# ETL (Extract, Transform, Load)

ETL is a process that extracts, transforms, and loads data from multiple sources to a data warehouse or other unified data repository.

What is ETL?

ETL, which stands for extract, transform and load, is a data integration process that combines data from multiple data sources into a single, consistent data store that is loaded into a data warehouse or other target system.

As the databases grew in popularity in the 1970s, ETL was introduced as a process for integrating and loading data for computation and analysis, eventually becoming the primary method to process data for data warehousing projects.

ETL provides the foundation for data analytics and machine learning workstreams. Through a series of business rules, ETL cleanses and organizes data in a way which addresses specific business intelligence needs, like monthly reporting, but it can also tackle more advanced analytics, which can improve back-end processes or end user experiences. ETL is often used by an organization to:

- Extract data from legacy systems
- Cleanse the data to improve data quality and establish consistency
- Load data into a target database

Featured products

DataStage

InfoSphere Information Server Enterprise Edition

ETL vs ELT

The most obvious difference between ETL and ELT is the difference in order of operations. ELT copies or exports the data from the source locations, but instead of loading it to a staging area for transformation, it loads the raw data directly to the target data store to be transformed as needed.

While both processes leverage a variety of data repositories, such as databases, data warehouses, and data lakes, each process has its advantages and disadvantages. ELT is particularly useful for high-volume, unstructured datasets as loading can occur directly from the source. ELT can be more ideal for big data management since it doesn't need much upfront planning for data extraction and storage. The ETL process, on the other hand, requires more definition at the onset. Specific data points need to be identified for extraction along with any potential "keys" to integrate across disparate source systems. Even after that work is completed, the business rules for data transformations need to be constructed. This work can usually have dependencies on the data requirements for a given type of data analysis, which will determine the level of summarization that the data needs to have. While ELT has become increasingly more popular with the adoption of cloud databases, it has its own disadvantages for being the newer process, meaning that best practices are still being established.

**How ETL works**

The easiest way to understand how ETL works is to understand what happens in each step of the process.

## Extract

During data extraction, raw data is copied or exported from source locations to a staging area. Data management teams can extract data from a variety of data sources, which can be structured or unstructured. Those sources include but are not limited to:

- SQL or NoSQL servers
- CRM and ERP systems
- Flat files
- Email
- Web pages

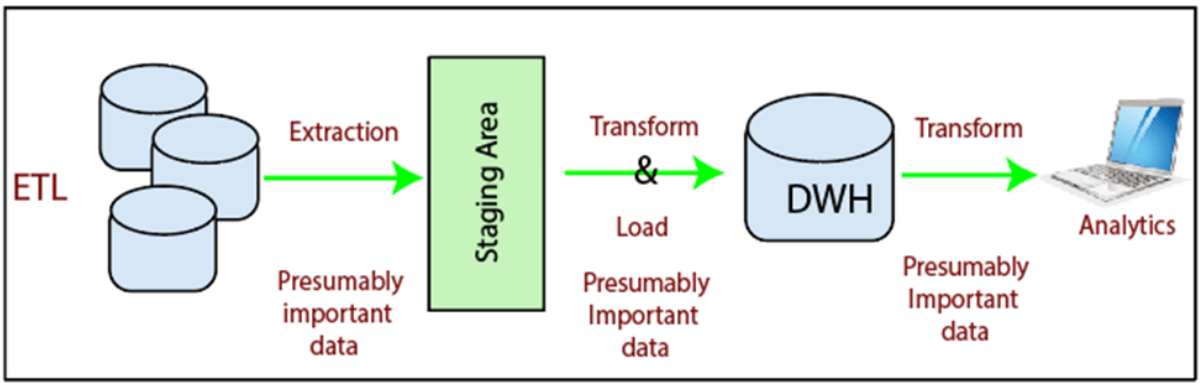
## Transform

In the staging area, the raw data undergoes data processing. Here, the data is transformed and consolidated for its intended analytical use case. This phase can involve the following tasks:

- Filtering, cleansing, de-duplicating, validating, and authenticating the data.
- Performing calculations, translations, or summarizations based on the raw data. This can include changing row and column headers for consistency, converting currencies or other units of measurement, editing text strings, and more.
- Conducting audits to ensure data quality and compliance
- Removing, encrypting, or protecting data governed by industry or governmental regulators
- Formatting the data into tables or joined tables to match the schema of the target data warehouse.

## Load

In this last step, the transformed data is moved from the staging area into a target data warehouse. Typically, this involves an initial loading of all data, followed by periodic loading of incremental data changes and, less often, full refreshes to erase and replace data in the warehouse. For most organizations that use ETL, the process is automated, well-defined, continuous and batch-driven. Typically, ETL takes place during off-hours when traffic on the source systems and the data warehouse is at its lowest.




## WHAT IS SPARK ETL?

Business demands across multiple industries has impacted the ETL landscape for data engineering, data science and machine learning. The ETL (Extract, Transform, Load) process can be lengthy and laborious. To generate usable data quickly, ETL pipelines must be constantly continuous data, churning, and loading data.

Apache Spark provides the framework to up the ETL game. Data pipelines enable organizations to make faster data-driven decisions through automation. They are an integral piece of an effective ETL process because they allow for effective and accurate aggregating of data from multiple sources.

Spark was known for innately supporting multiple data sources and programming languages. Whether relational data or semi-structured data, such as JSON, Spark ETL delivers clean data.

Spark data pipelines have been designed to handle enormous amounts of data.

#### SNOWFLAKE AND SPARK ETL

Snowflake's Snowpark Delivers the Benefits of Spark ETL with None of the Complexities

Snowflake's Snowpark framework brings integrated, DataFrame-style programming to the languages developers like to use and performs large-scale data processing, all executed inside of Snowflake for ETL jobs. Here are just a few of the things that organizations are accomplishing using Snowpark.

- **Improve collaboration:** Bring all teams to collaborate on the same data in a single platform that natively supports everyone's programming language and constructs of choice, including Spark DataFrames.
- **Accelerate time to market:** Enable technical talent to increase the pace of innovation on top of existing data investments with native support for cutting-edge open-source software and APIs.

- **Lower total cost of ownership:** Streamline architecture to reduce infrastructure and operational costs from unnecessary data pipelines and Spark-based environments.
- **Reduce security risks:** Exert full control over libraries being used. Provide teams with a single and governed source of truth to access and process data to simplify data security and compliance risk management across projects.

  Thanks to Snowflake's Snowpark, organizations can achieve lightning-fast data processing for ETL jobs and data pipelines via their developer's favorite programming languages and coding constructs. At the same time, they can enjoy all the advantages that the Snowflake Data Cloud offers.

  In addition, Snowflake's platform can also connect with Spark. The Snowflake Connector for Spark keeps Snowflake open to connect to some complex Spark workloads.

  To learn more about Snowpark, watch an on-demand session on What's New with Snowpark.

### Scala and Apache Spark in Tandem as a Next-Generation ETL Framework

Scala and Apache Spark might seem an unlikely medium for implementing an ETL process, but there are reasons for considering it as an alternative. After all, many Big Data solutions are ideally suited to the preparation of data for input into a relational database, and Scala is a well thought-out and expressive language. Krzysztof Stanaszek describes some of the advantages and disadvantages of a scala-based approach to implementing and testing an ETL solution.

## ETL redefined

ETL is a well-known acronym, standing for Extract, Transform and Load. It is identified particularly with data warehouse workloads and business intelligence applications. Nowadays, however, any regular process of data movement in a data-driven organization could be considered to be a type of ETL, and there are many alternative ways of managing such a process. There are, for example, several data integration tools, ranging from open source all the way to enterprise-level commercial solutions. Many of them, if not all, are GUI-based tools. The main advantage of the graphical approach is that you can rapidly provision your ETL packages, simply by using drag and drop capabilities; and have a nice visual overview. There are a whole range of scripting approaches as well, even using workflow techniques to ensure that ETL processes can be persistent and resilient. The advent of Big Data has given us another alternative approach using Scala powered by the built-in parallel processing capabilities of Apache Spark. In this article, we will look closer at this alternative, and try assess whether it is worth the hype.

## GUI based ETL

Before we jump into the details of coding in Scala, let’s check out those advantages that GUI-based ETL tools give us out of the box. For this purpose, we’ll build a simple integration package that will simply count the words from a text file. The Microsoft SQL Server Integration Services (MS SSIS) example could look like this.

# Scala advantages

By choosing Spark as a processing framework that is internally written in Scala, you will be limited in programming languages to Scala, Python, Java, C# and R. However, you become enabled to write unit and integration tests in a framework of your choice, set up a team-based development project with less painful code merges, leverage source control, build, deployment and continuous integration features. The framework and language will protect you from many errors and bugs that are bread and butter of every IT project. Besides all of the advantages of modern programming language, Scala gives you extra benefits such as:

- conciseness and less code than in a language such as Java or C#,
- immutable data structures allowing for parallel, lock-free data processing,
- full-featured object-oriented paradigms, so you are not limited to functional programming,
- performance that is better than interpreted languages such as Python,
- good compatibility with the MapReduce processing model,
- a plethora of well-designed libraries for scientific computing.