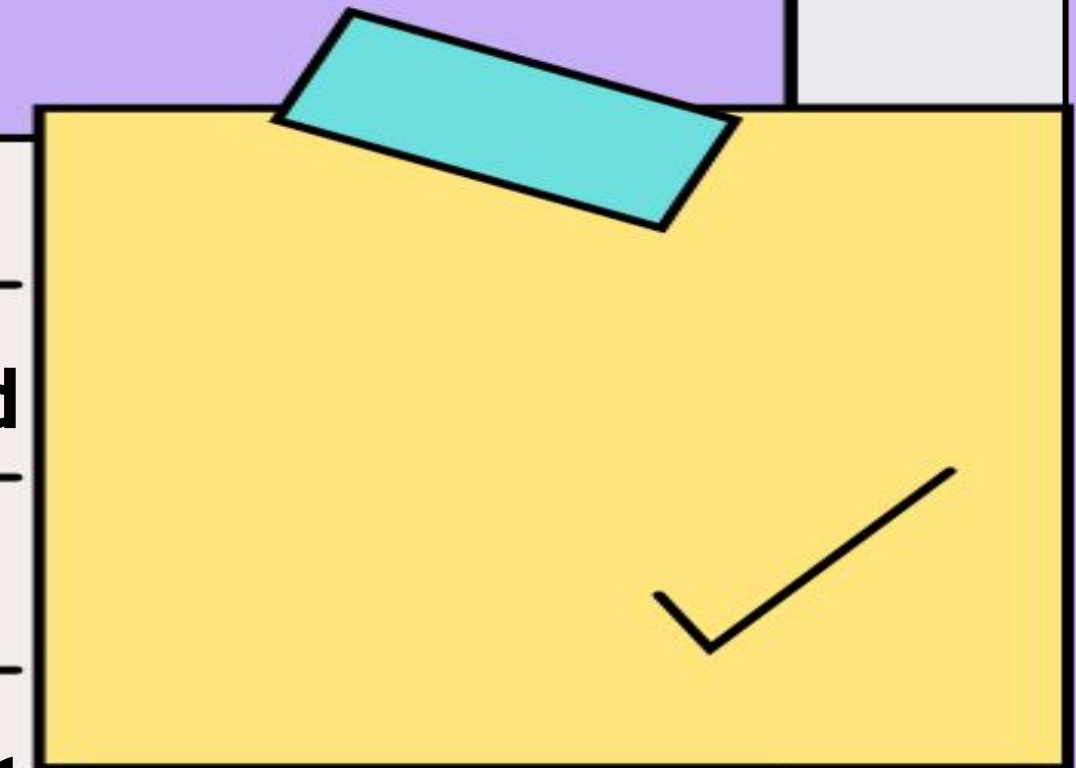




Cross Industry Standard Process for Data Mining (CRISP-DM)

What is CRISP in Data Mining?

CRISP-DM stands for the cross-industry standard process for data mining. The CRISP-DM methodology provides a structured approach to planning a data mining project. It is a robust and well-proven methodology. We do not claim any ownership over it. We did not invent it. We are a converter of its powerful practicality, flexibility, and usefulness when using analytics to solve business issues. It is the golden thread that runs through almost every client meeting.



How does CRISP Help?



CRISP DM provides a roadmap, it gives you best practices, and it provides structures for better and faster results of using data mining, so that's how it helps the business follow while planning and carrying out a data mining project.

Phases of CRISP-DM

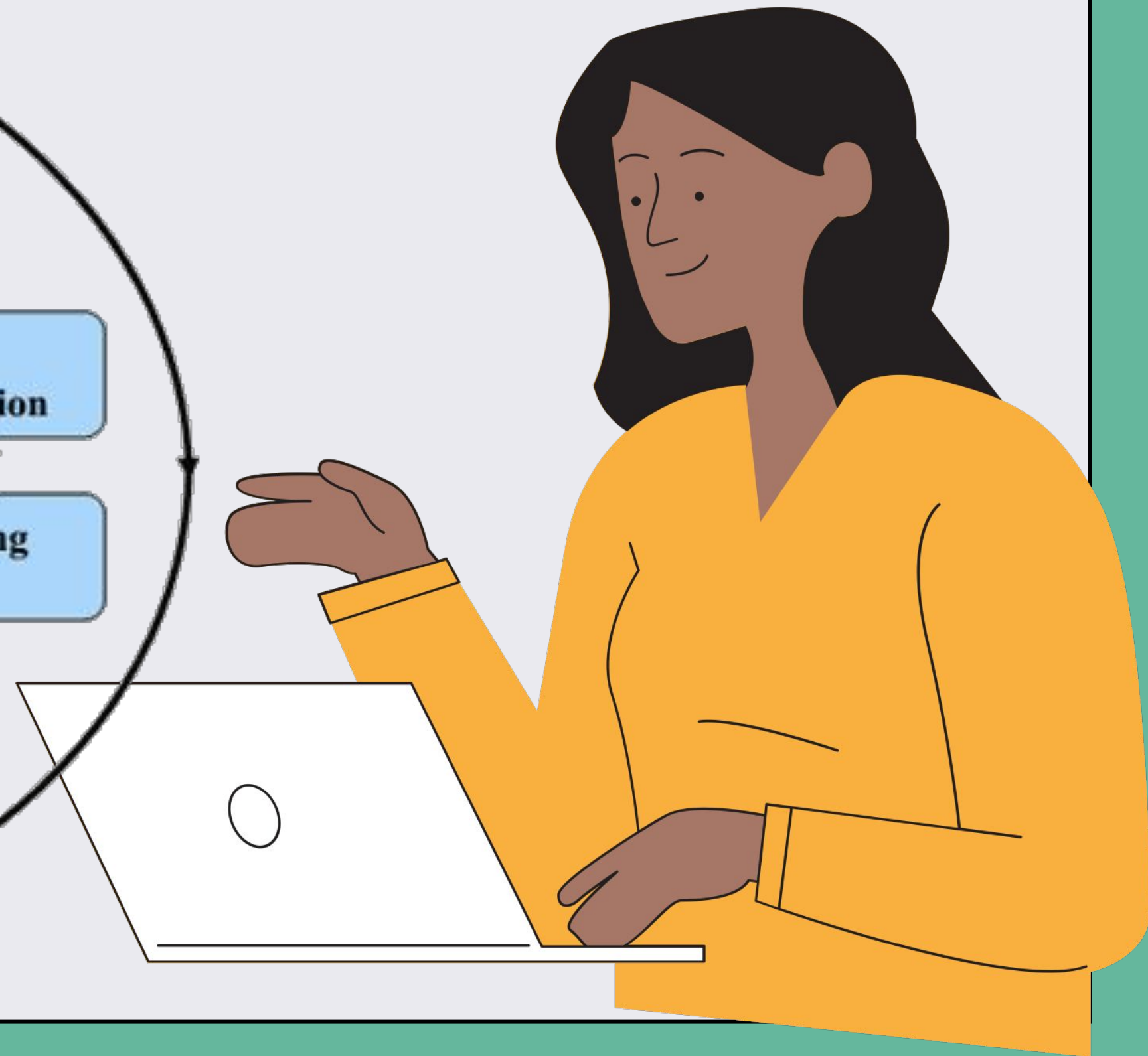
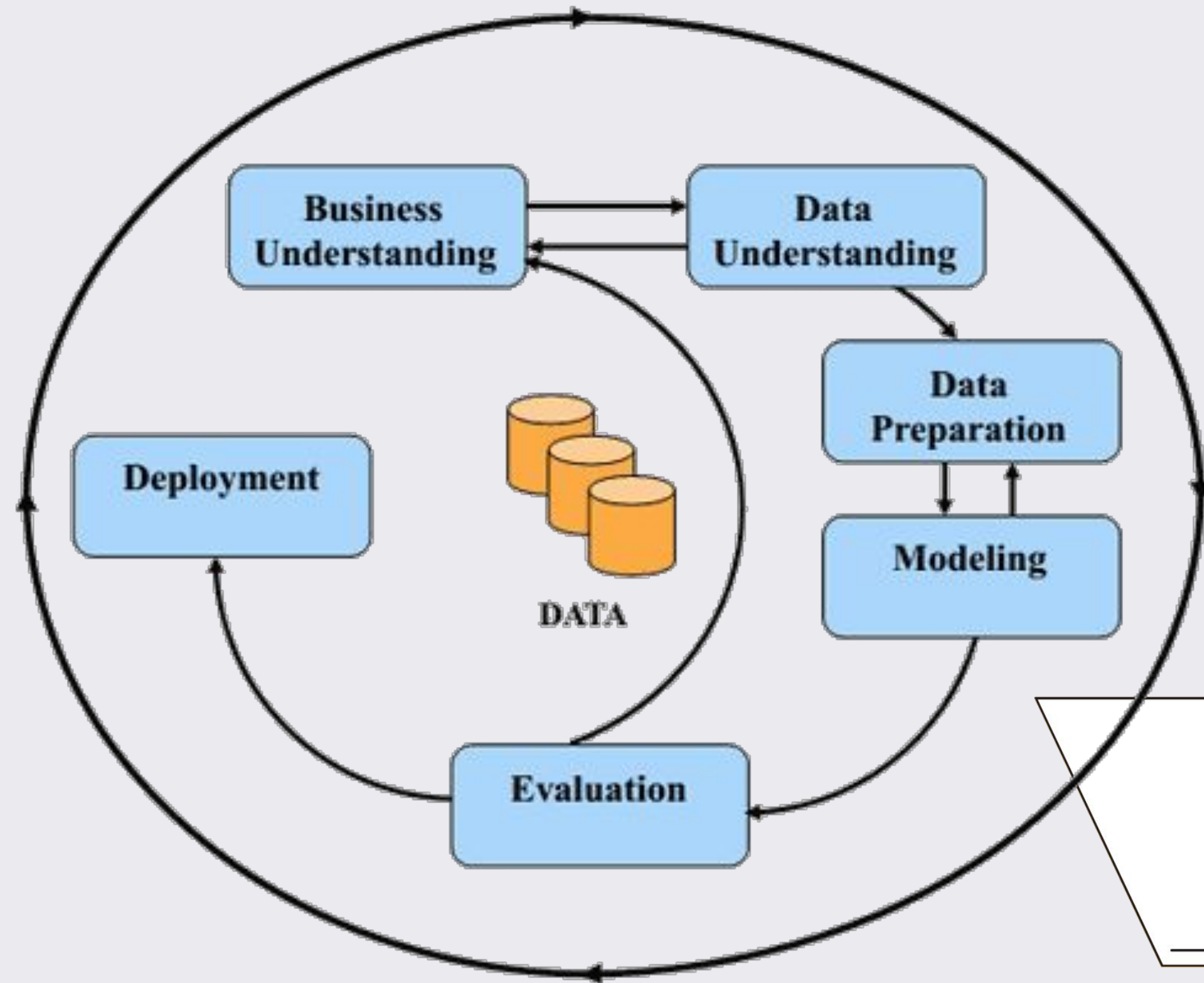
CRISP-DM provides an overview of the data mining life cycle as a process model. The life cycle model comprises six phases, with arrows indicating the most important and frequent dependencies between phases. The sequence of the phases is not strict. And most projects move back and forth between phases as necessary. The CRISP-DM model is flexible and can be customized easily.

For example, if your organization aims to detect money laundering, you will likely sift through large amounts of data without a specific modelling goal. Instead of modelling, your work will focus on data exploration and visualization to uncover suspicious patterns in financial data. CRISP-DM allows you to create a data mining model that fits your needs.

It includes descriptions of typical phases of a project, the tasks involved with each phase, and an explanation of the relationships between these tasks.



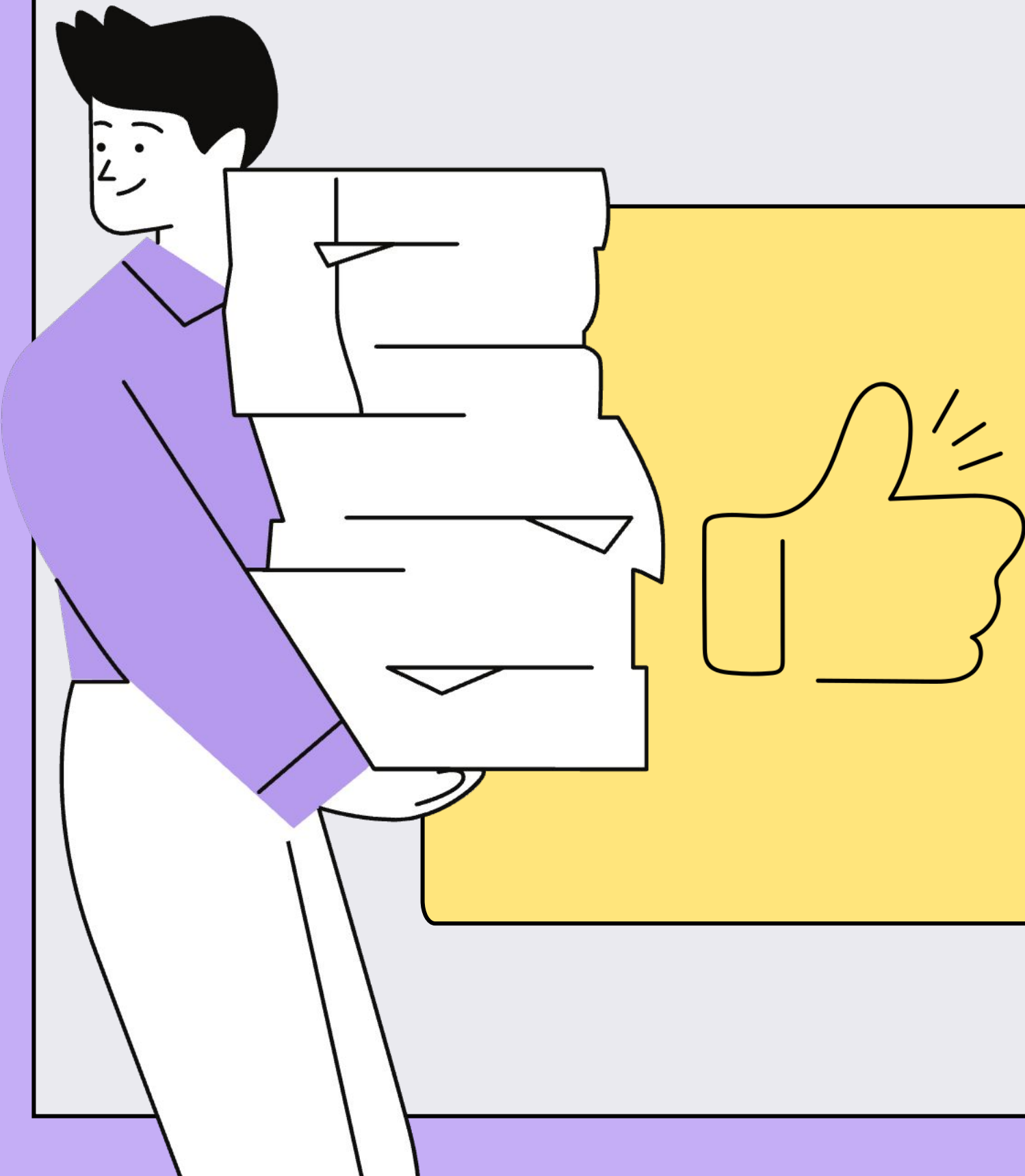
Phases of CRISP-DM



Business understanding

This initial phase focuses on understanding the project goals, objectives, and requirements from a business perspective, then converting this knowledge into a data mining problem definition and a preliminary plan with a specific set of tasks and desired outcomes to achieve project-level objectives

Data Understanding



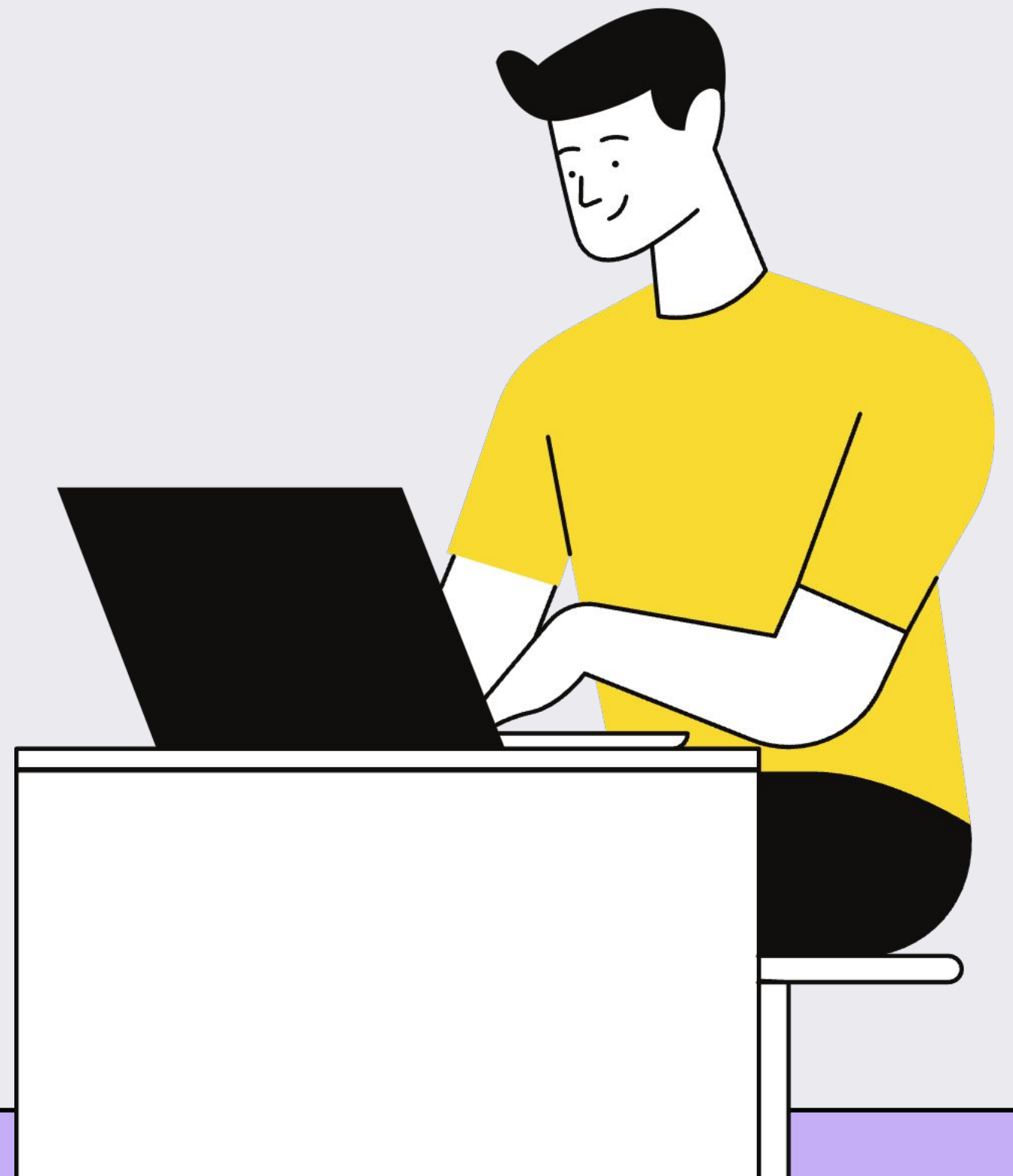
The data understanding phase starts with initial data collection and proceeds with activities such as feature description, primary data analysis, and exploratory data analysis that enable you to become familiar with the data, identify the data quality problems such as missing values, inconsistent data entries, and/or identify compelling subsets to form a hypothesis regarding confidential information.

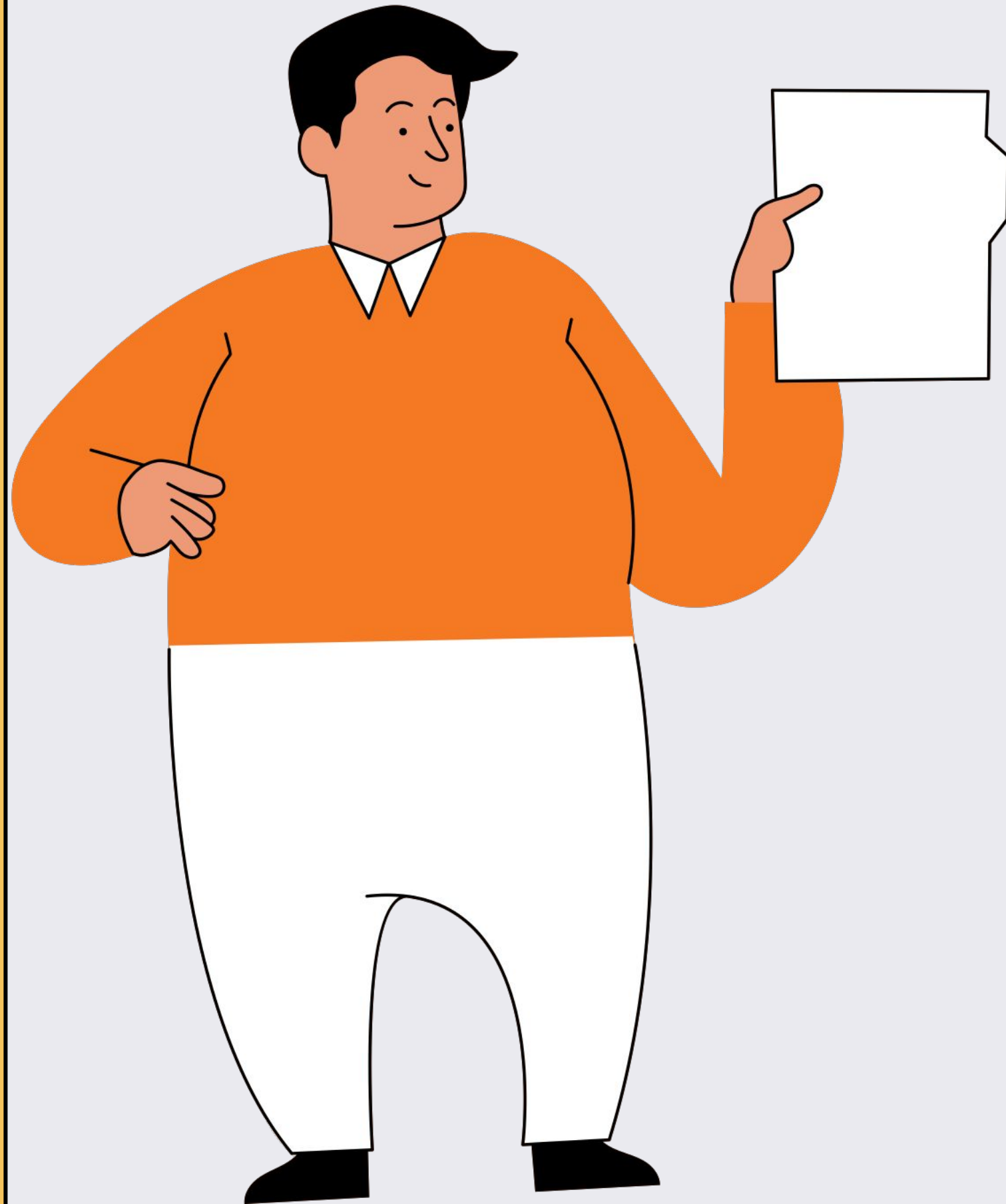
Data Preparation

The data preparation phase covers all activities needed to construct the final dataset fed into the modeling tools from the initial raw data.

Data preparation tasks are likely to be performed multiple times and not in any prescribed order.

Tasks include data table, feature selection, feature engineering, as well as feature transformation and cleaning of data for modeling tools and techniques.





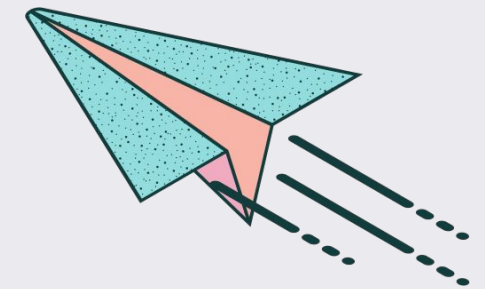
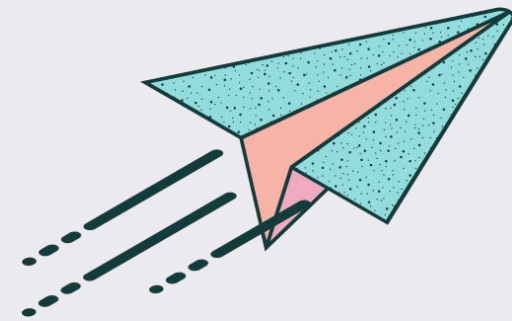
MODELLING

Widely regarded as data science's most exciting work is also often the shortest phase of the project

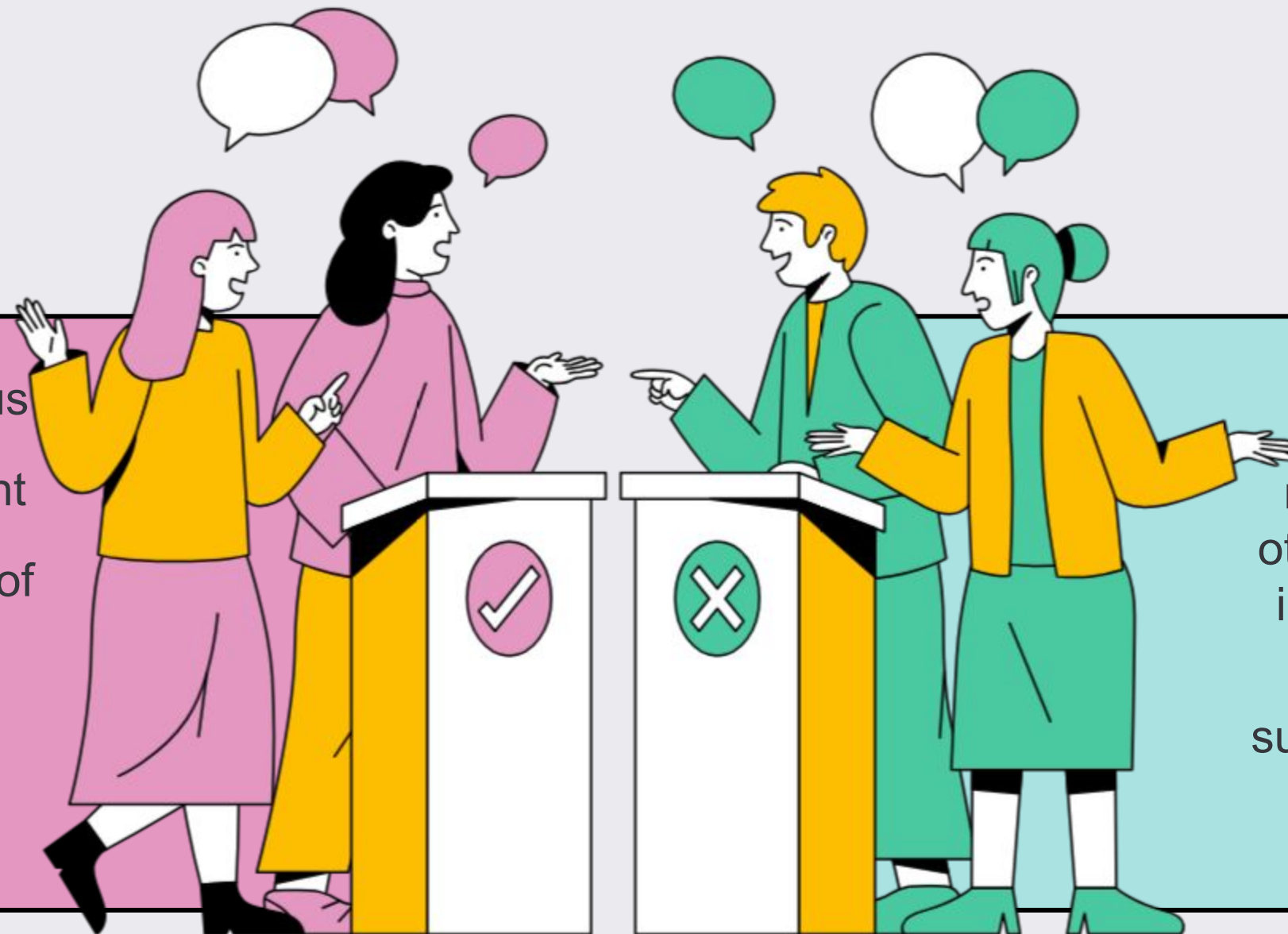
1. **Select modeling techniques:** Determine which algorithms to try (e.g. regression, neural net).
2. **Generate test design:** Pending your modeling approach, you might need to split the data into training, test, and validation sets.

MODELLING

Let's practice creating some arguments!



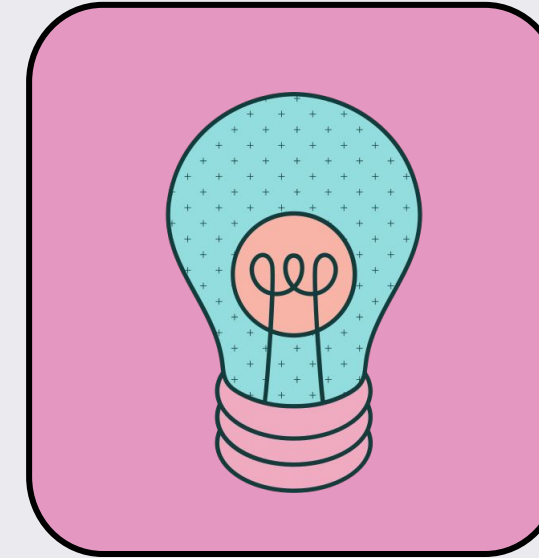
3. Build model: As glamorous as this might sound, this might just be executing a few lines of code like “reg = LinearRegression().fit(X, y)”.



Assess model: Generally, multiple models are competing against each other, and the data scientist needs to interpret the model results based on domain knowledge, the pre-defined success criteria, and the test design..

EVALUATION

Whereas the *Assess Model* task of the *Modeling* phase focuses on technical model assessment, the *Evaluation* phase looks more broadly at which model best meets the business and what to do next. This phase has three tasks:



1. **Evaluate results:** Do the models meet the business success criteria? Which one(s) should we approve for the business?



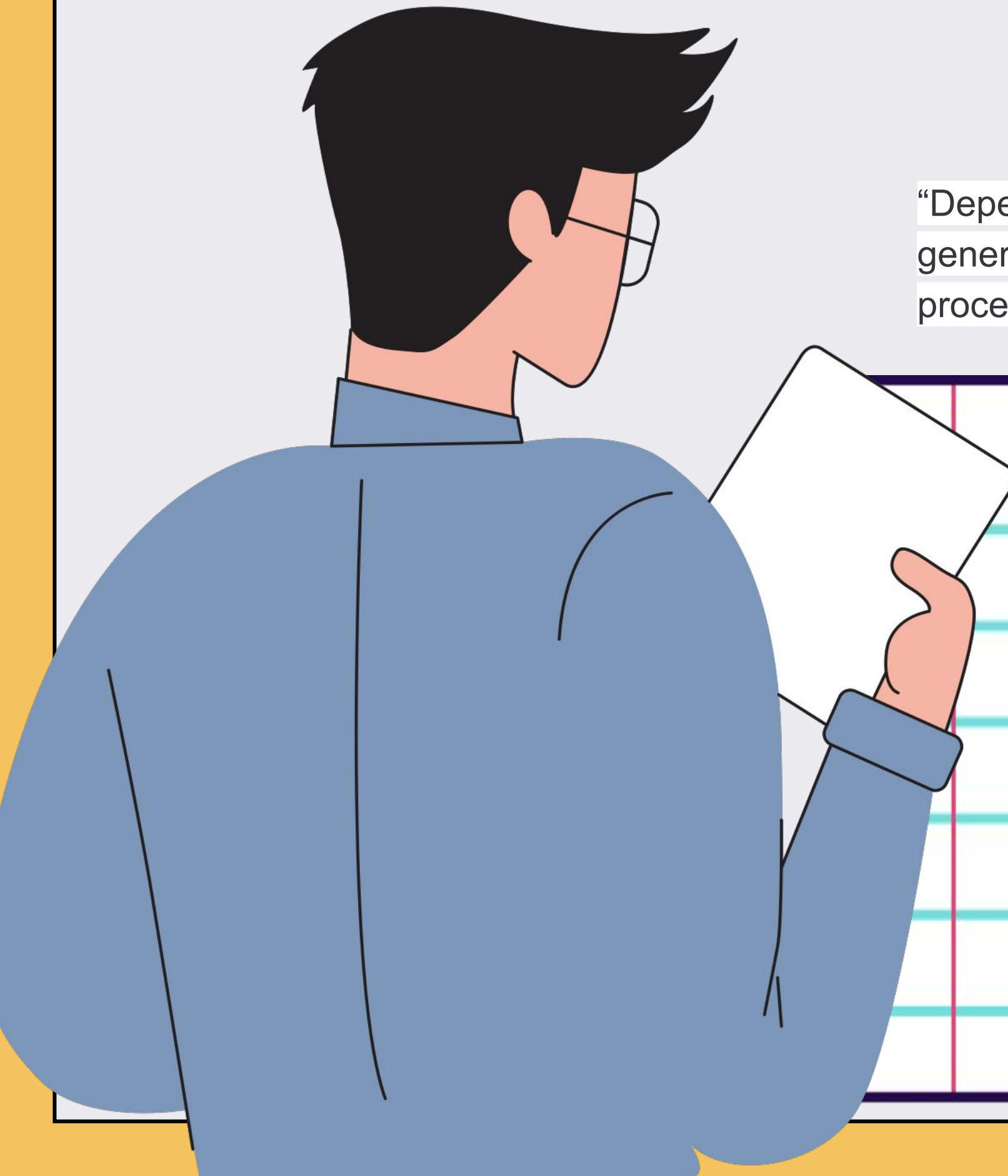
2. **Review process:** Review the work accomplished. Was anything overlooked? Were all steps properly executed? Summarize findings and correct anything if needed.

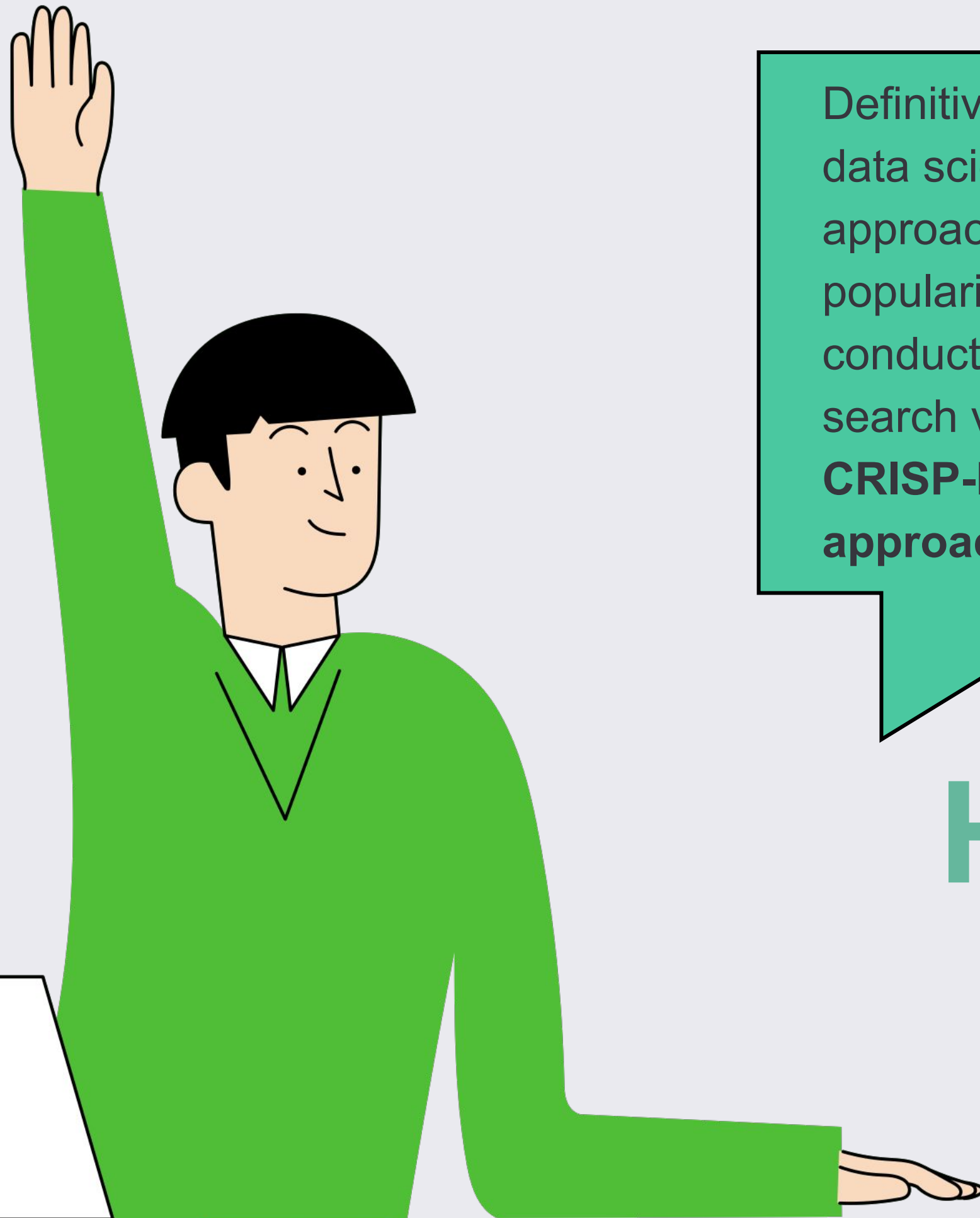


3. **Determine next steps:** Based on the previous three tasks, determine whether to proceed to deployment, iterate further, or initiate new projects.

DEPLOYMENT

“Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process across the enterprise.”

- 
1. **Plan deployment:** Develop and document a plan for deploying the model
 2. **Plan monitoring and maintenance:** Develop a thorough monitoring and maintenance plan to avoid issues during the operational phase (or post-project phase) of a model
 3. **Produce final report:** The project team documents a summary of the project which might include a final presentation of data mining results.
 4. **Review project:** Conduct a project retrospective about what went well, what could have been better, and how to improve in the future.



Definitive research does not exist on how frequently data science teams use different management approaches. So to get an idea on approach popularity, we investigated KDnuggets polls, conducted our own poll, and researched Google search volumes. Each of these views suggests that **CRISP-DM is the most commonly used approach** for data science projects.

HOW POPULAR IS CRISP DM?

IS CRISP DM AGILE OR WATERFALL?

Answer

Technology pushes us further apart.

Reason 1

Technology keeps us from spending quality time with loved ones.

Reason 2

Seeing perfect lives online can make us feel insecure or unhappy.

Reason 3

Too much technology makes us less active and more unhealthy.

Thesis Statement

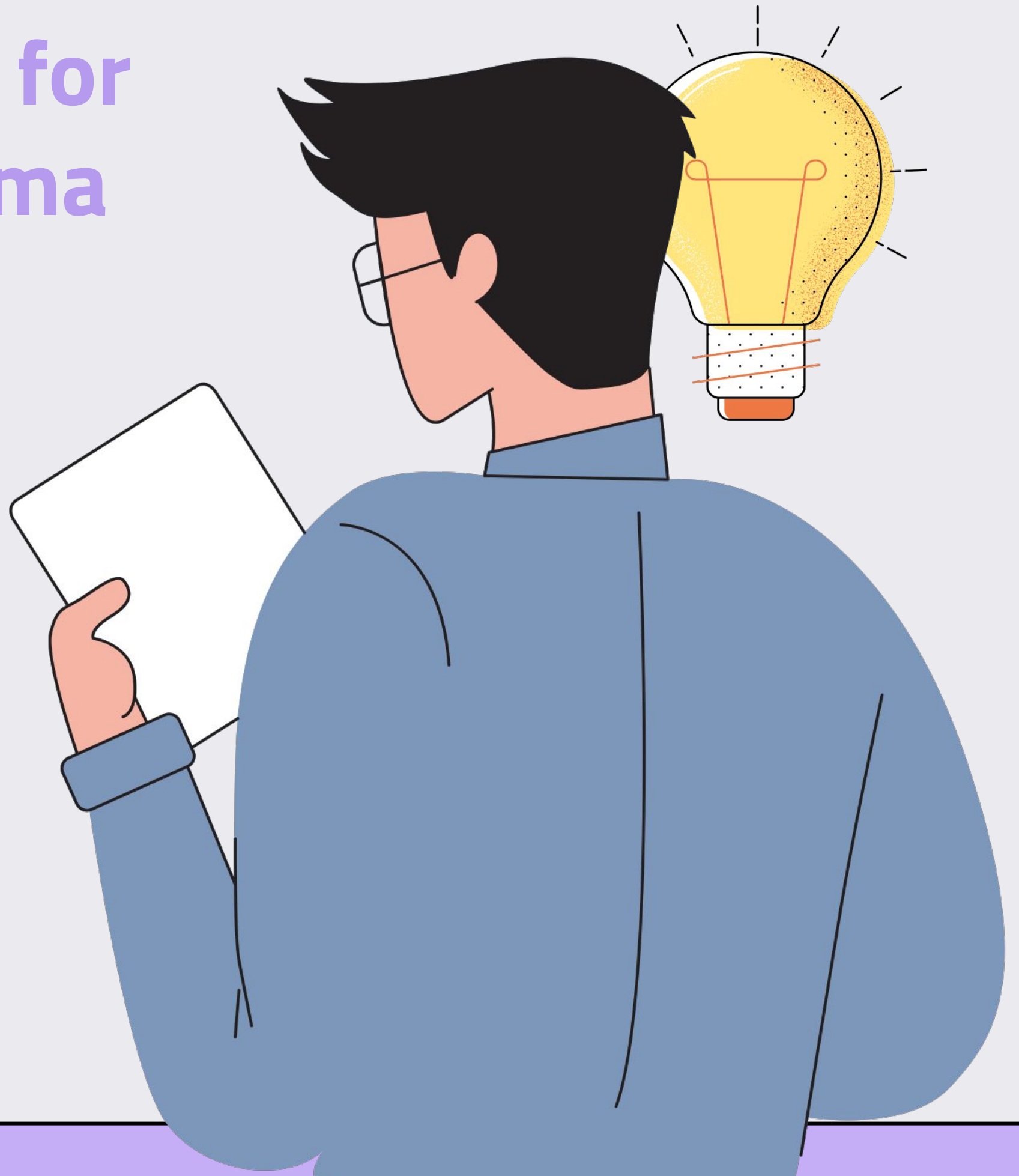
Technology pushes us further apart as **it keeps us from spending quality time with loved ones**, **it can make us feel insecure or unhappy**, and **it can make us less active and more unhealthy**.



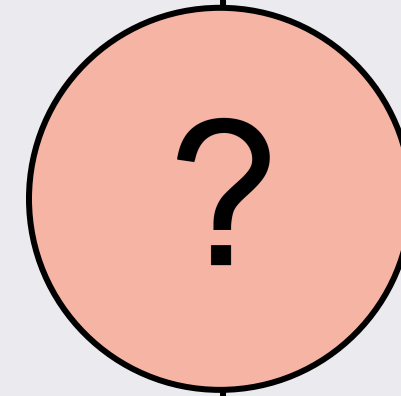
Homogeneous Ontology for Recursive Uniform Schema (HORUS)

Homogeneous Ontology for Recursive Uniform Schema (HORUS)

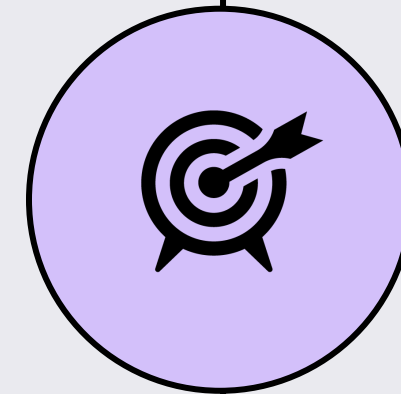
**A Unified Framework for Data
Transformation and
Integration**



Introduction to HORUS



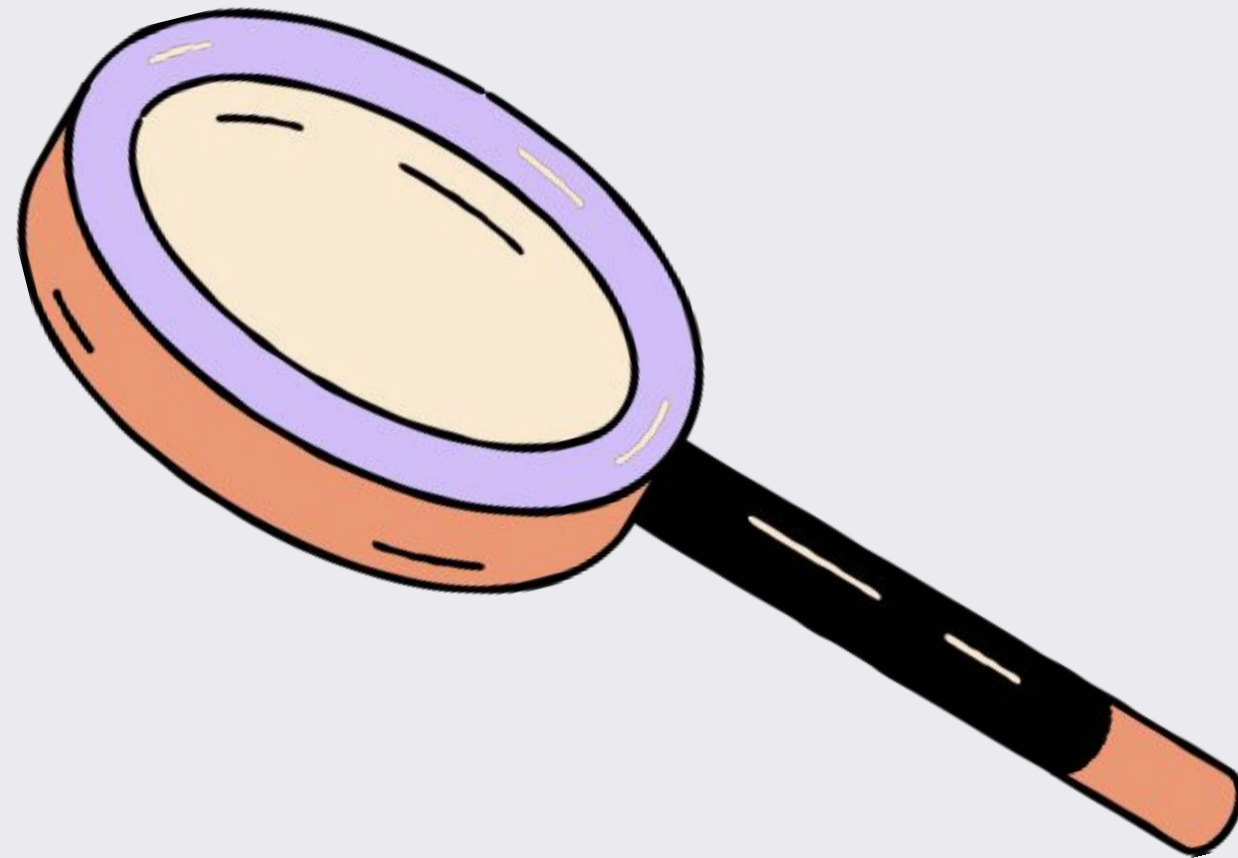
What is HORUS?



Goal of HORUS



Why is HORUS
Important?



What is HORUS?

Key Focus: Uniform representation across all datasets.

HORUS is a **data science framework** designed to create a **standardized, unified data format**.

It transforms **diverse external data sources** into a **single, consistent structure**.

By providing a unified schema, HORUS simplifies the process of **handling and analyzing data** from multiple sources.

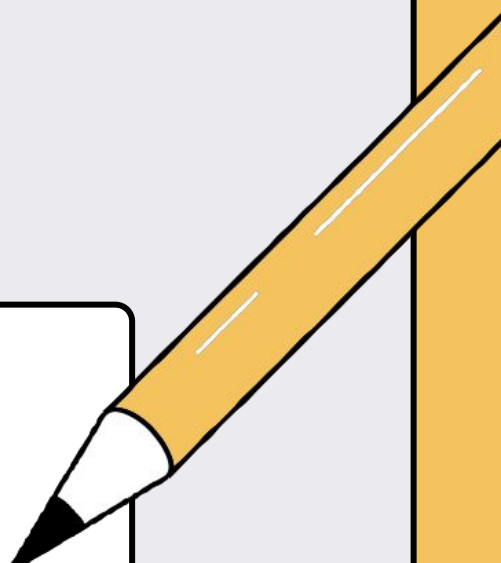
Goal of HORUS

Key Focus: Efficiency and integration across the data lifecycle.

Simplify Data Analysis and Integration

- Reduce the complexity of working with multiple formats.
- Streamline the analysis process.

Act as a Central "Hub"

- Serve as the **core transformation layer** in a data pipeline.
 - Enable seamless movement of data between **sources, processing tools, and outputs.**
- 

Why is HORUS Important?

Answer

Addressing modern data challenges effectively.

Modern Data Complexity:

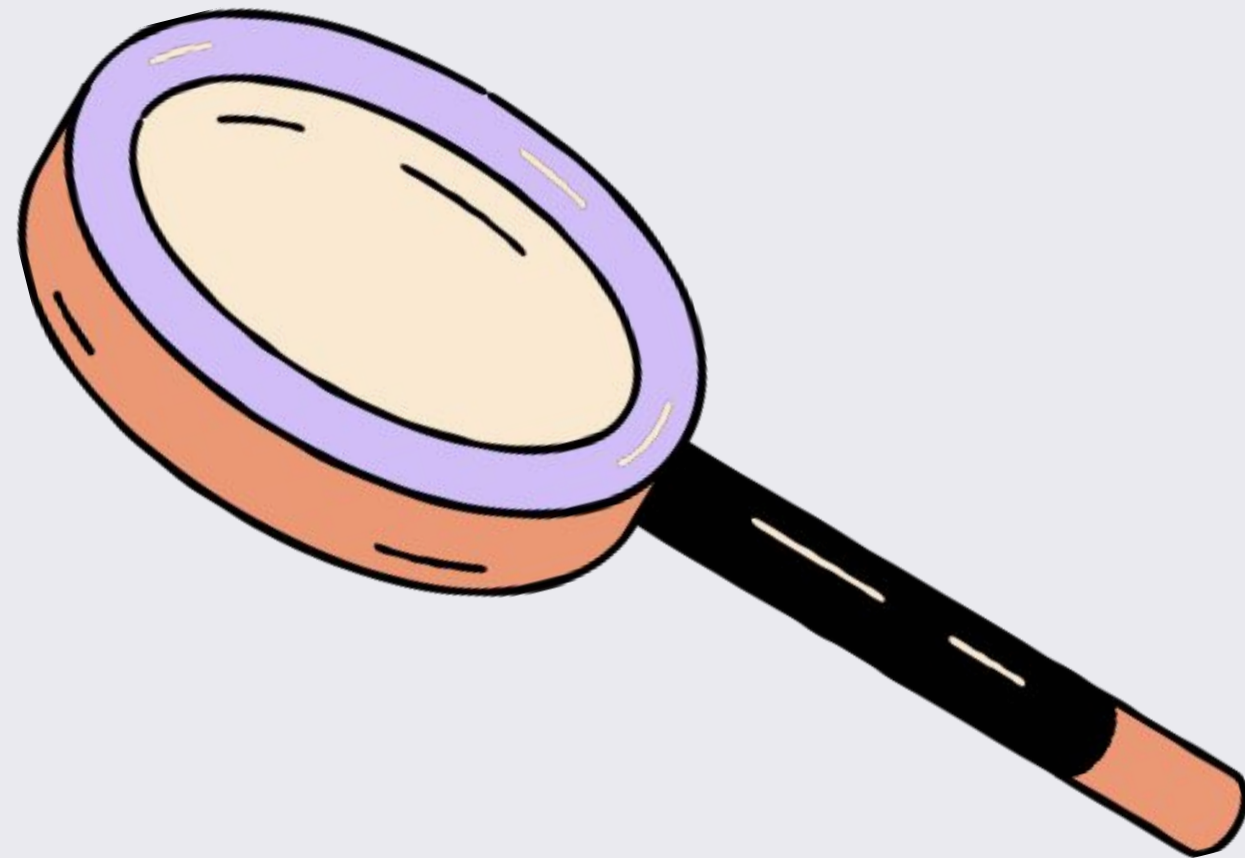
Datasets today are **vast, diverse, and originate from heterogeneous sources** (e.g., APIs, databases, files).

Challenges in Standardization:

Handling **different structures and formats** often requires custom logic and mappings.

HORUS Advantage:

- **Standardized schema** reduces complexity.
- **Accelerates insights** by minimizing manual intervention.



Key Aspects of HORUS?

Uniformity

- Ensures data is represented in a consistent schema.
- Eliminates the need for complex mapping between formats.

Recursive Structure

- Supports nested data structures and hierarchical relationships.
- Represents complex relationships accurately.

Ontology-based

- Defines clear relationships and semantics between data elements.
- Enhances data understanding and interoperability.

How HORUS Works ?

Step 1: Data Ingestion

- Raw data from **diverse sources** is ingested.
- Transformed into the **HORUS format** through data-format-to-HORUS transformation.

Step 2: Data Processing

- Data in HORUS format is **manipulated, analyzed, and integrated** seamlessly.
- Supports advanced analytics and workflows.

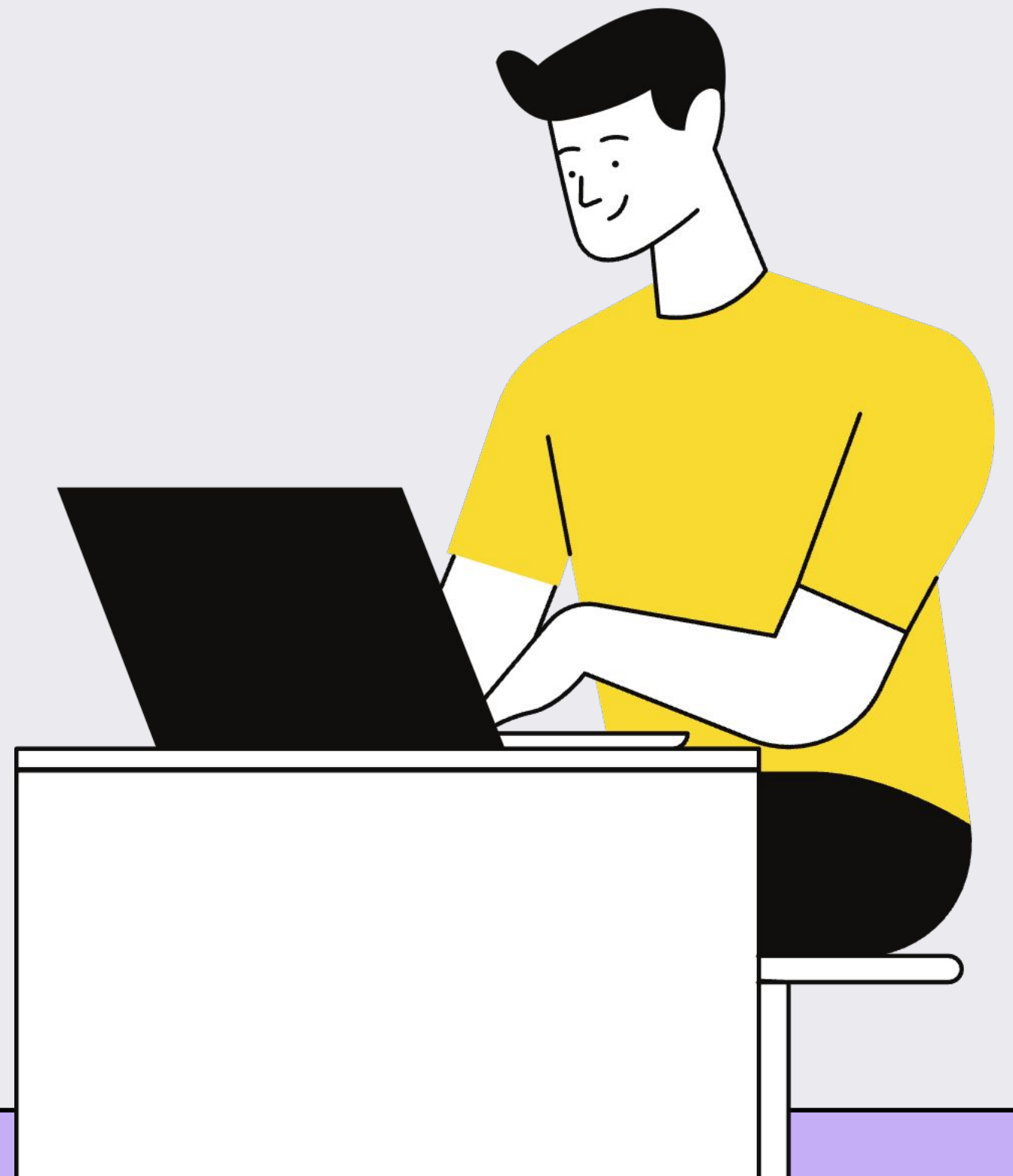
Step 3: Output Generation

- Processed data can be converted back to **external formats** if needed.
- Ensures compatibility with downstream tools and systems.

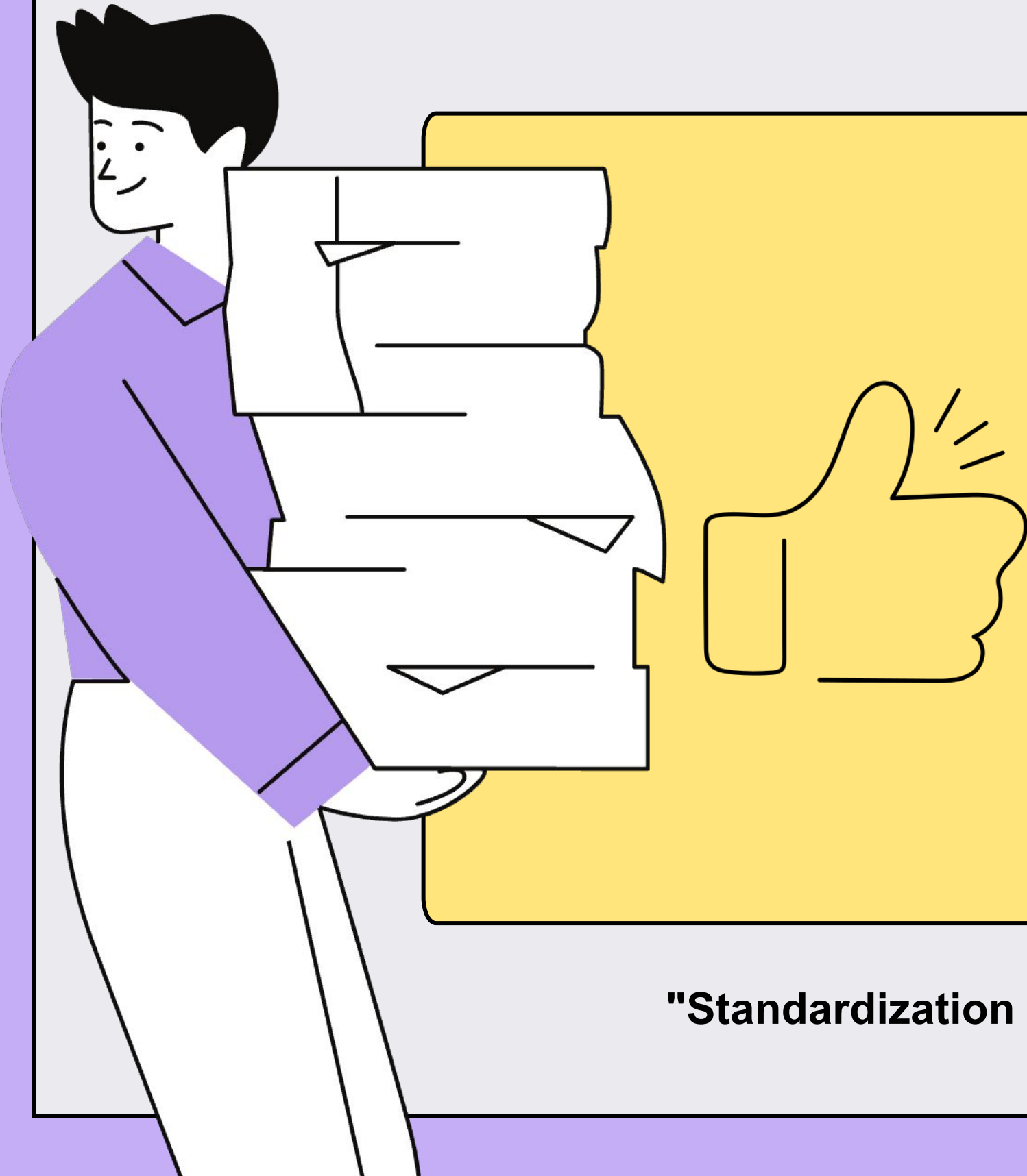


Benefits of Using HORUS

- **Simplified Data Integration:** Combines data from multiple sources with minimal friction.
- **Reduced Development Time:** Less need for **custom transformation logic**.
- **Improved Data Quality:** Enforces a consistent structure for better data integrity.
- **Scalability:** Easily handles **large and complex datasets**.
- **Interoperability:** Supports integration across **different platforms and systems**.



Conclusion



- HORUS acts as a **unifying framework** for complex data ecosystems.
- Its **ontology-driven approach** ensures clarity and consistency.
- Ideal for domains like **e-commerce, healthcare, finance**, and more.
- Future-proof solution for data-driven organizations.

"Standardization is the key to unlocking the full potential of data science pipelines."

Example Code:

Homogeneous Ontology:

- A homogeneous ontology emphasizes **uniformity** in the representation of concepts and relationships within a schema.
- It ensures that all elements within the schema follow the same structure or template, making it easier to manage and query data across recursive or nested structures.

Recursive Uniform Schema:

- A recursive schema allows elements to reference themselves or be nested in a way that follows a defined pattern. For example, a directory structure where folders can contain files or other folders.
- Uniformity ensures that the same rules, attributes, and relationships apply consistently at every level of the recursion.

```
HORUS

{
  "name": "FileSystem",
  "type": "Directory",
  "size": 0,
  "contents": [
    {
      "name": "Documents",
      "type": "Directory",
      "size": 0,
      "contents": [
        {
          "name": "Work",
          "type": "Directory",
          "size": 0,
          "contents": []
        },
        {
          "name": "Personal",
          "type": "Directory",
          "size": 0,
          "contents": []
        }
      ]
    }
  ]
}
```

