ETL and Data Pipelines with Shell, Airflow and Kafka Notes

Chuks Okoli

04 January, 2024

Week 1

ETL using Shell Scripts

ETL and ELT Processes
ETL stands for Extract, Transform, and Load. ETL is an automated data pipeline engineering methodology, whereby data is acquired and prepared for subsequent use in an analytics environment, such as a data warehouse or data mart. ETL refers to the process of curating data from multiple sources, conforming it to a unified data format or structure, and then loading the transformed data into its new environment.

- E = Extraction: Extraction process obtains or reads the data from one or more sources
  - Some common methods include: Web scraping, where data is extracted from web pages using applications such as Python or R to parse the underlying HTML code
  - Using APIs to programmatically connect to data and query it
- T = Transformation: Transformation process wrangles the data into a format that is suitable for its destination and its intended use. Transformation is done in an intermediate working environment called a "staging area" and can include any of the following kinds of processes:
  - Cleaning: fixing errors or missing values
  - Filtering: selecting only what is needed
  - Joining disparate data sources: merging related data
  - Normalizing: converting data to common units
  - Data Structuring: converting one data format to another, such as JSON, XML, or CSV to database tables
  - Feature engineering: such as creating KPIs for dashboards or machine learning
  - Anonymizing and Encrypting: ensuring privacy and security
  - Sorting: ordering the data to improve search performance
  - Aggregating: summarizing granular data
  - Formatting and data typing: making the data compatible with its destination.
- L = Load: Loading takes the transformed data and loads it into its new environment (.e.g., database, data warehouses or data mart), ready for visualization, exploration, further transformation, and modelling
  - Data loading techniques
    - Full
    - Incremental
    - Scheduled
    - On-demand
    - Batch and stream
    - Push and pull
    - Parallel and serial

The ETL process is shown in **Fig. 1**.

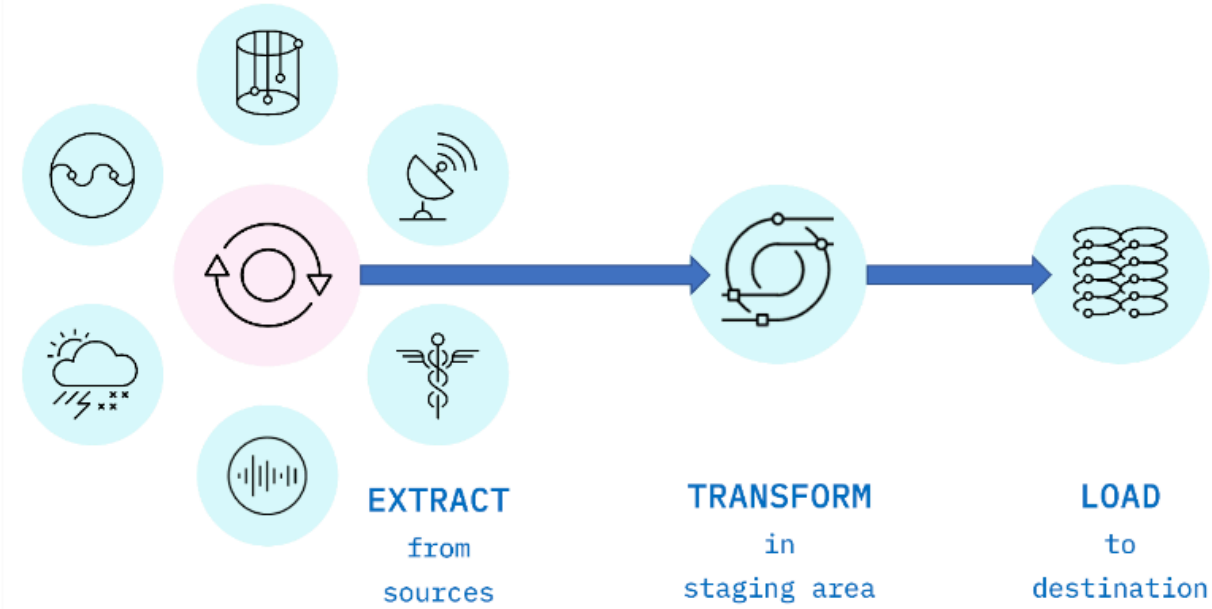

Figure 1: The ETL process

Summary of ETL and ELT Processes

- ETL stands for Extract, Transform, and Load
- Loading means writing the data to its destination environment
- Cloud platforms are enabling ELT to become an emerging trend
- The key differences between ETL and ELT include the place of transformation, flexibility, Big Data support, and time-to-insight
- There is an increasing demand for access to raw data that drives the evolution from ETL, which is still used, to ELT, which enables ad-hoc, self-serve analytics
- Data extraction often involves advanced technology including database querying, web scraping, and APIs
- Data transformation, such as typing, structuring, normalizing, aggregating, and cleaning, is about formatting data to suit the application
- Information can be lost in transformation processes through filtering and aggregation
- Data loading techniques include scheduled, on-demand, and incremental
- Data can be loaded in batches or streamed continuously

Week 2

ETL Workflows as Data Pipelines

Traditionally, the overall accuracy of the ETL workflow has been a more important requirement than speed, although efficiency is usually an important factor in minimizing resource costs. To boost efficiency, data is fed through a *data pipeline* in smaller packets (see **Fig. 2**). While one packet is being extracted, an earlier packet is being transformed, and another is being loaded. In this way, data can keep moving through the workflow without interruption. Any remaining bottlenecks within the pipeline can often be handled by parallelizing slower tasks.

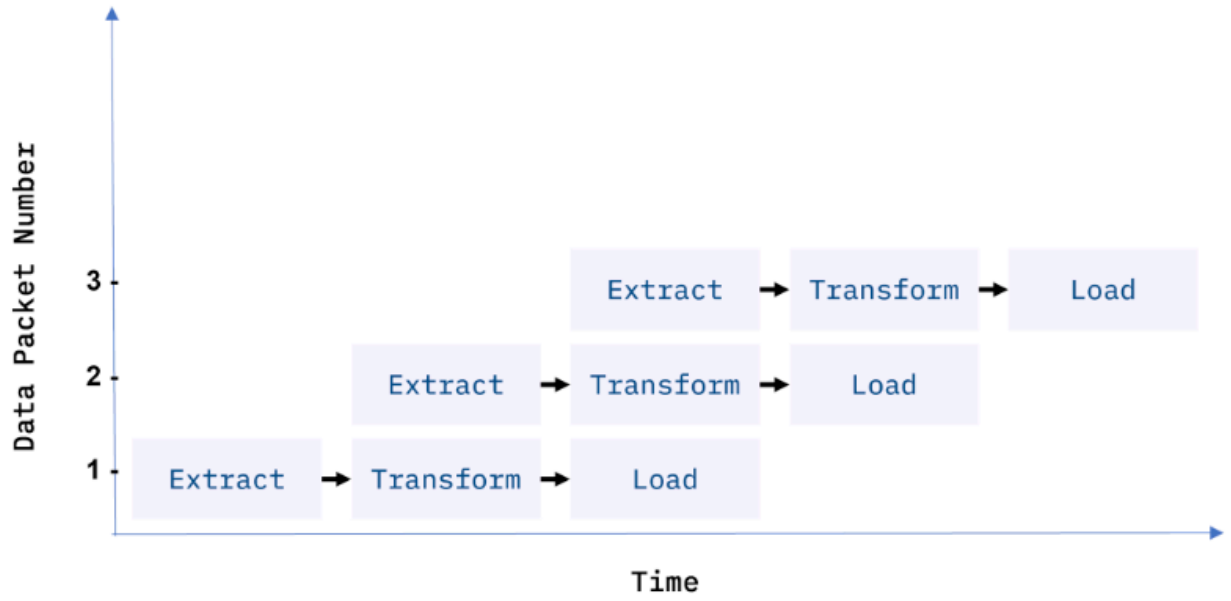

Figure 2: ETL Workflows as Data Pipelines

With conventional ETL pipelines, data is processed in *batches*, usually on a repeating schedule that ranges from hours to days apart. For example, records accumulating in an Online Transaction Processing System (OLTP) can be moved as a daily batch process to one or more Online Analytics Processing (OLAP) systems where subsequent analysis of large volumes of historical data is carried out.

Staging Areas

ETL pipelines are frequently used to integrate data from disparate and usually siloed systems within the enterprise. These systems can be from different vendors, locations, and divisions of the company, which can add significant operational complexity. As an example, (see **Fig. 3**) a cost accounting OLAP system might retrieve data from distinct OLTP systems utilized by the separate payroll, sales, and purchasing departments.

ETL pipelines are frequently used to integrate data from disparate and usually *siloed* systems within the enterprise. These systems can be from different vendors, locations, and divisions of the company, which can add significant operational complexity. As an example, (see **Fig. 3**) a cost accounting OLAP system might retrieve data from distinct OLTP systems utilized by the separate payroll, sales, and purchasing departments.

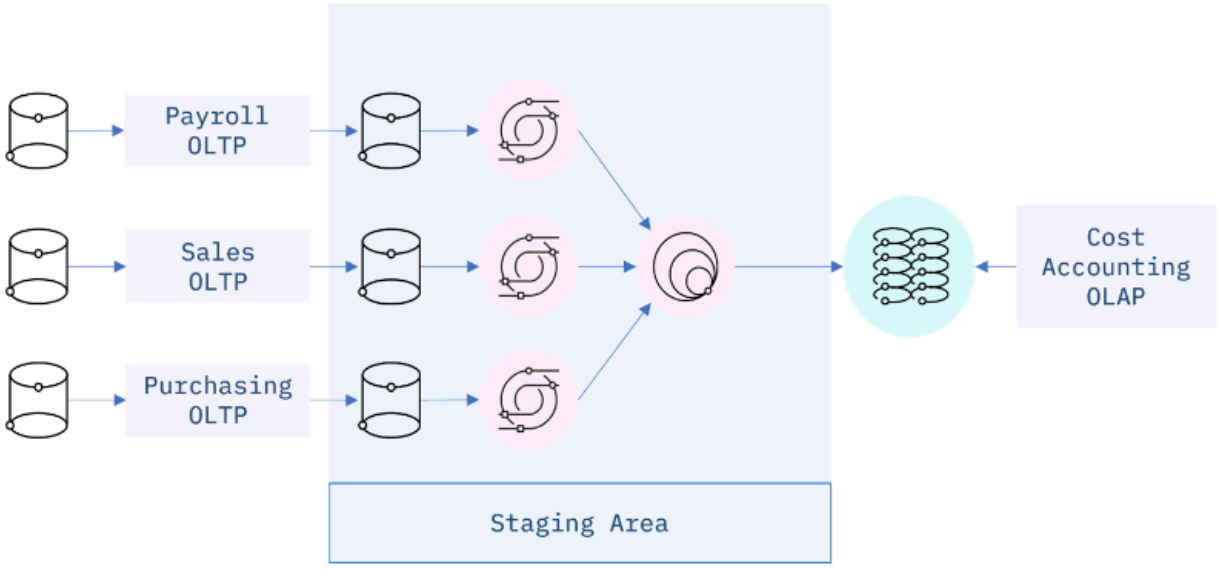

Figure 3: An ETL data integration pipeline concept for a Cost Accounting OLAP

ETL Workflows as DAGs

ETL workflows can involve considerable complexity. By breaking down the details of the workflow into individual tasks and dependencies between those tasks, one can gain better control over that complexity. Workflow orchestration tools such as Apache Airflow do just that.

Airflow represents your workflow as a directed acyclic graph (DAG). A simple example of an Airflow DAG is illustrated in **Fig. 4**. Airflow tasks can be expressed using predefined templates, called operators. Popular operators include Bash operators, for running Bash code, and Python operators for running Python code, which makes them extremely versatile for deploying ETL pipelines and many other kinds of workflows into production.

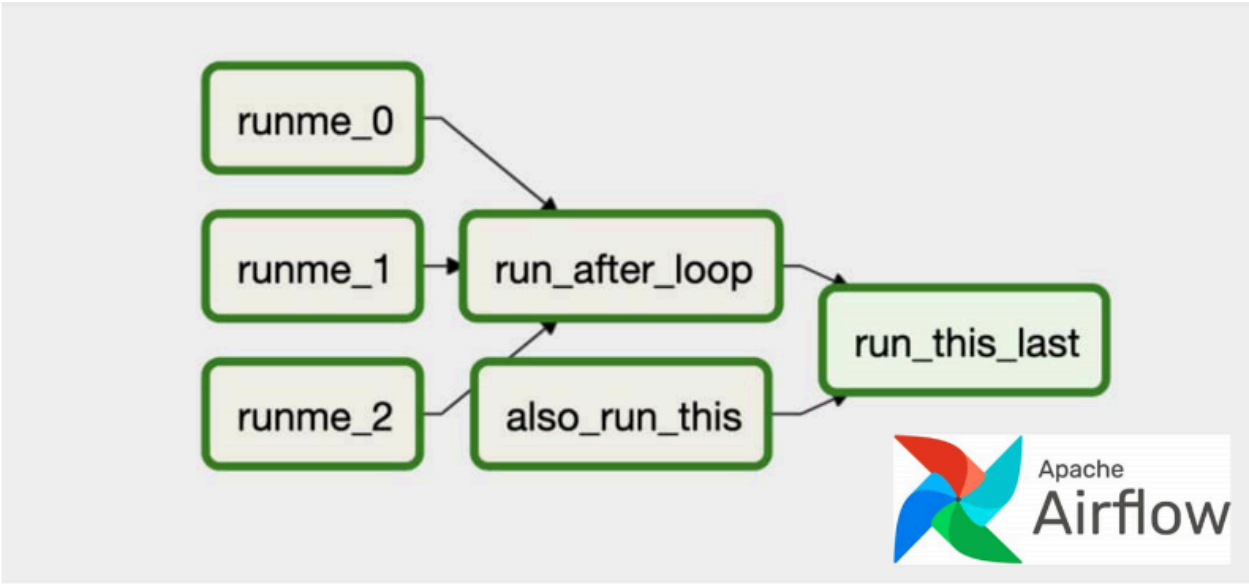

Figure 4: An Apache Airflow DAG representing a workflow

An Introduction to Data Pipelines

Summary of .....

- Data pipelines move data from one place, or form, to another
- Data flows through pipelines as a series of data packets
- Latency and throughput are key design considerations for data pipelines
- Data pipeline processes include scheduling or triggering, monitoring, maintenance, and optimization
- Parallelization and I/O buffers can help mitigate bottlenecks
- Batch pipelines extract and operate on batches of data
- Batch processing applies when accuracy is critical, or the most recent data isn't required
- Streaming data pipelines ingest data packets one-by-one in rapid succession
- Streaming pipelines apply when the most current data is needed
- Examples of streaming data pipelines use cases, such as social media feeds, fraud detection, and real-time product pricing
- Modern data pipeline technologies include schema and transformation support, drag-and-drop GUIs, and security features
- Stream-processing technologies include Apache Kafka, IBM Streams, and SQLStream

Week 3

Building Data Pipelines using Apache Airflow

Week 4

Building Streaming Pipelines using Apache Kafka

ETL using Shell Scripts

ETL T

-
-
-
-