Data Lakes & ELT

ELENA BARALIS & DANIELE APILETTI

POLITECNICO DI TORINO

Data Base and Data Mining Group of Politecnico di Torino

ETL - Extract, Transform, Load

- ETL involves extracting data from various sources, transforming it into a suitable format, and then loading it into a **data warehouse**
- ETL is used in **data warehouses** where structured data from multiple sources is transformed to fit into predefined schemas.
  - This ensures data consistency and cleanliness before it is loaded into the warehouse
- The transformation stage happens **before** data is loaded into the destination.
  - This typically involves cleaning, filtering, aggregating, and enriching the data

ETL challenges

- The transformation process can be **resource-intensive** and **slow**, especially with large and possibly streaming datasets.
- ETL can struggle with **unstructured** or semi-structured data, such as logs or social media feeds.

ELT - Extract, Load, Transform

- In ELT, the data is first **extracted** from the source and then **loaded** directly into the **data lake** or warehouse in its raw form
  - The transformation happens **later** within the storage system itself
- ELT is commonly associated with **data lakes**, where raw, unstructured, and semi-structured data is stored and transformed on demand, rather than upfront
- Unlike ETL, ELT postpones transformation to after the data is loaded. This allows transformation to occur only when it's needed for specific use cases (e.g., analysis, reporting)

ELT - Extract, Load, Transform

- **Scalability**: ELT handles large, unstructured datasets and scales well with big data
  - The raw data can be stored and processed later without initial transformation bottlenecks
- **Flexibility**: The raw data remains available for different types of analysis without predefining a rigid schema
- **Performance**: Since transformation is done on the storage layer (such as using **cloud data warehouses** like Snowflake or Redshift, or with distributed systems like Hadoop), ELT can leverage the computing power of modern platforms for faster processing
  - ELT enables the use of distributed computing frameworks like **Apache Spark**, **Hadoop**, or **Databricks**, which can perform complex transformations directly on the stored raw data, making it ideal for large-scale, unstructured data analysis

ELT challenges

- Potential for **data sprawl** or data management issues because the data is loaded before being cleaned or transformed
- **Data quality** can suffer unless managed carefully, as raw data can be incomplete or inconsistent

Data lake

- Data repository for
  - Original data in ***raw*** format
  - Transformed data used for various types of reporting
- Data formats
  - Structured data (e.g., relational data)
  - Semi-structured data (e.g., CSV, JSON, XML)
  - Unstructured data (e.g., text documents, emails)
  - Binary data (e.g., images, audio files)
- Query more similar to a google search (+ data wrangling)

Why data lakes?

- Often not all questions data can answer are known a-priori
  - hard to store data in some «optimal» form
- An attempt to break down information silos
  - Information not adequately shared among data systems
- Based on exploiting massive, cheap data storage
  - Modern cloud platforms (e.g., AWS S3 + Athena, Azure Data Lake, or Google Cloud BigQuery) that back Data Lakes have powerful processing engines. This allows the transformation in ELT to be done efficiently on-demand when querying the data, without the need for upfront processing.

Data lakes characteristics

- Data lakes store all data
  - DW design requires deciding what data to include (and to not include) in the warehouse
  - Data lakes include also data that might be used “someday”
- Data lakes manage all data types
- Data lakes provide service to all users
  - Users process a variety of different types of data and answer new questions
- Data lakes adapt easily to changes
  - All data is stored in its raw form and is always accessible
  - Users are empowered to explore data in novel ways
- Data lakes provide faster insight
  - … but early access to the data comes at a price

Data warehouse

- Relational data coming from transactional systems, operational databases, and line of business applications
- Schema designed prior to DW implementation (schema-on-write)
- High cost storage
- Data quality: highly curated data that serves as the central version of the truth
- Users are business analysts
- Analytics: BI and visualization, batch reporting

Data lake

- Data is both non-relational and relational, coming from IoT devices, web sites, mobile apps, social media, and corporate applications
- Schema is written at the time of analysis (schema-on-read)
- Low-cost storage
- Data quality: Any data that may or may not be curated (ie. raw data)
- Users are data scientists, data developers
  - business analysts, if using curated data
- Analytics: full-text search, machine learning, predictive analytics, data discovery and profiling

Pros of data lakes

- Ability to harness more data, from more sources, in less time
- Data structures and business requirements are defined only when needed
- Empowering users to collaborate and analyze data in different ways
  - self service analytics
- Integration happens outside the storage environment
- Minimal involvement of IT
  - Wrangling with data is a self-service function
- Sandboxes for self-service analytics
  - Need well defined problems

Cons of data lakes

- Raw data is stored with no oversight of the contents
  - Storing data does not, on its own, provide business value
  - Need data governance, semantic consistency, mechanism to catalog data
- Consistency and data quality are uncertain
  - Data brought into a data lake is co-located not integrated
- Business users don't have time/willingness to learn
  - How can they wrangle with raw data?
- Rogue queries can bring down big clusters

The central question is whether collecting and storing data without a pre-defined business purpose is a good idea

Example 1 – DW + ETL

- Use Case: A company wants to integrate **transactional data from multiple sources like CRM, ERP, and finance systems** into a centralized system for reporting and BI. The data needs to be cleansed, structured, and made consistent across all systems
- ETL Process: The data is extracted from each source, transformed (cleaned, aggregated, and joined into a unified format), and then loaded into the data warehouse
- Result: Data is available for **high-quality**, **predefined reports** and dashboards, optimized for speed and **consistency** in relational queries.

Example 2 – Data Lake + ELT

- Use Case: A company collects **a vast array of unstructured data, such as social media feeds, IoT sensor data, and website logs**. They want to store this data for potential analysis without knowing how they will use all of it right away

- ELT Process: The raw data is loaded into a **Data Lake** (e.g., Amazon S3 or Hadoop). Transformation happens later when analysts need specific insights, using tools like Spark or Athena to process the raw data

- Result: The raw data is available for a **wide range of use cases**, including **machine learning**, exploratory data analysis, and ad-hoc queries, without having to predefine the data schema

From data lakes to data swamps

Massive repositories of data that are completely inaccessible to end users

- data collected without any clear way to get value from it
- risk to be abandoned (budget cut)

To avoid drowning in your data lake

- Collect less data, at least in the beginning…