A Brief Overview of Data Engineering

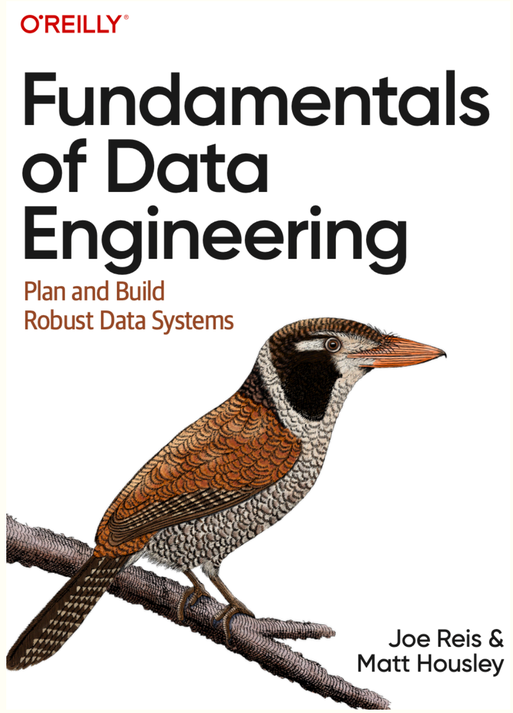
Rahul Dave (@rahuldave), Univ.Ai

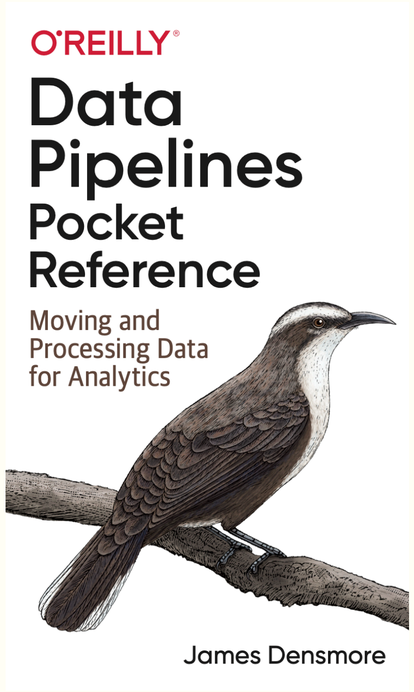
What is data engineering

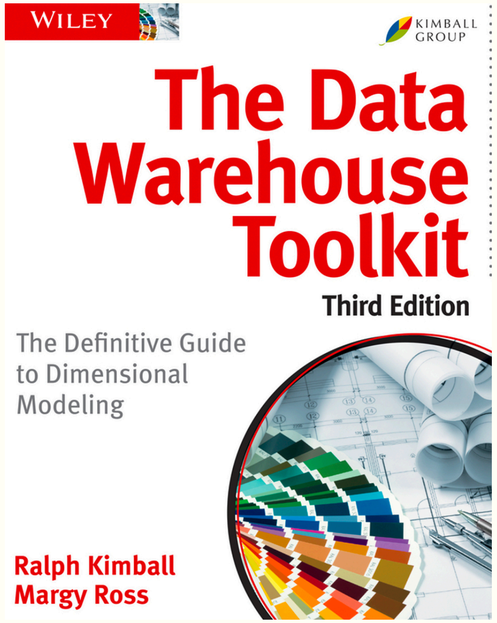
Data engineering is the development, implementation, and maintenance of systems and processes that take in raw data and produce high-quality, consistent information that supports downstream use cases, such as analysis and machine learning. Data engineering is the intersection of security, data management, DataOps, data architecture, orchestration, and software engineering. A data engineer manages the data engineering lifecycle, beginning with getting data from source systems and ending with serving data for use cases, such as analysis or machine learning.

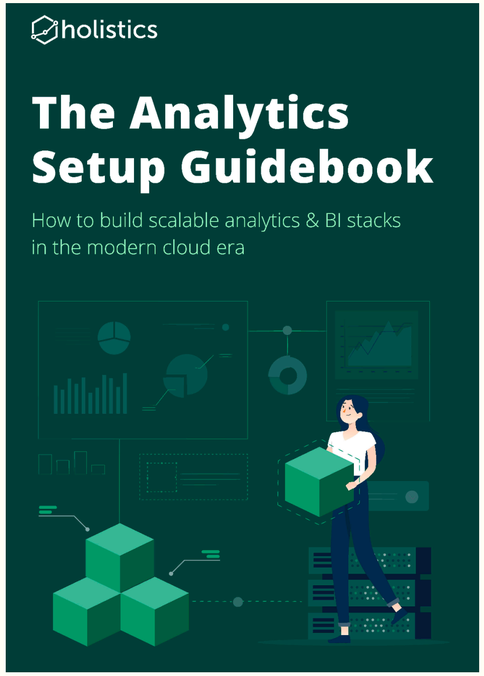
-Fundamentals of Data Engineering

Books to read



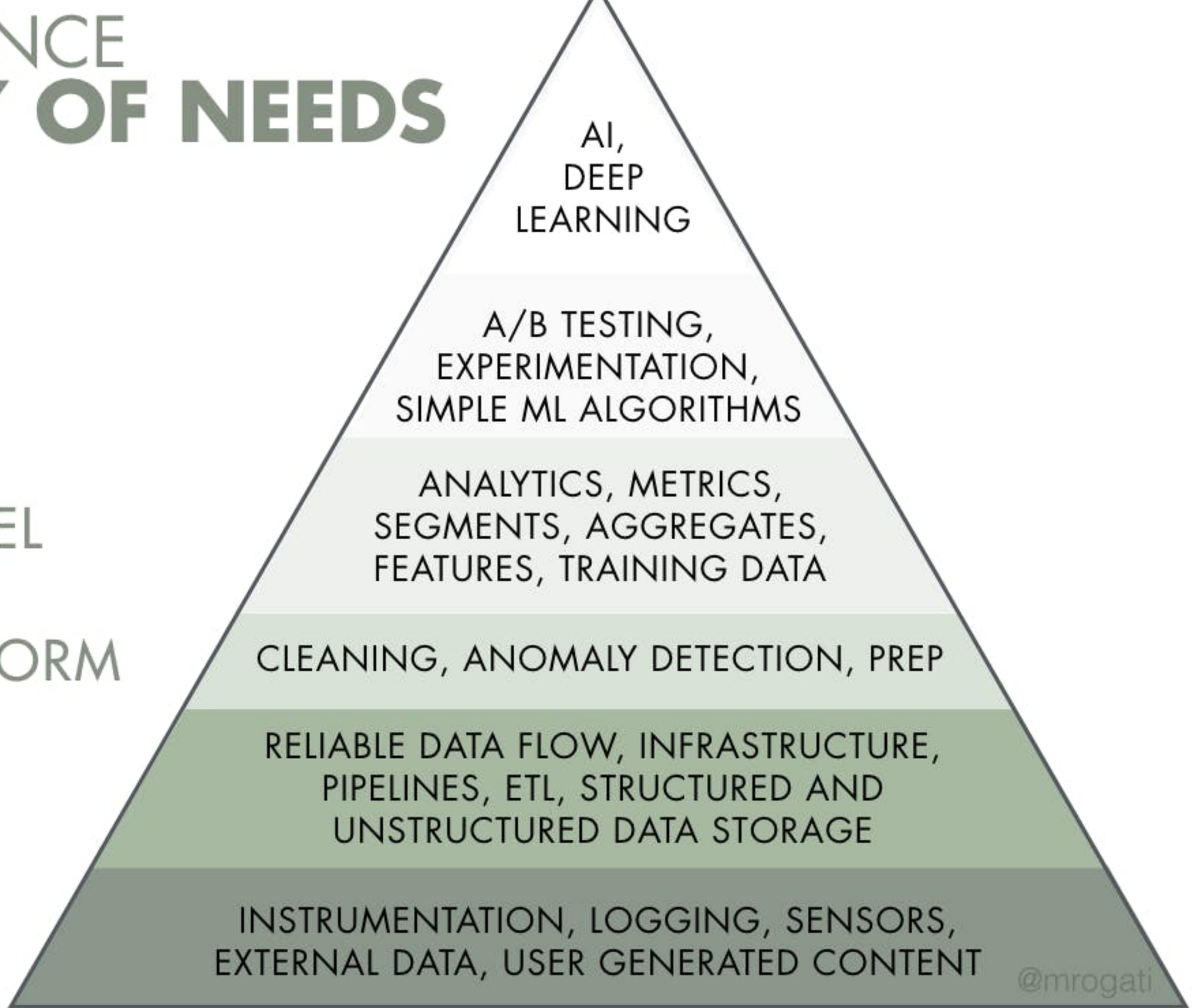






Univ.AI

THE DATA SCIENCE HIERARCHY OF NEEDS



LEARN/OPTIMIZE

AGGREGATE/LABEL

EXPLORE/TRANSFORM

MOVE/STORE

COLLECT

This bootcamp

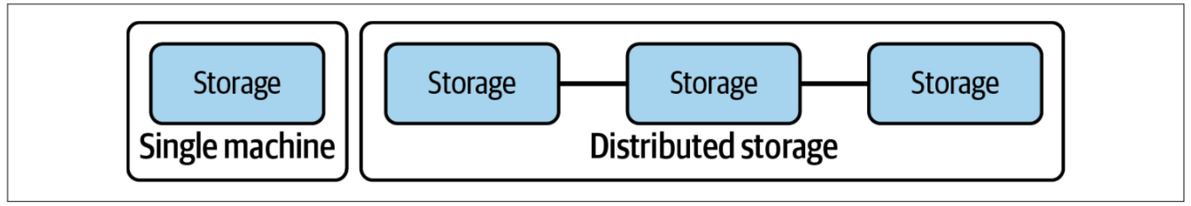
- will get you familiar with the data engineering landscape today by forcing you to think how we got here
- demands that you have python and sql/pands skills.
- has been made assuming you are either one of our students/alumni or have applied to one of our Data Engineering jobs and thus have 2-3 years experience in the field
- some parts will be hard: you will need to put in reading, thinking and doing work
- this iteration has no labs. Subsequent ones will. Still we have provided you work in notebooks, and homework. These are YOUR RESPONSIBILITY to do. Please do to get the most out of the bootcamp.

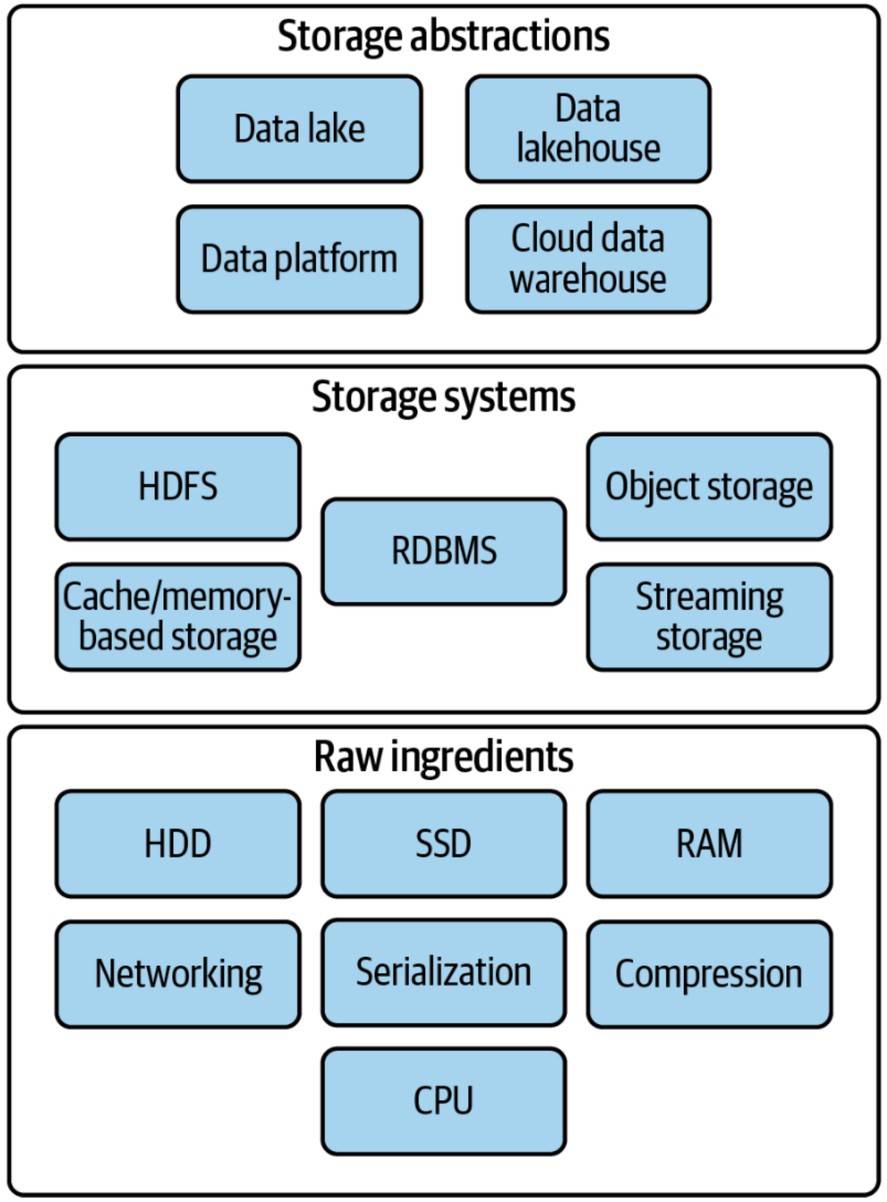
Bootcamp Structure

- Session 1: Historical Overview, OLTP, RDBMS, SQL, Normalization, and ACID
- Session 2: OLAP, Warehouses, Kimball Modeling, Modern warehouses, Columnar Databases, SQL for warehousing
- Session 3: Initial Data Lakes with Hadoop. File formats like Parquet. Warehouses as SQL against files, Spark. Cloud Setups.
- Session 4: Modern modeling using software development best practices. DAGs. Orchestration, dbt. BI and use facing analytics.

Storage Abstractions

- We have evolved from buiding **data warehouses** on RDBMS to columnar databases on-prem to cloud storage
- **Data Lakes** were originally created for Hadoop/HDFS systems, and are now backed with object storage storing any kind of file, with HDFS clusters being spun up on demand.
- **Data lakehouse** combines the warehouse with the lake by adding SQL querying, schema support, and metadata management.
- **Data platforms** add mahagement and catalog properties that allow us to create registries across al the data we have.





Data Engineering Lifecycle

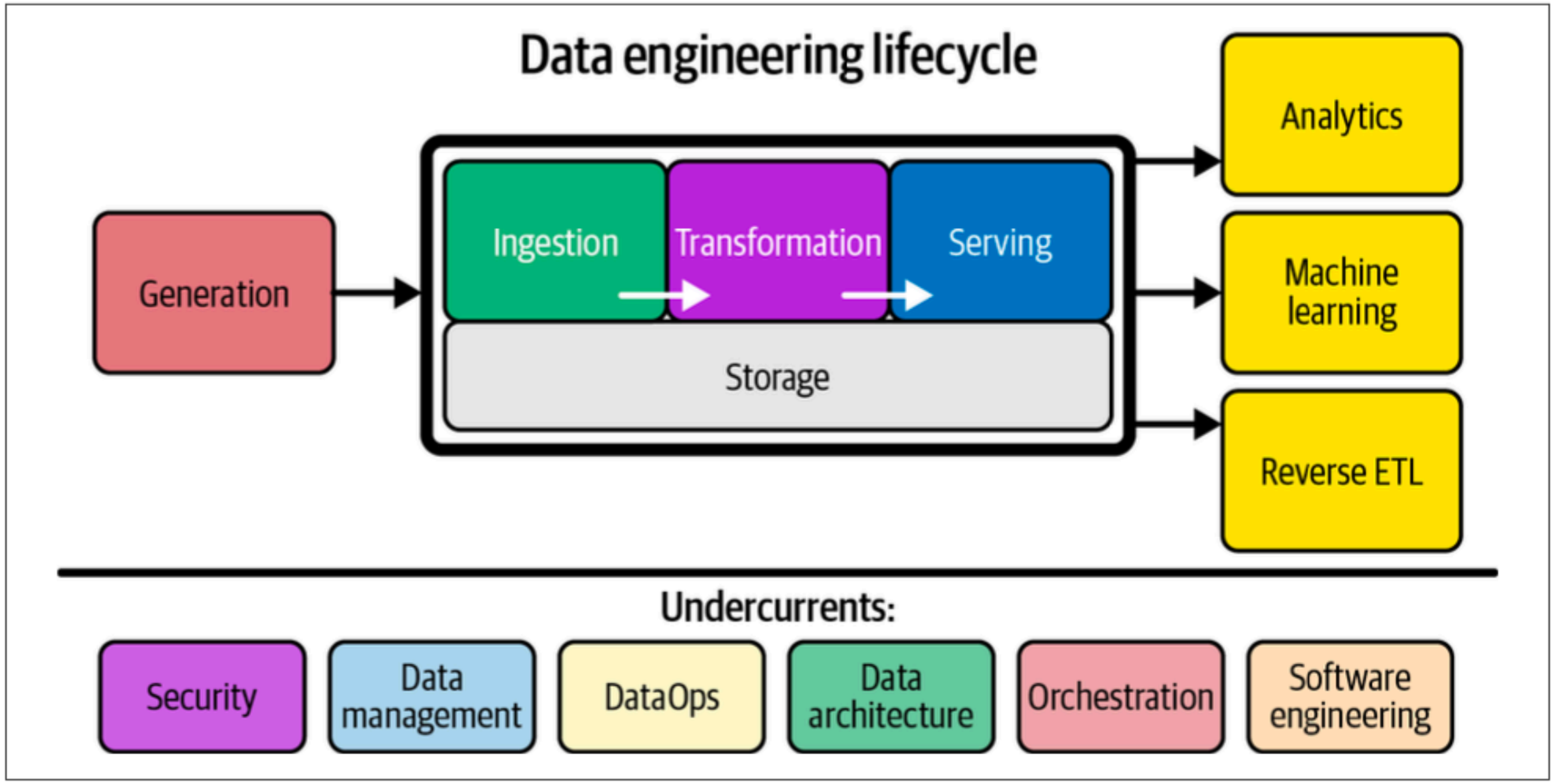
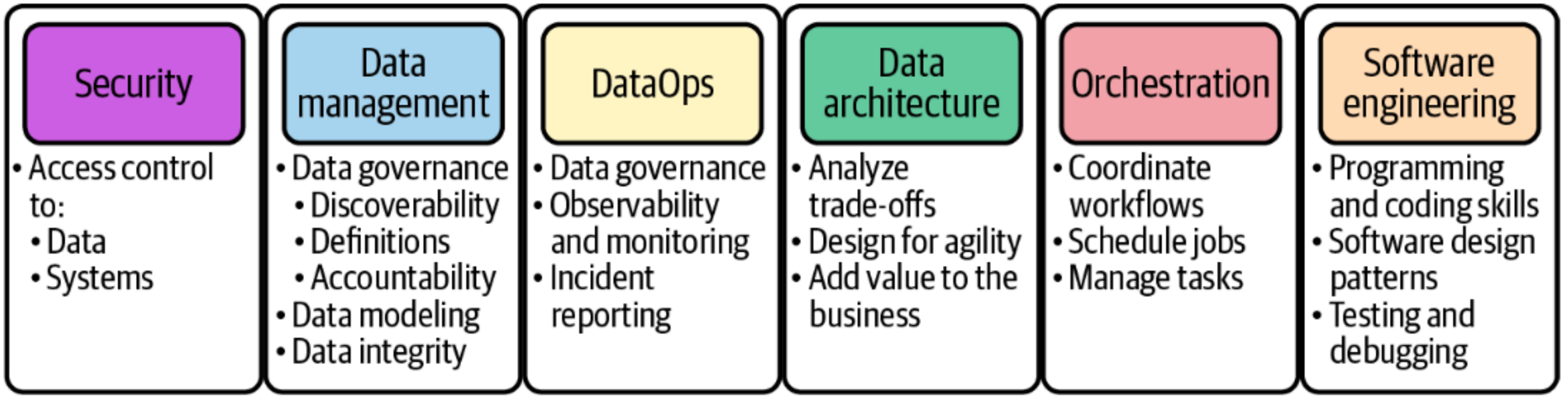


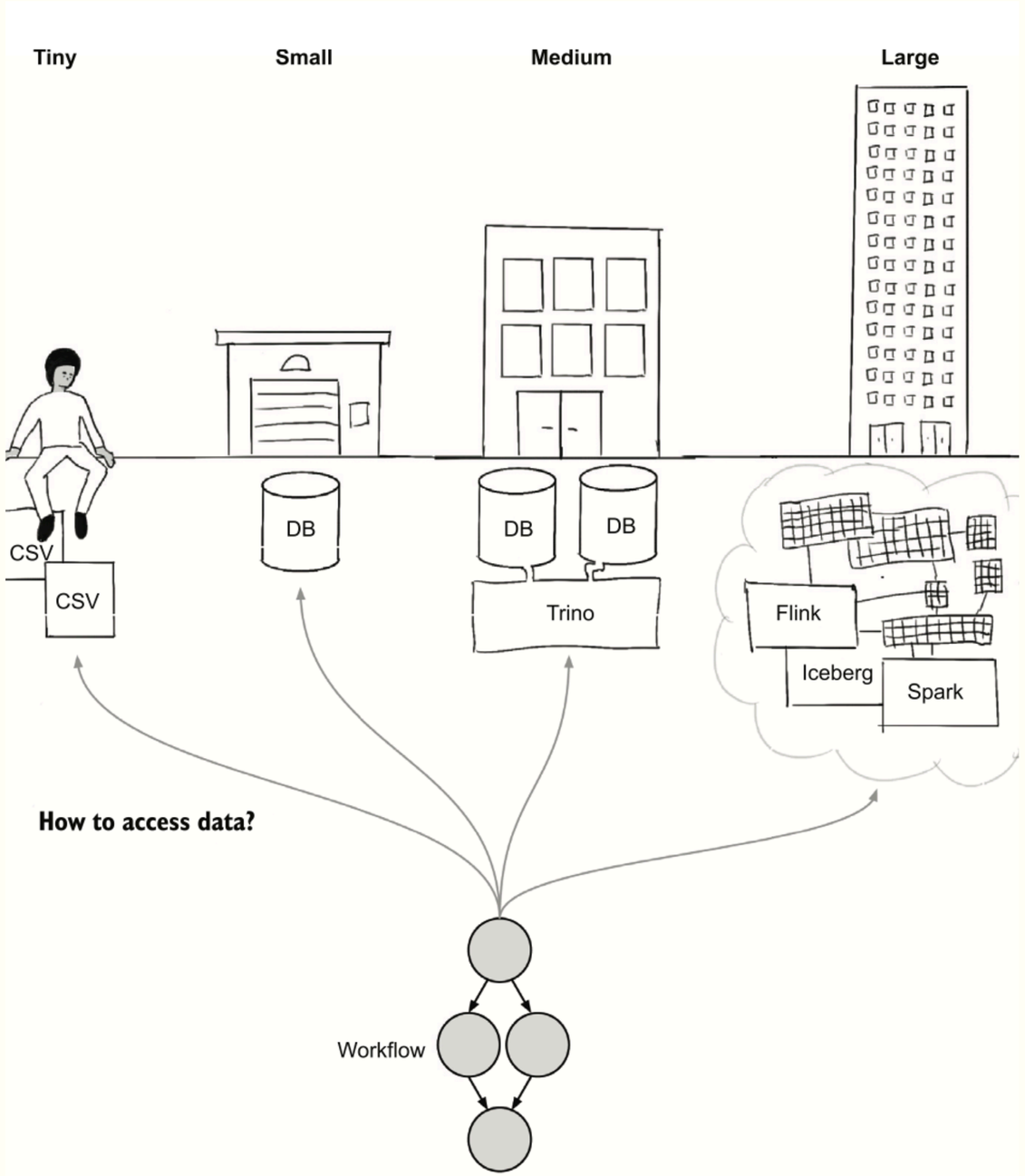
Figure 1-1. The data engineering lifecycle

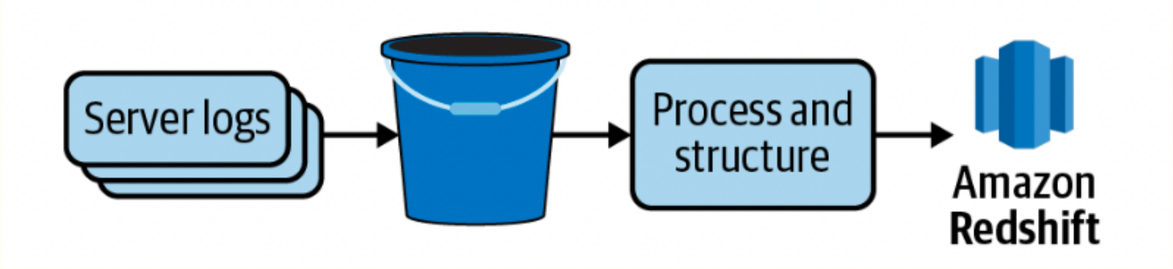
Undercurrents



Where is your data?

Does not matter if you can get to it
via SQL or API(Spark).





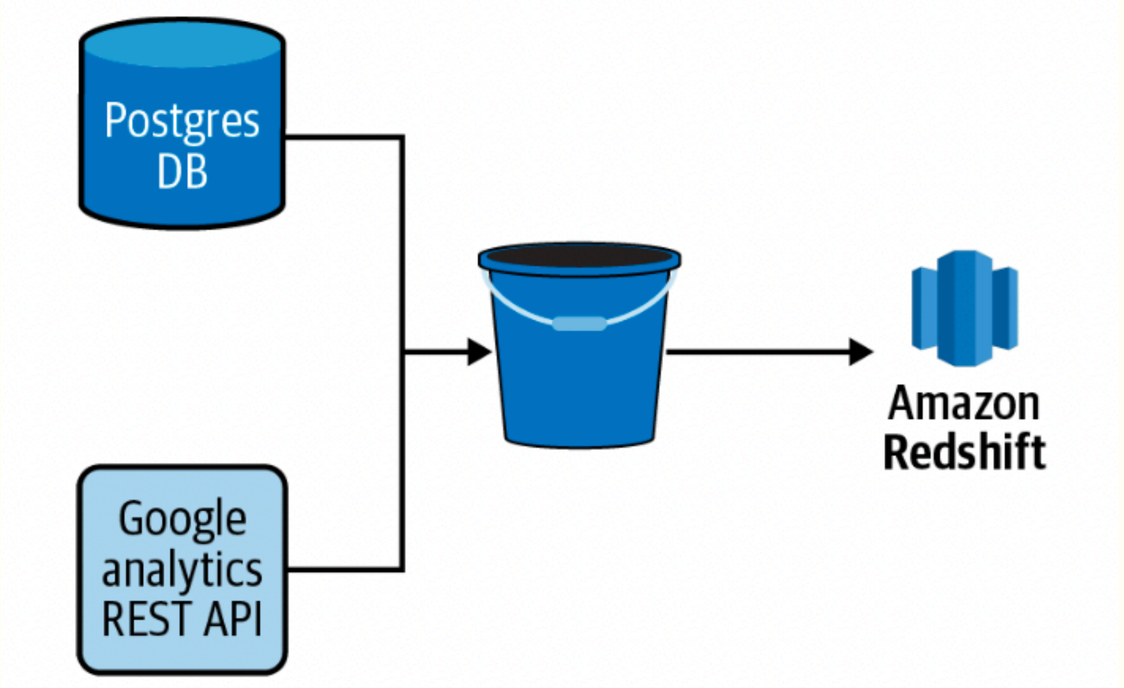
Where does data come from?

E-commerce databases

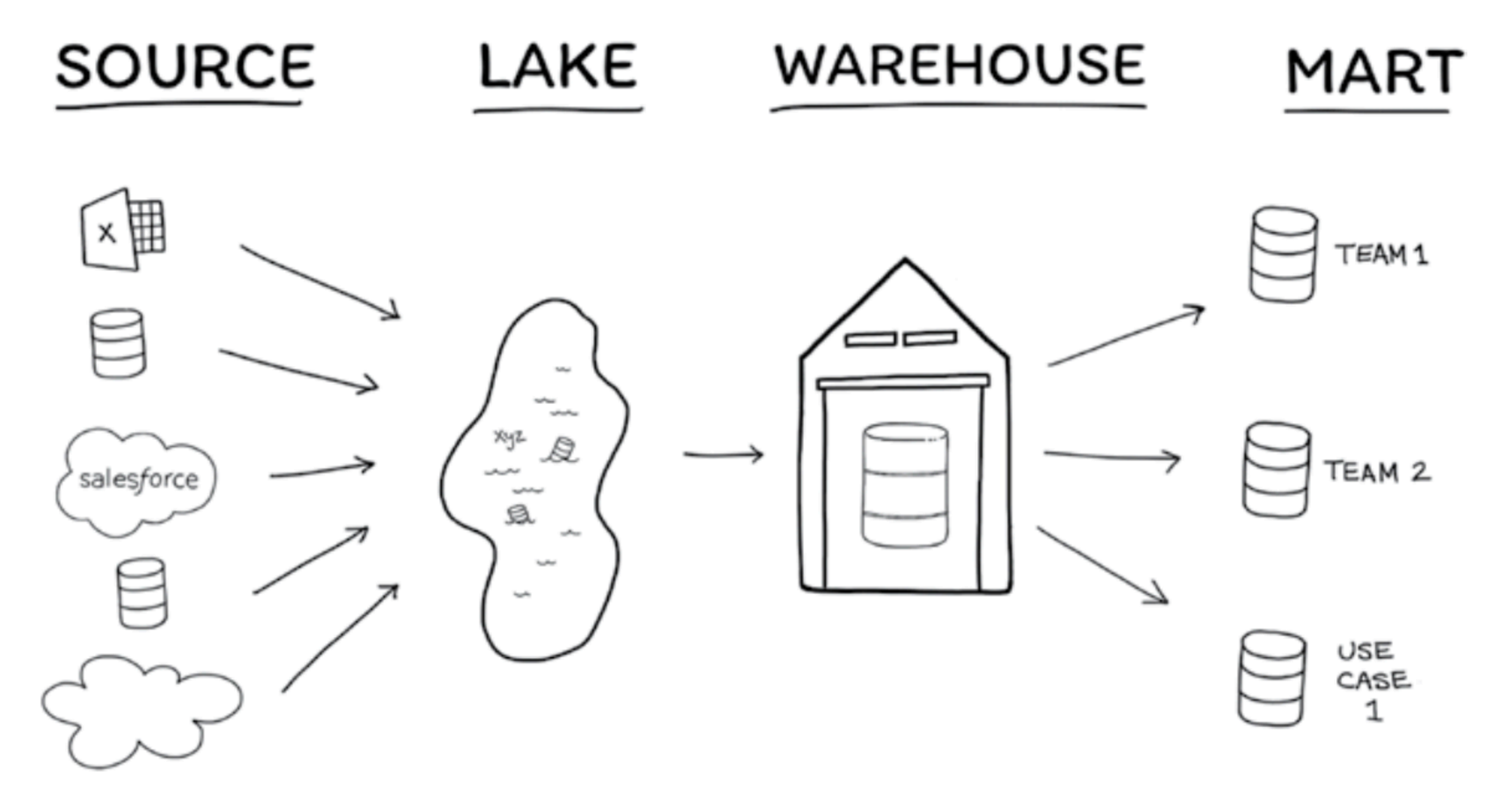
Logs

Customer tracking

Other sites

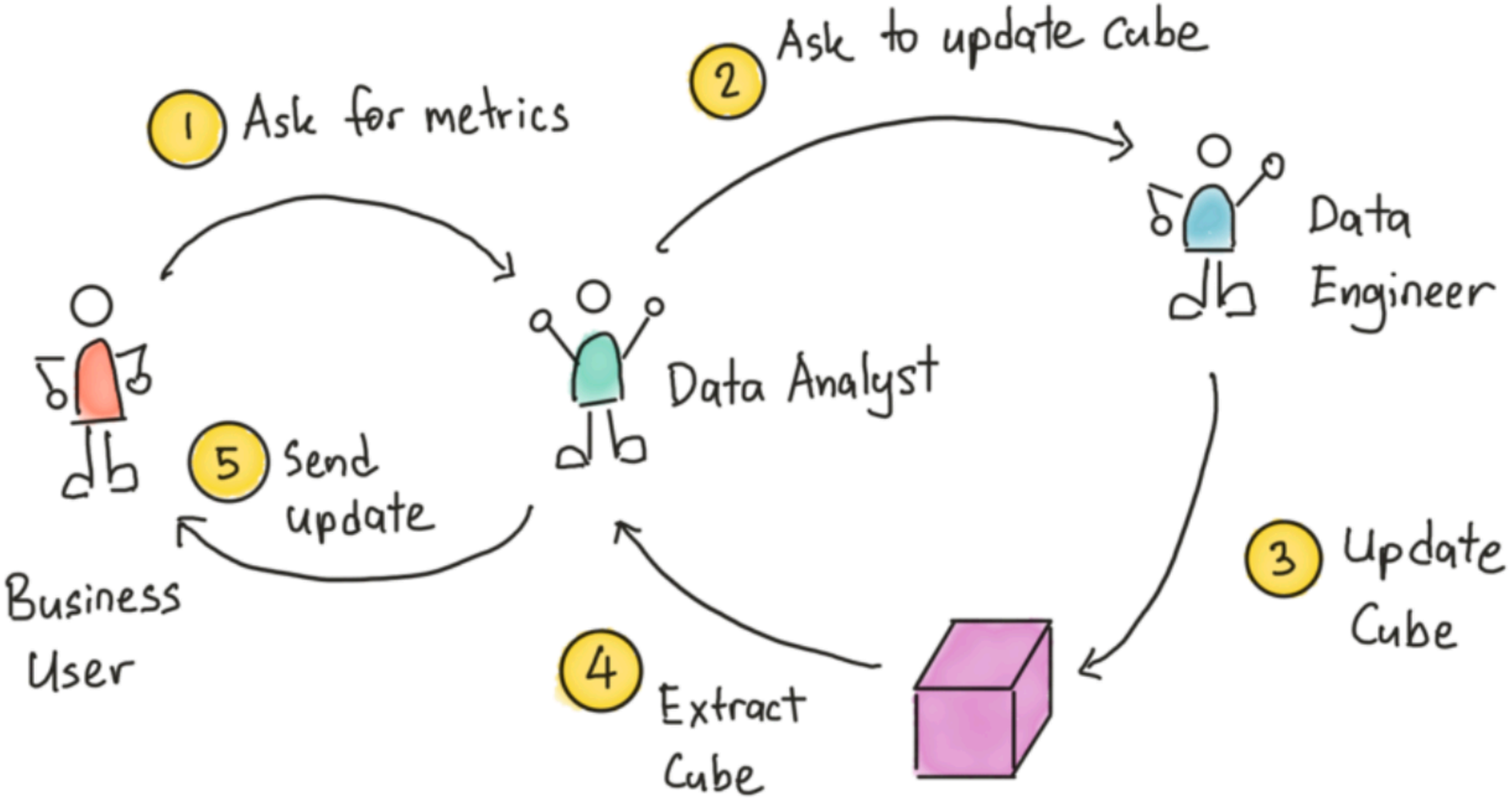


Components of a modern data platform

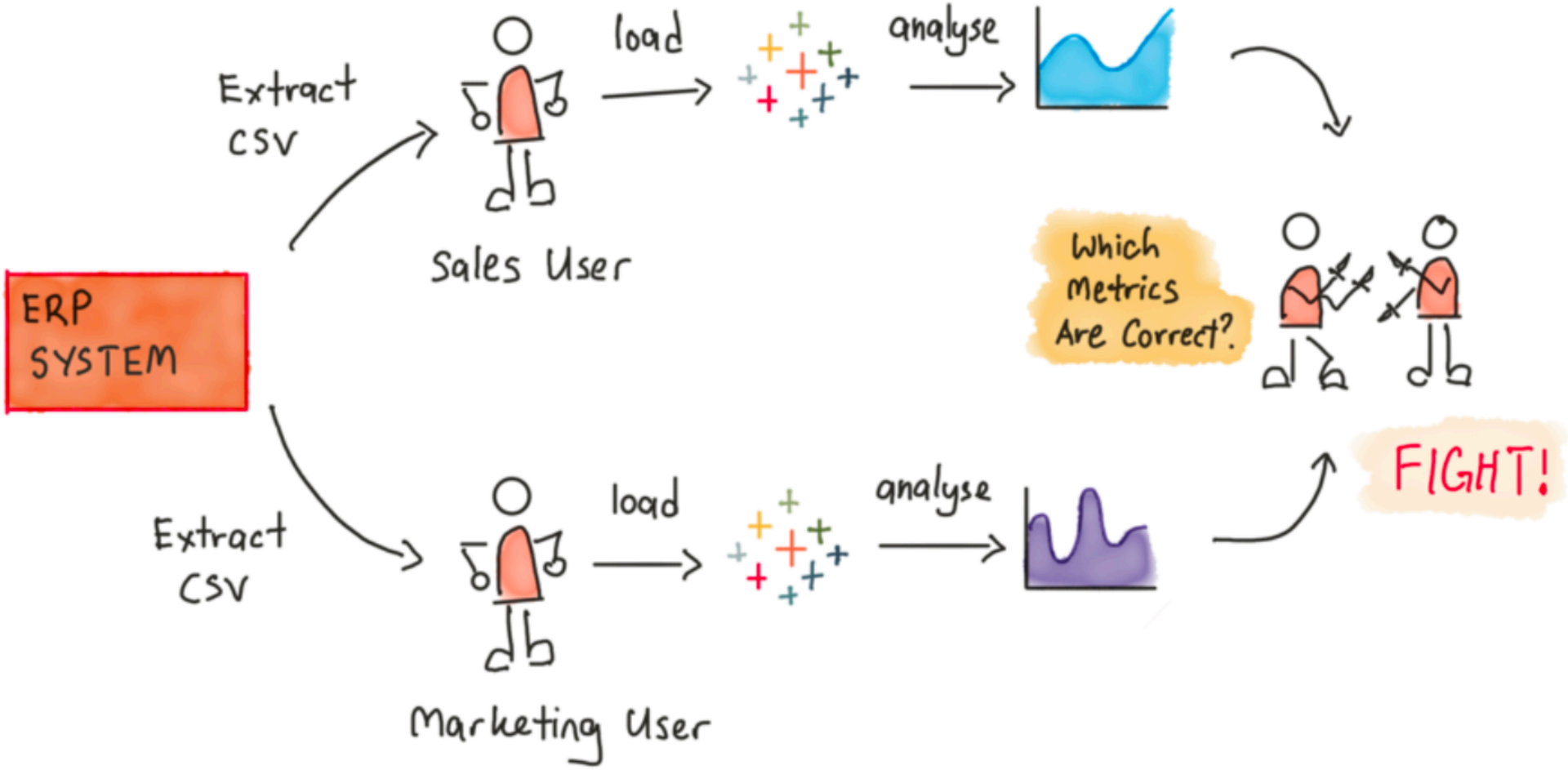


Daniel’s 3 jobs

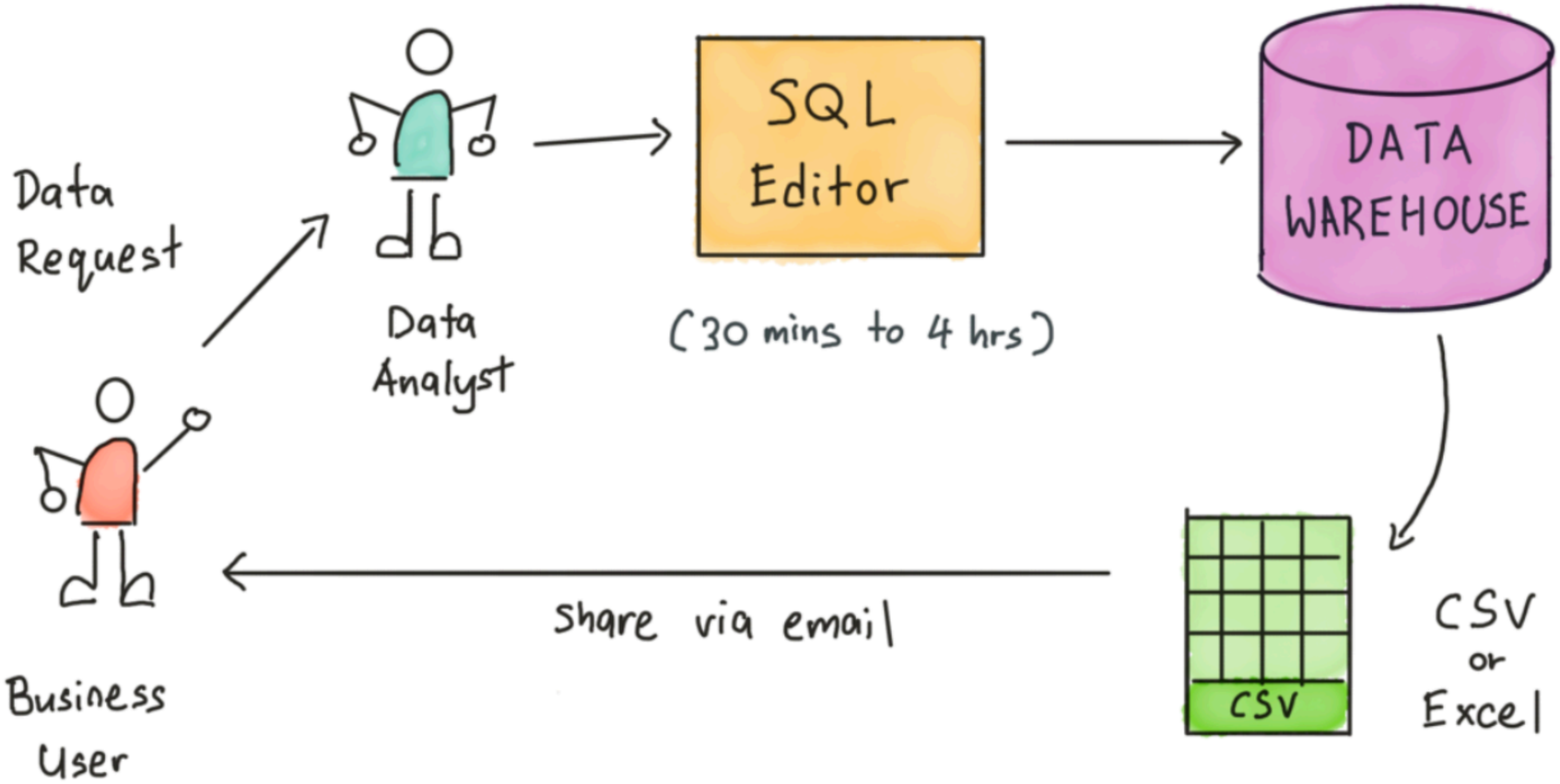
Back in the day



The second job

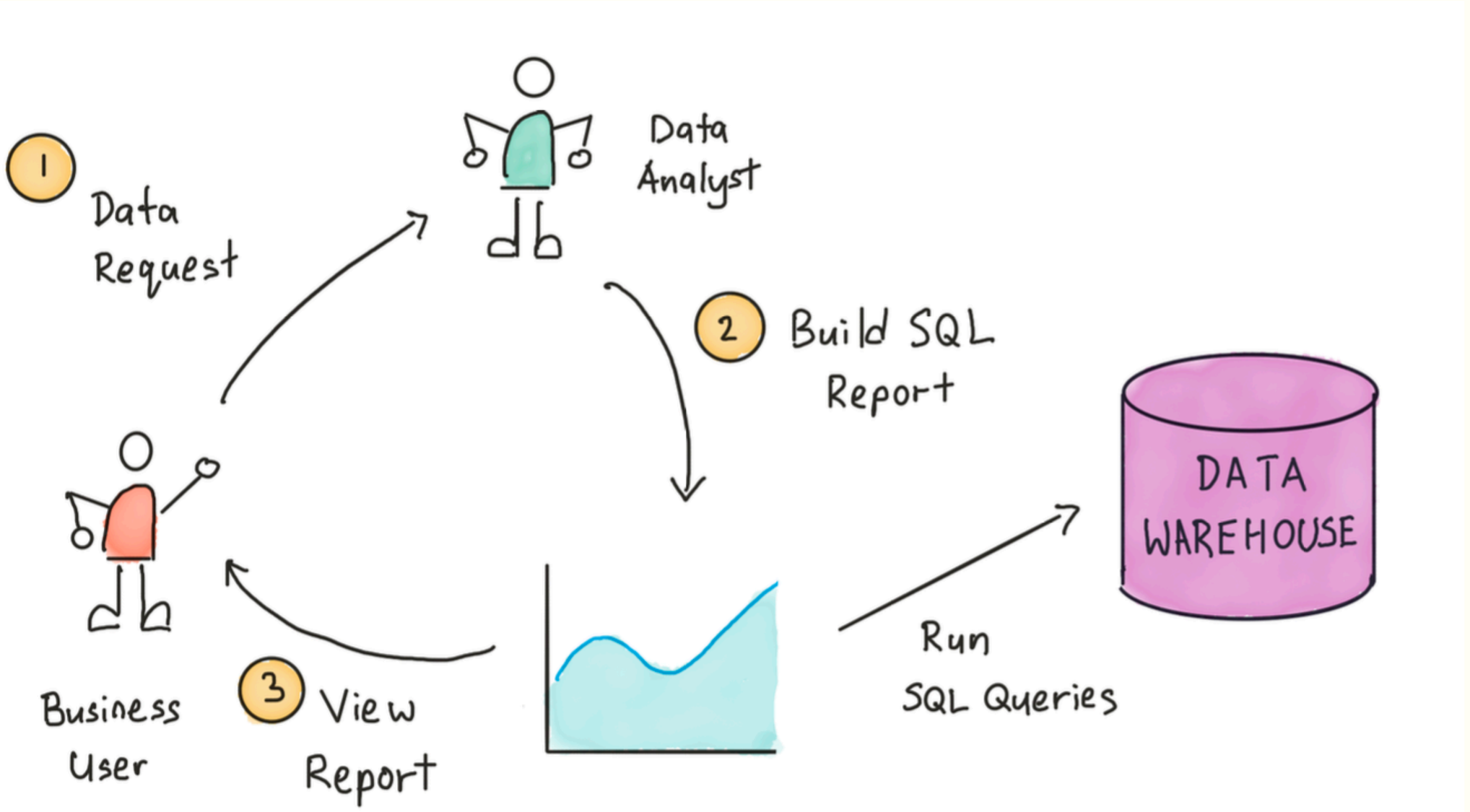


Ok, so we started using a warehouse

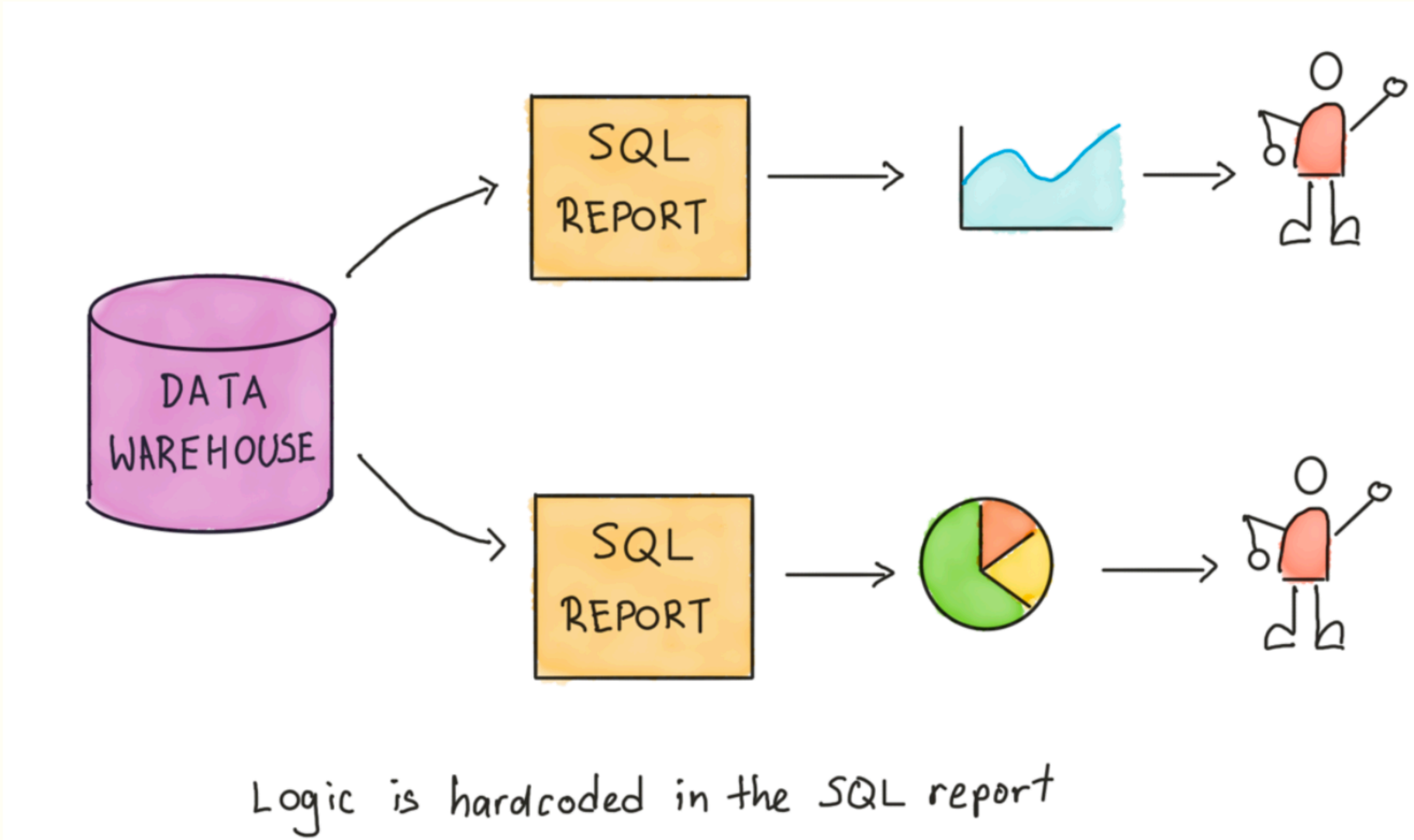


Slightly different input: repeat the whole process!

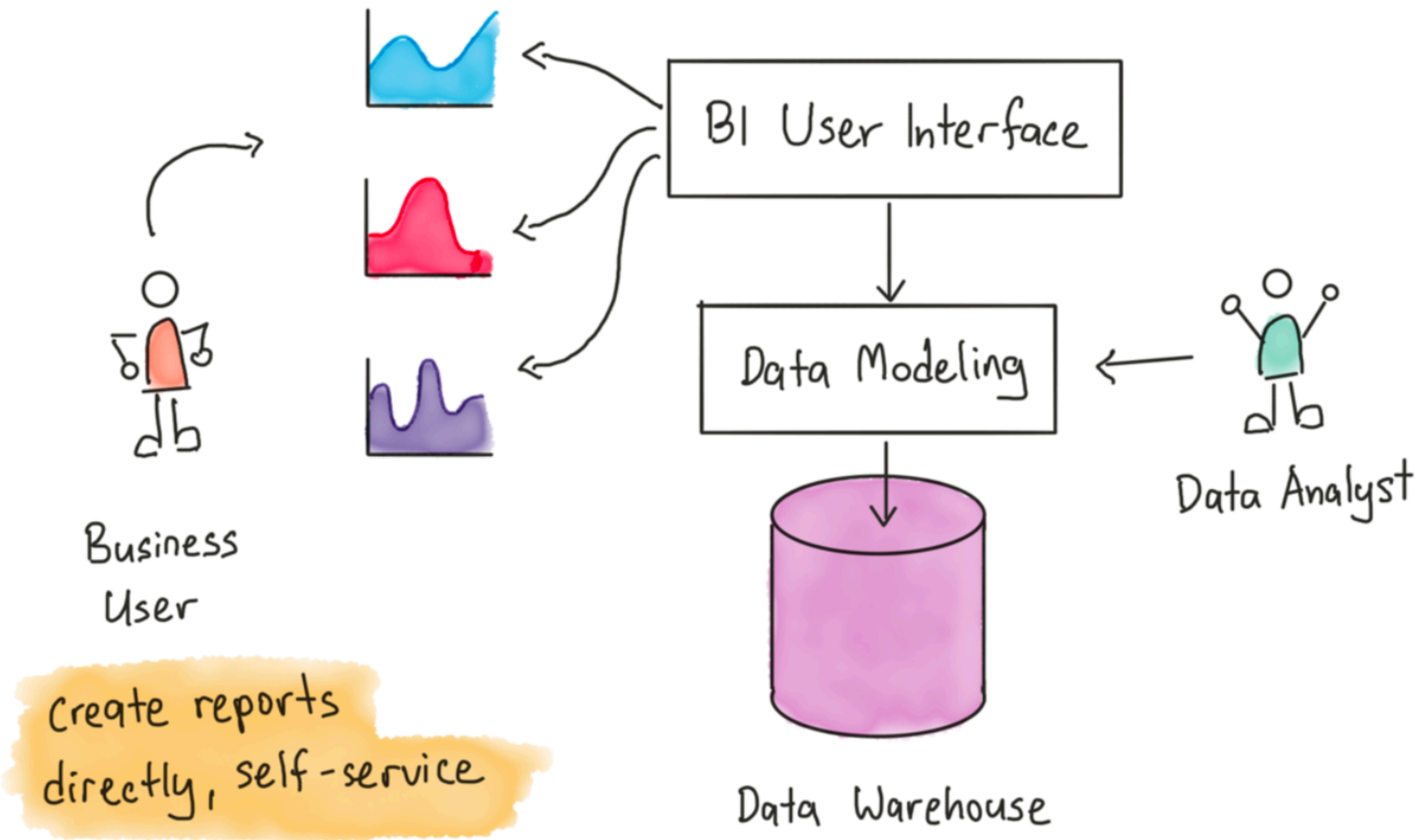
The third job



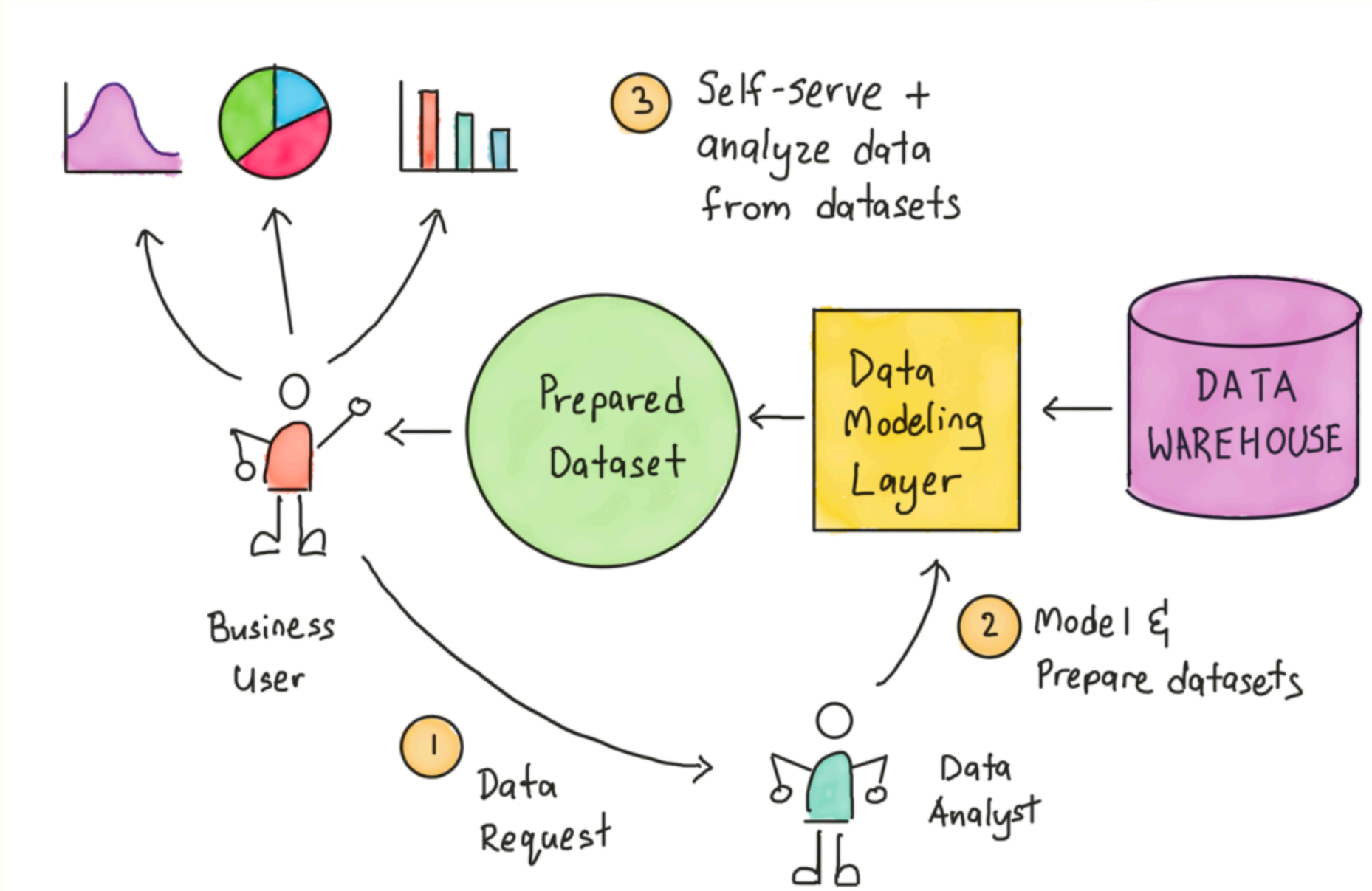
We have not solved the separate metrics problem



What we want: The fourth job

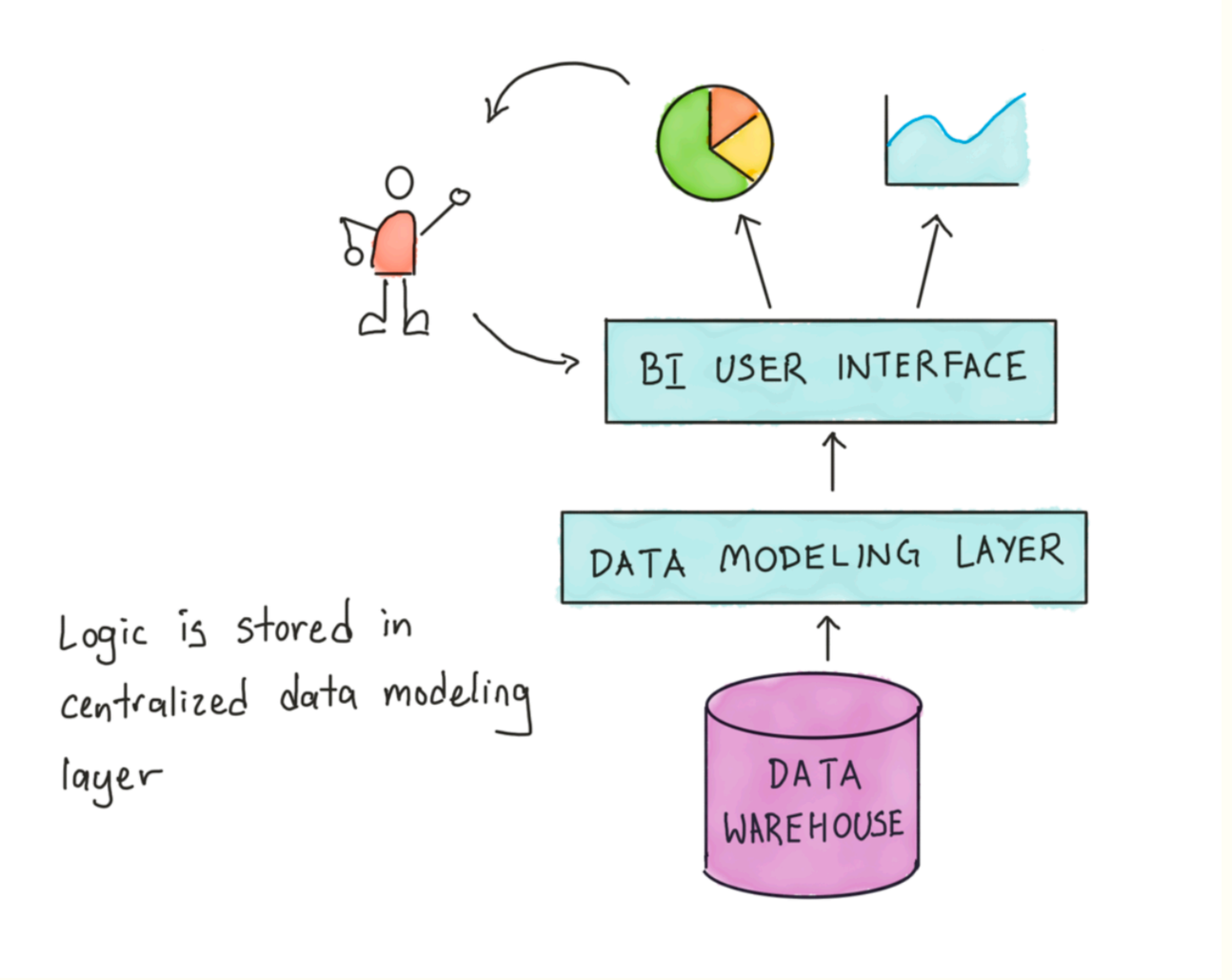


The data modeling layer



The modern world

- We now have lakes and warehouses combined into lakehouses or platforms.
- we add a model layer around these lakes and warehouses to create a registry of datasets and a DAG of transformations that lead to a dataset
- When new data comes in we can refresh these datasets by running the dag again on a sensor



The evolution data engineering

- 1980-2000: Started with Data Warehousing as envisaged by Immon and Kimball using SQL to translate transactional data into a form that could be queried to support business decisions: good data modeling.

- The Internet went mainstream mid-late 1990s creating an explosion in the amount of data collected

- Early 2000s: data exploded, and commodity clusters became commonplace. This pushed data services into a decentralization out of a single relational database. In 2003 Google came up with the GFS and MapReduce papers, and Yahoo developed Hadoop in 2006. The big data engineer was born.

Key: co-locate storage and compute

- Late 2000s to mid 2010s: the big data engineer went mainstream. Data was no longer constrained by the warehouse, and the initial data lakes were formed. The Hadoop ecosystem thrived: Hadoop, Apache Pig, Apache Hive, Dremel, Apache HBase, Apache Storm, Apache Cassandra, Apache Spark, Presto. YARN became the cluster manager.

- Late 2000s: AWS started to go mainstream. Thus the cloud arrived.

- Mid-Late 2010s: The Hadoop ecosystem started to lose some steam, as there was too much on-prem baby-sitting of big-data systems to do. But the explosion of the cloud meant that these clusters could be spun up on the cloud. With Hive, Presto, and SparkSQL came the return of SQL, this time against data lake file formats like parquet. With the cloud also came hosted warehouses such as Redshift, BigQuery, and Snowflake.

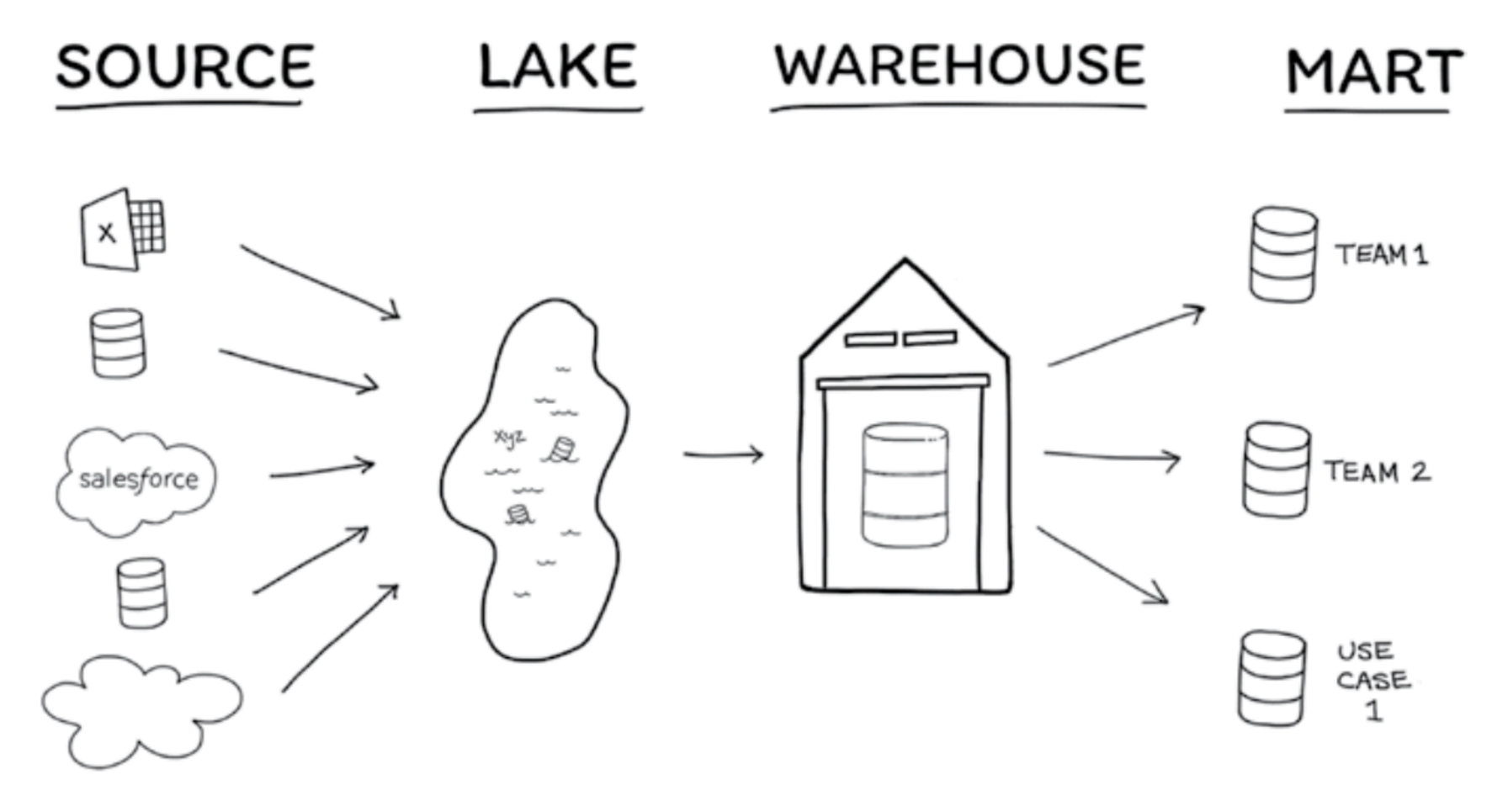
Cloud: separate compute from storage

- Mid-Late 2010s: **Was the use of the cloud to separate compute from storage**. One could autoscale up a hadoop/spark cluster; add more query engines to redshift.

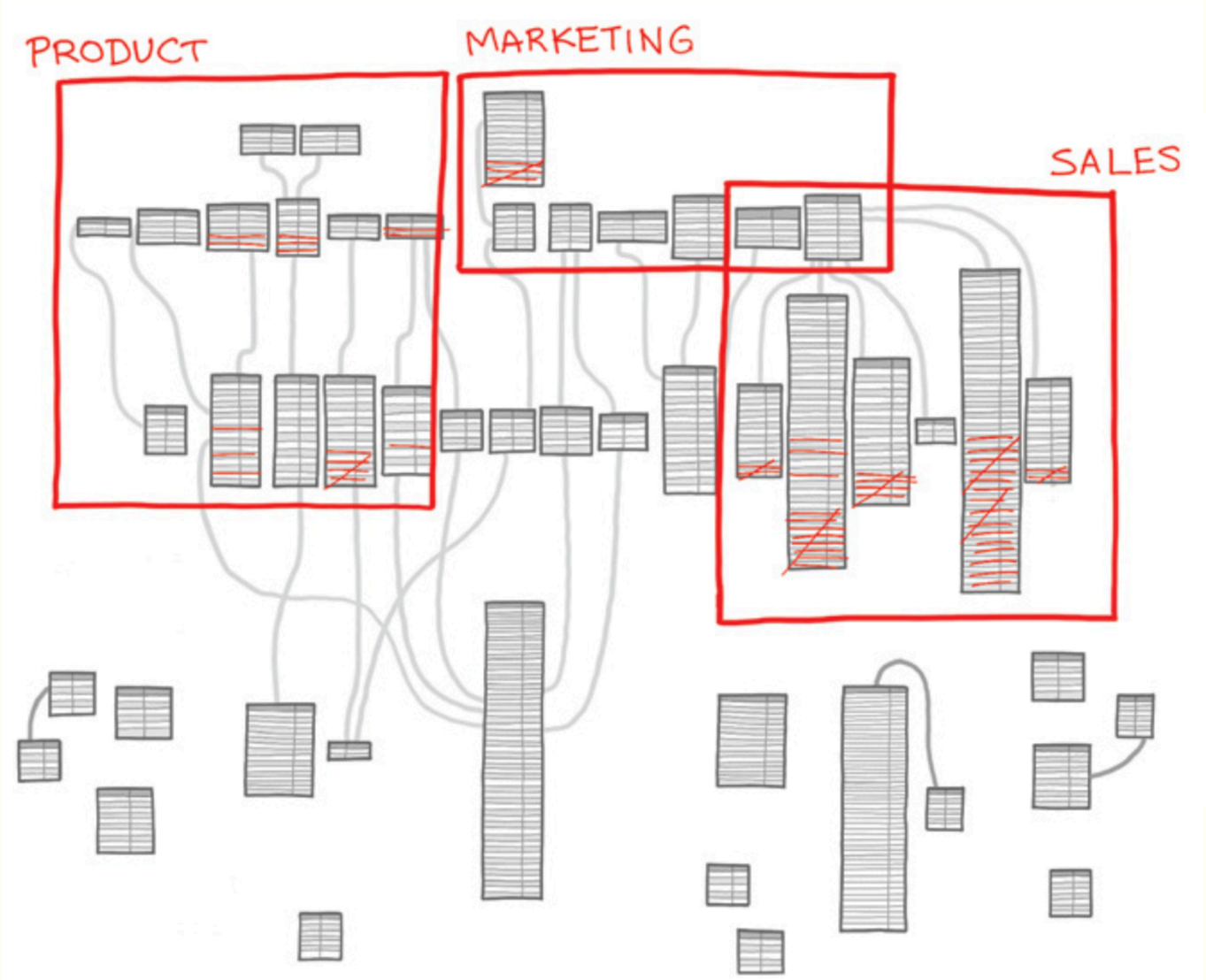
- This lead to the development of the modern data lake. Starting from Hive, engineers recognized the importance of metadata catalogs in organizing different lake files and SQL sources, and thus formats such as Hudi/Iceberg/Delta Lake were born. **Metadata**.

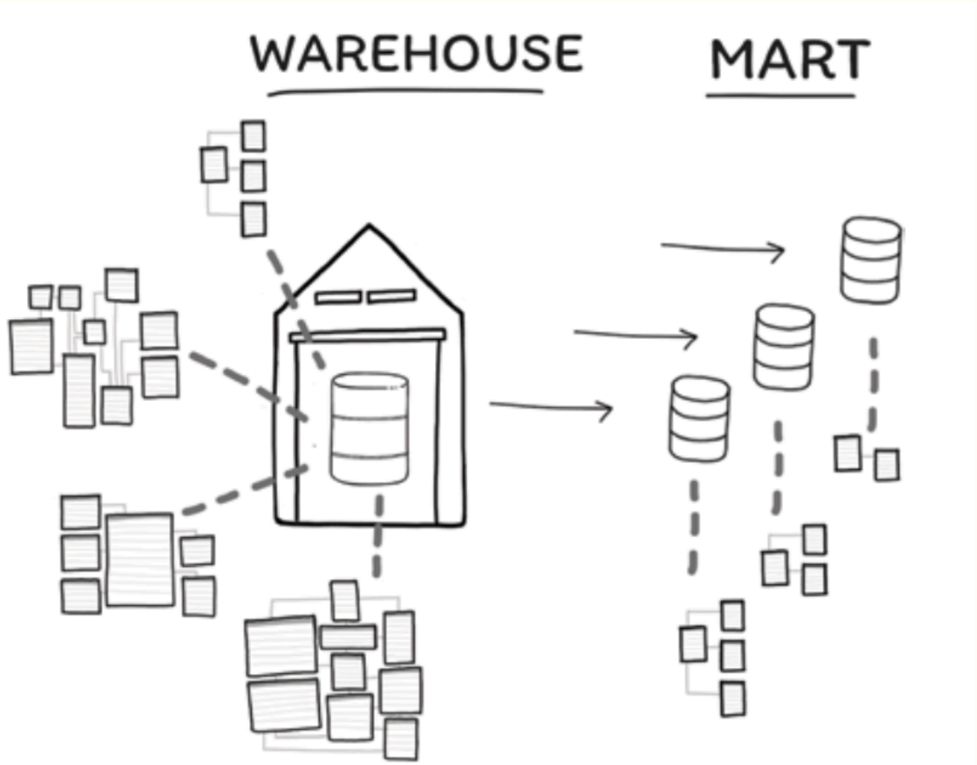
- Late 2010s-2020s: With scaleable options available for data warehouses and data lakes, engineers have once again shifted to data and data pipeline modeling, with an emphasis on reusable models and good software development practices, using tools such as orchestrators and dbt; still using SQL, but now in a composable fashion. It is the age of the data catalog and the data modeling layer. **Repeatable DAGs**.

Components of a modern data platform



Data Marts for specific Verticals



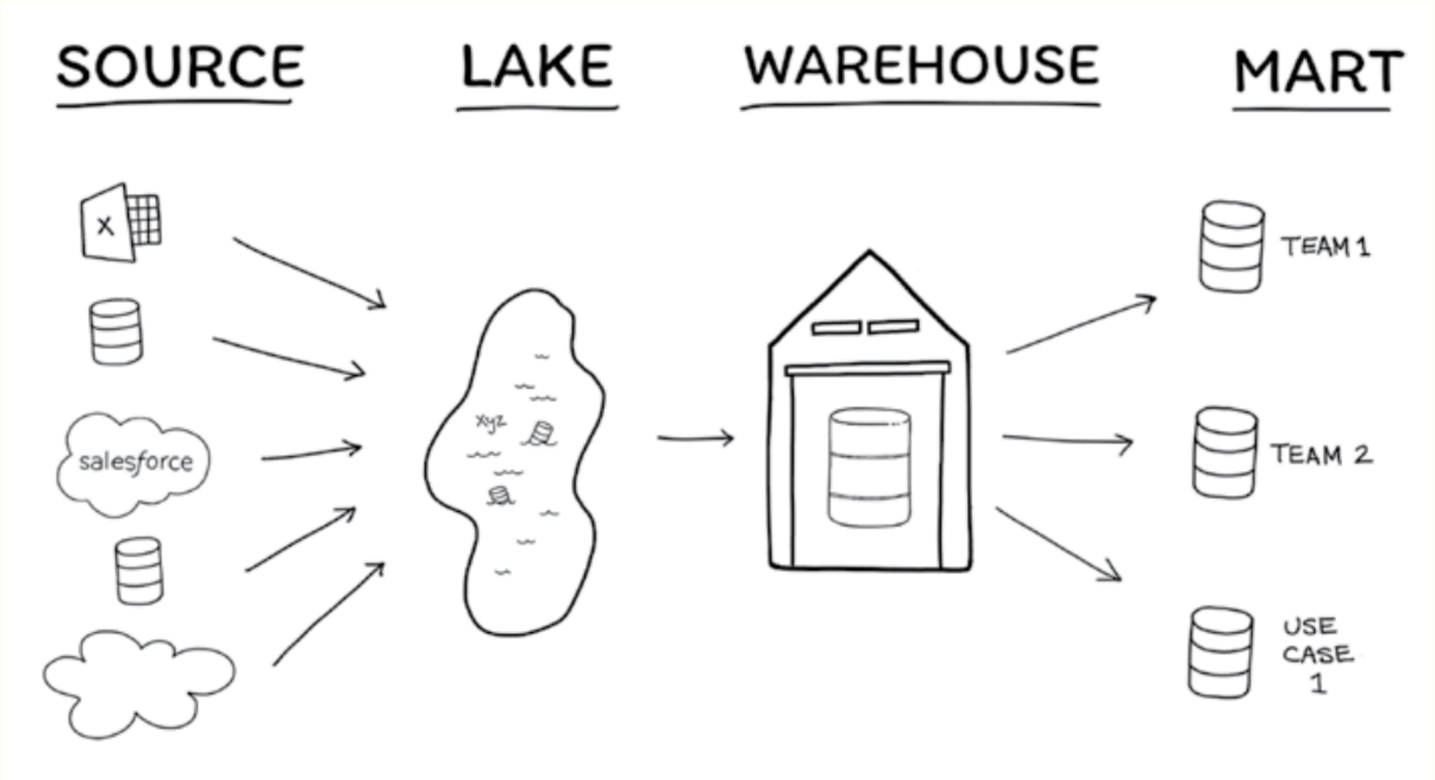


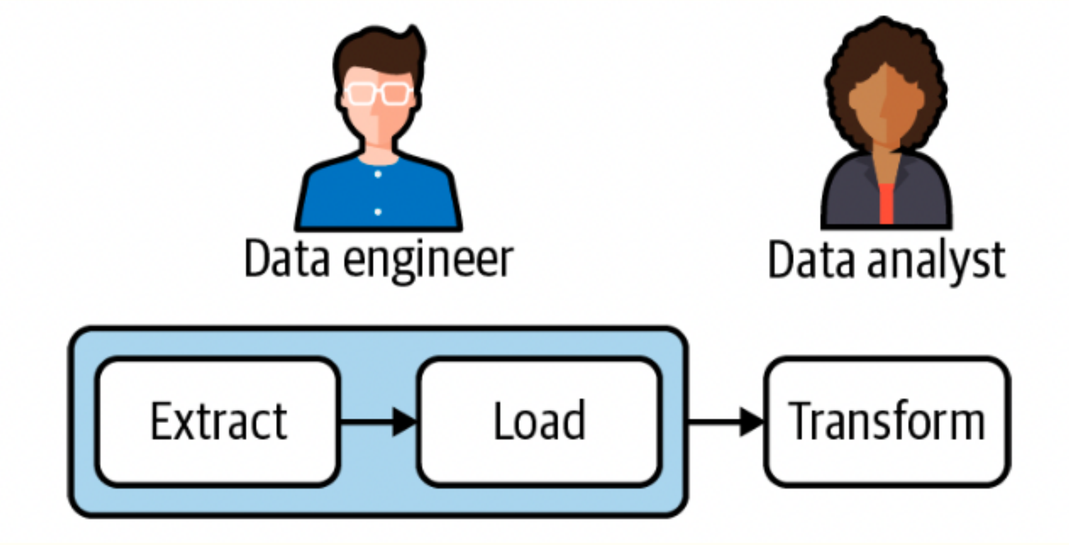
Who are you?

DS DS

DA DA

DE DE DE DE





Univ.AI

Waves of Data Analytics

- from https://community.looker.com/blog-archives-1027/catching-the-third-wave-of-business-intelligence-29412?postid=54151#post54151
- **WAVE1**: 1990s. 1GB RAM=$32K! OLAP Cubes and OLTP RDBMS. Cognos, MicroStrategy and Business Objects. Bottleneck on Data Engineers.
- **WAVE2**: 2000s. Cheaper compute, memory, disk. RDBMS for warehouses. Self service Tableau and Data Marts. Lack of governance and centralization but analysts not blovked on engineers except for ETL to warehouses
- **WAVE3**: 2010s. Columnar Databases. Massively parallel lakes/warehouses: Hadoop and Spark and SQL on hadoop and files. These MPP data warehouses and SQL-on-Hadoop systems are so fast and so cheap that you no longer have to extract your data to analyze it. You can do your analytics right in the database.: ELT
- **WAVE4**?: 2020s. Pays attention to metadata, metastores, the models around the business, and the sharing of these models, but all in the warehouse/lake.

- If you are a data analyst, you should have passing familiarity with all the approaches from all these paradigms.

- Concretely, what it looks like is the following: if you are working as a data analyst in a startup today, you may find yourself operating in a 'first wave' BI environment if you decide to move to a larger, older company tomorrow.

- Conversely, if you are a data analyst in an old- school/2nd wave data department, you may be taken by surprise if you leave and find yourself in an ELT-first paradigm at a younger company or a FAANG.

Your responsibilities

- Generation (sometimes)
- Storage
- Ingestion
- Transformation
- Serving

- Security (sometimes)Data Management
- Data Management
- DataOps
- Data architecture
- Orchestration
- Software Engineering

Your skills

- SQL/dbt
- Python
- Java/Scala
- bash
- linux/docker/systems
- Pandas/Dask/Spark
- Git/Github/Actions
- Databases/Formats and Modeling
- BI/Dashboards/Analytics/ml
- Architecture

Can you think it through?

This is your strongest skill