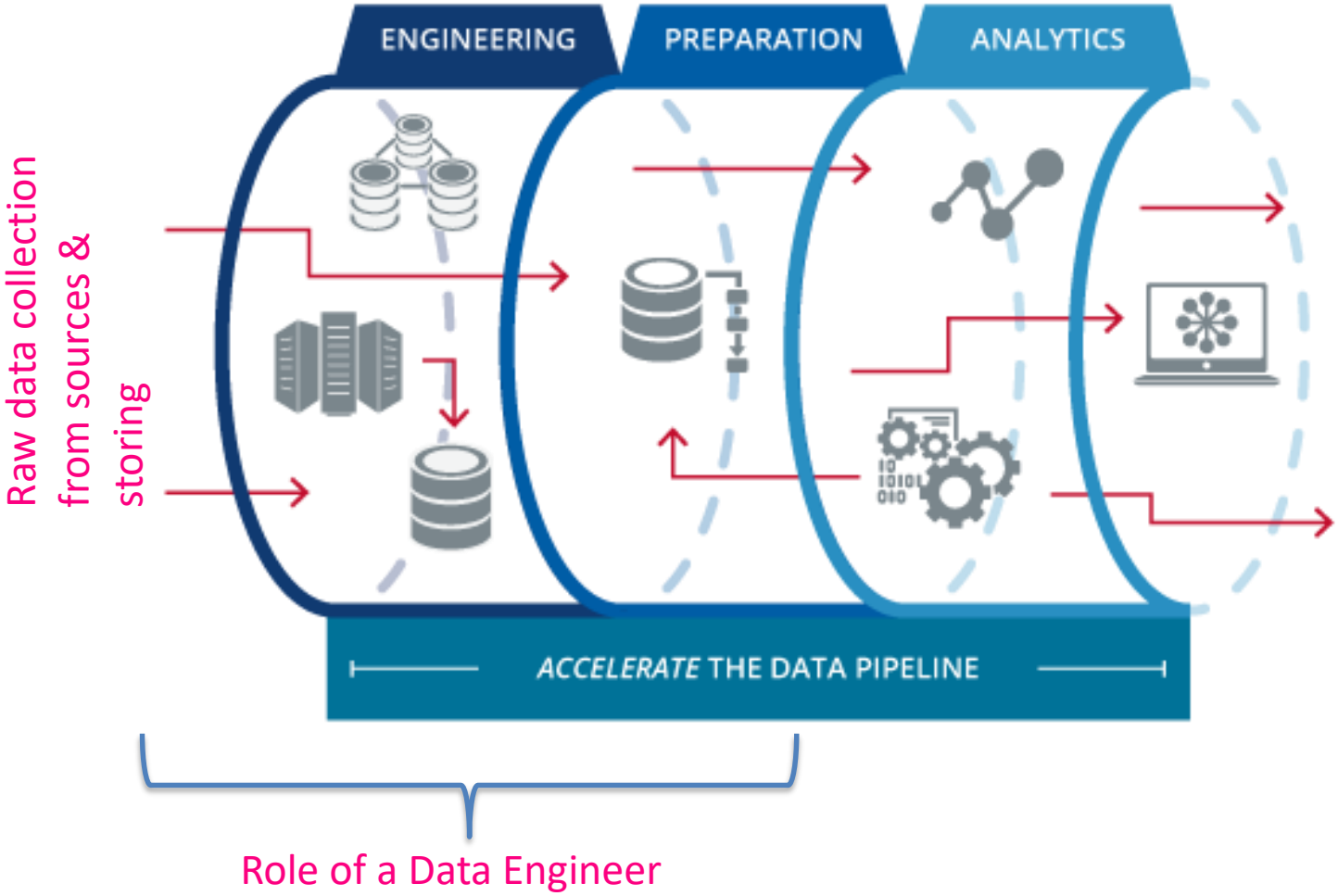
Data Extraction-Transform-Load (ETL)

Extraction: Retrieving raw data from an unstructured data pool and migrating it into a temporary, staging data repository;

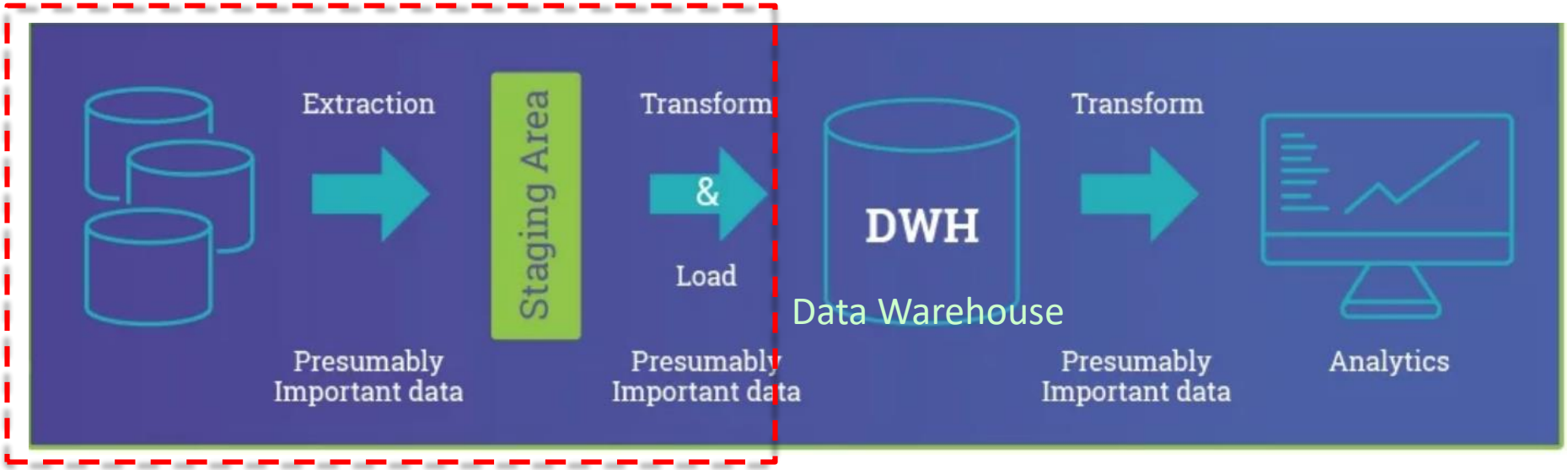
Transformation: Structuring, enriching and converting the raw data to match the target source/application;

Loading: Loading the structured data into a data warehouse to be analyzed and used by scientific tools (ML/DL, etc);

Extraction-Transform-Load (ETL)



Data Extraction-Transform-Load (ETL)



Using Python Pandas, we can do this effectively

Modern way of doing (still evolving!)

Extraction-Load-Transform (ELT)

What are the key differences?

Differences: ETL vs ELT

E-T-L:

- Each stage - extraction, transformation and loading - demands interaction by data engineers and developers, and also dealing with capacity limitations of traditional data warehouses.

- Using ETL, analysts and other users have no choice but to wait for the information as it is not made available until the whole ETL process has been completed!

Differences: ETL vs ELT

ELT – You can start immediately the loading phase - moving all the data sources into a single, centralized data repository. Start your analytics right after extraction phase!

Courtesy: Hadoop

E&L here

ETL Tools

- ETL is the heart of any data warehousing system. As a Data Warehouse (DWH) is built to serve pre-processing of data, Python developers over years have developed a number of ETL tools!

Here are a few tools!

- *Airflow; Spark; Petl; Panoply; Pandas, Bubbles; Bonobo; etlalchemy; mETL; Open Semantic ETL; Mara; riko; Carry; locopy; etlpy; pygrametl;*

Quick note on Data Warehouse

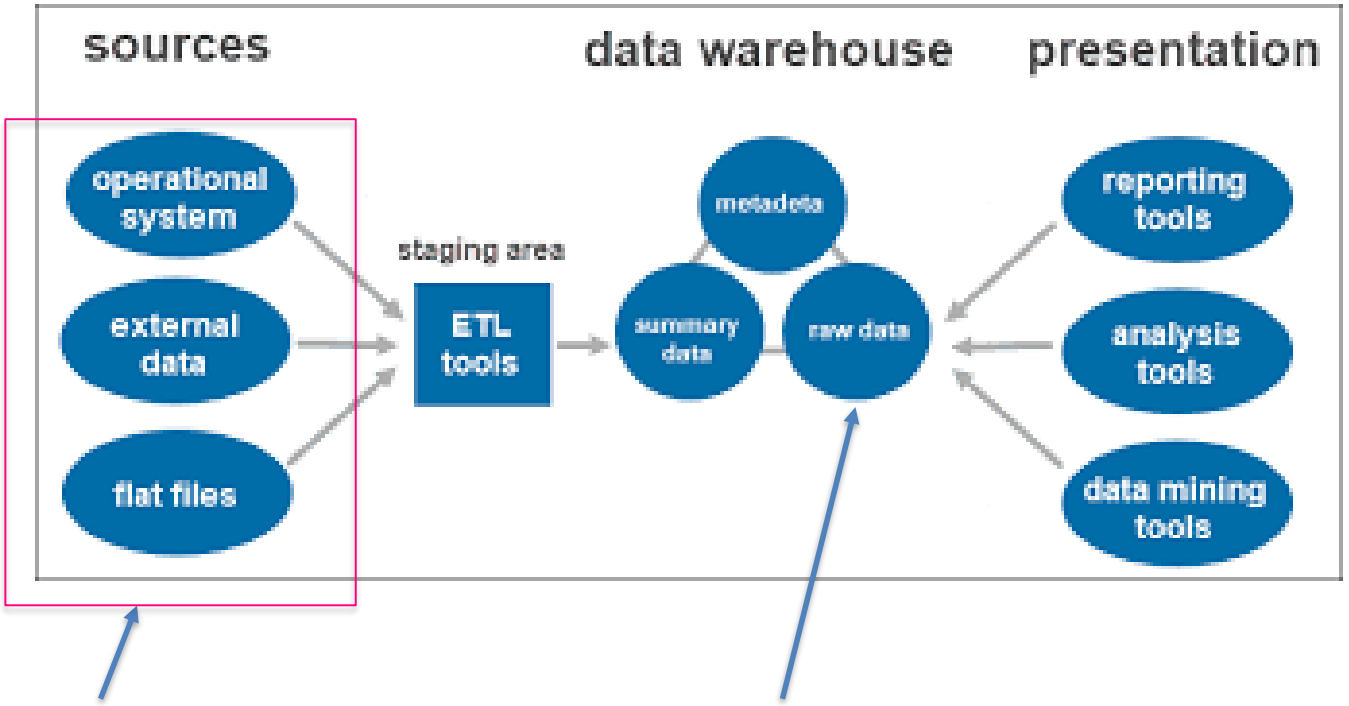
- DWH - A *decision support system* which stores historical data from across the organization;

- *What does DWH do?*

  DWH processes data and makes it available for critical business analysis, reports, and dashboards

  Stores data from numerous data sources, mostly structured - These include, *Online Transaction Processing (OLTP) data* such as invoices and financial transactions, *Enterprise Resource Planning (ERP) data*, and *Customer Relationship Management (CRM) data*

Data Warehouse



- IoT raw data
- From existing DBs
- Third-party vendors
- Data generated by other applications

Sort of "cleaned"

DWH & Data Lake

DWH – *Not a new concept*, however, with the current day compute platforms - in a Cloud setting DWH is gaining attention as it is offered as a *well managed service*!

Cloud Service Provider (CSP) assures higher performance, optimized resource usage, etc., thus bringing the monetary cost to a minimum!

DWHs predominantly store *structured & semi-structured data* for processing at later stages – usually to support SQL and SQL-like DBs for query processing;

DWH & Data Lake

Data Lake – Origin since 2011, capable of storing structured, unstructured and semi-structured data, in their original forms to be used at a later stage

>> https://aws.amazon.com/big-data/datalakes-and-analytics/what-is-a-data-lake/

>> Good read! - https://www.dataversity.net/brief-history-data-lakes/

Out of scope of this course. Interested students can refer to the above links to catch up further reading!

Fundamentals of Data Pipelines

- *What is a Data Pipeline (DP)?*

DP *refers to processing of the underlying raw data in an ordered sequence of steps*, as needed by an end application; Each step feeds the next stage with a meaningful output

- Primary components: A source, a processing step or a set of steps, and a destination/sink

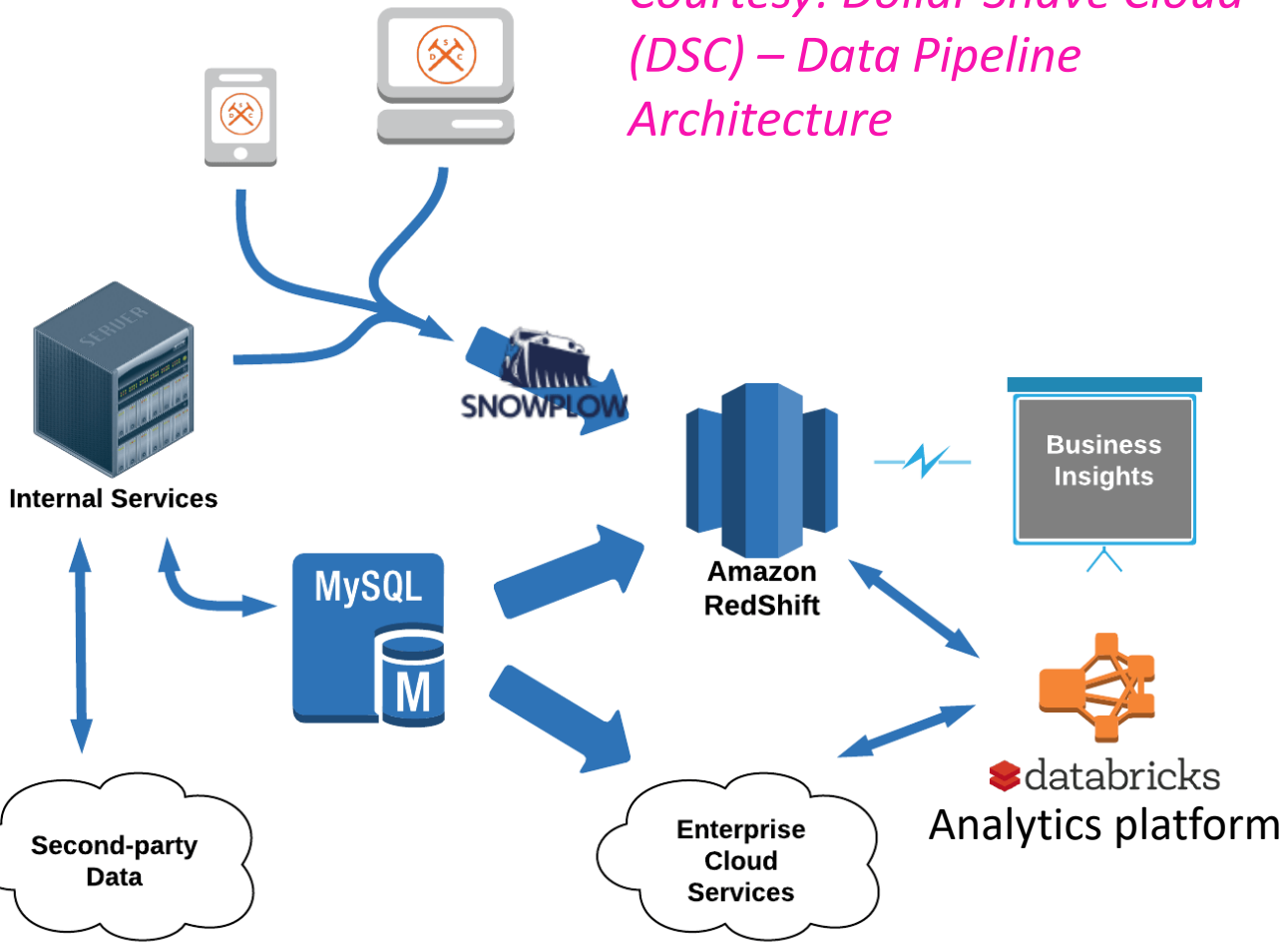
Remarks: Data pipelines may also have the same source and sink – *this means that the pipeline is purely about modifying the data set!*

Note that any time data is processed between steps X and Y there is a data pipeline between those steps.

Data Pipeline – An example

A DP may contain several intermediate components that include, temporary storages, computing nodes, communication modules, security modules, and a final storage for extracting any insights into the processed data.

Courtesy: Dollar Shave Cloud (DSC) – Data Pipeline Architecture



>> https://aws.amazon.com/solutions/case-studies/dollar-shave-club-case-study/

- DSC's web applications + internal services + data infrastructure *fully hosted on AWS*!
- Redshift cluster - central data warehouse, receiving data from various systems.
- Data movement - facilitated with *Apache Kafka* in all directions between the components
- Snowplow – Entity that collects data from the web and mobile clients.
- Analytics platforms extract data from Redshift for monitoring, visualization, and insights; This is done using the tool - Apache Spark, which is mainly used to build predictive models, to build recommendation systems for future sales;

Quick note on Apache Spark

Apache Spark - Heart of a Distributed Processing Framework!

- Data processing framework to handle processing tasks on very large data sets;
- Distribute data processing tasks across multiple nodes - either on its own or in tandem with other distributed computing tools

The above two characteristics are the key to handle big data and machine learning because they need to tap massive computing power to crunch through large data stores.

- Spark provides an easy-to-use APIs that abstracts away much of the mumble work in distributed computing and big data processing.

https://aws.amazon.com/big-data/what-is-spark/#:~:text=Apache%20Spark%20is%20an%20open,against%20data%20of%20any%20size.

To be cont’d…