**TASK 4 DATA ENGINEERING BASICS 16-03-23**

**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

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 work streams. 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.

**HOW ETL WORKS**

A data lake house is a new, open data management architecture that combines the flexibility, cost-efficiency, and scale of data lakes with the data management and ACID transactions of data warehouses, enabling business intelligence (BI) and machine learning (ML) on all data.

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

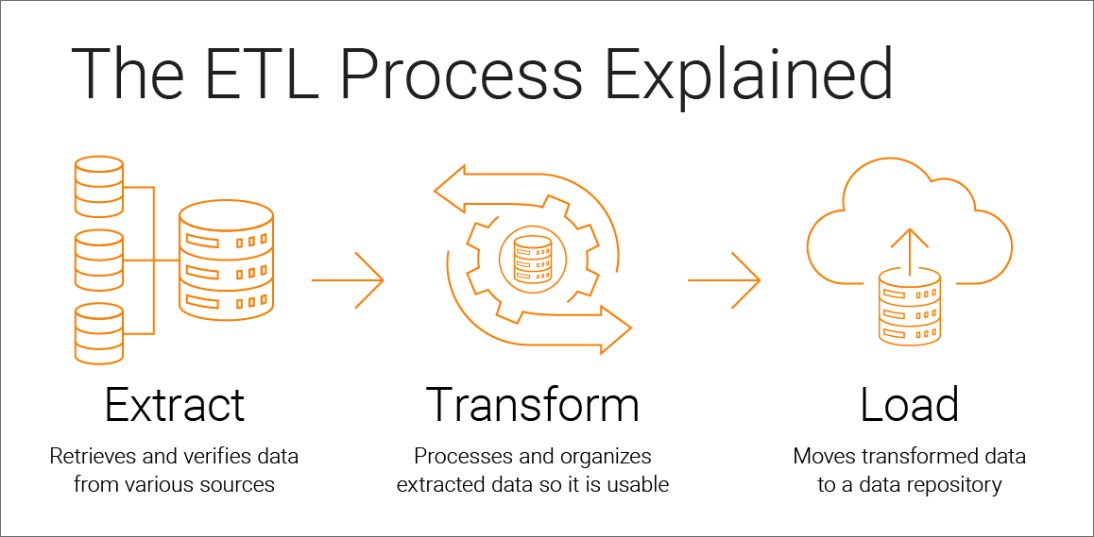
**TRANSFORM**

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**

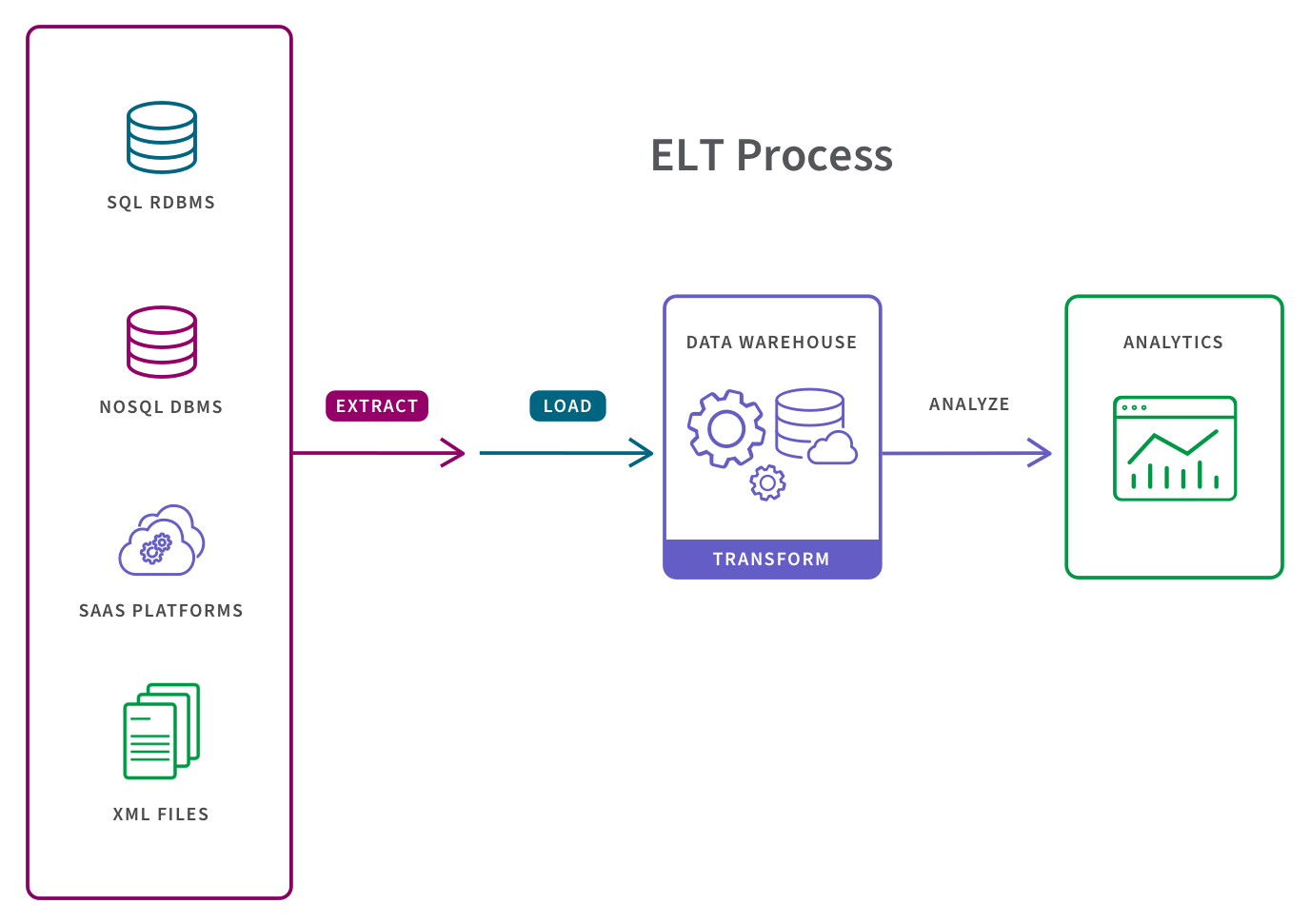
In this last step, the transformed data is moved from the staging area into a target data warehouse. Typically, this involves an initial loading of all data, followed by periodic loading of incremental data changes and, less often, full refreshes to erase and replace data in the warehouse. For most organizations 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.



**ELT (EXTRACT, LOAD, TRANSFORM)**

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

ELT, which stands for “Extract, Load, Transform,” is another type of data integration process, similar to its counterpart ETL, “Extract, Transform, and Load”. This process moves raw data from a source system to a destination resource, such as a data warehouse. While similar to ETL, ELT is a fundamentally different approach to data pre-processing which has only more recently gained adoption with the transition to cloud environments.



**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.

**other data integration methods**

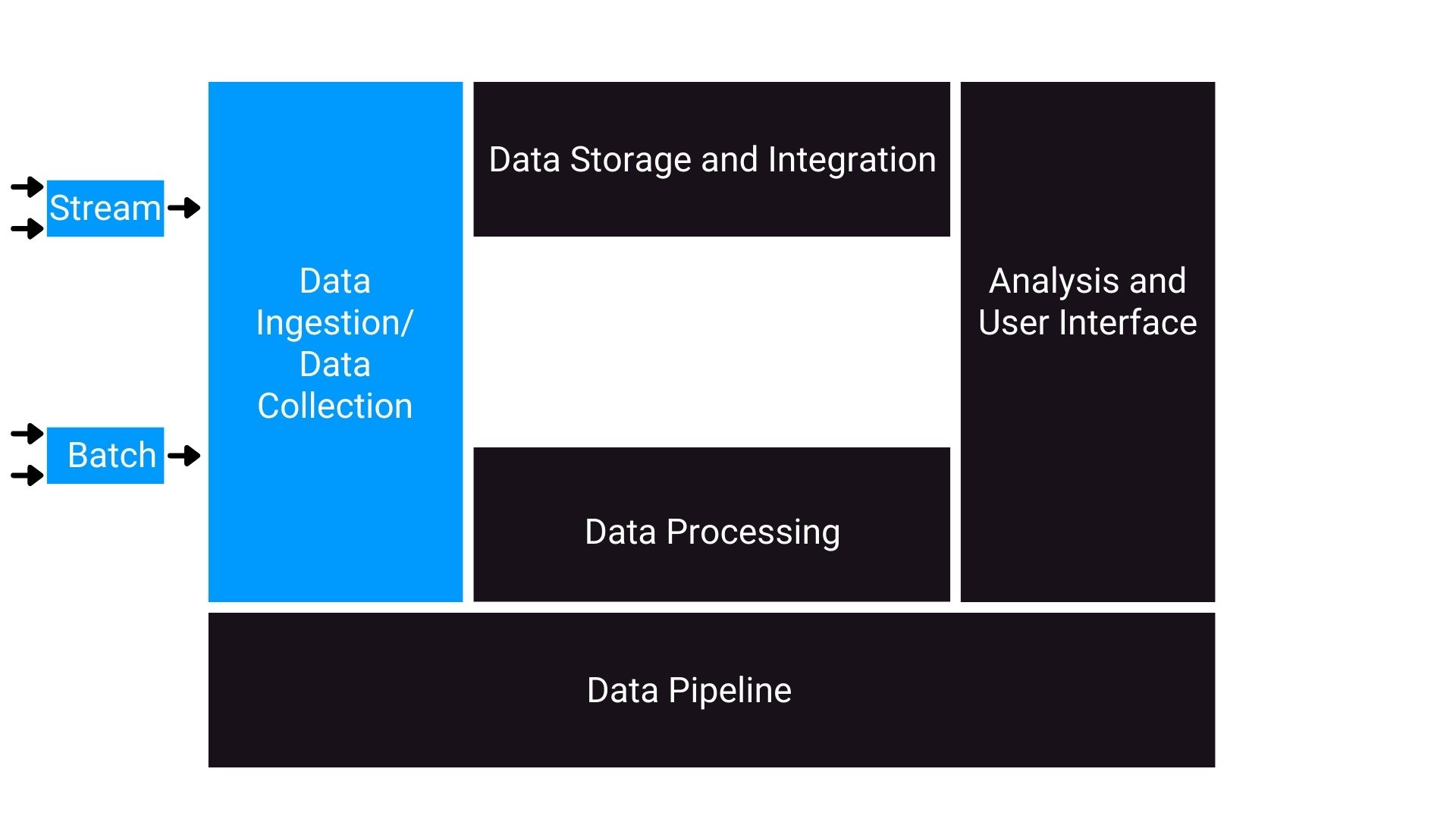
ETL and ELT are just two data integration methods, and there are other approaches that are also used to facilitate data integration workflows. Some of these include:

* **Change Data Capture (CDC)** identifies and captures only the source data that has changed and moves that data to the target system. CDC can be used to reduce the resources required during the ETL “extract” step; it can also be used independently to move data that has been transformed into a data lake or other repository in real time.
* **Data replication** copies changes in data sources in real time or in batches to a central database. Data replication is often listed as a data integration method. In fact, it is most often used to create backups for disaster recovery.
* **Data virtualization** uses a software abstraction layer to create a unified, integrated, fully usable view of data—without physically copying, transforming or loading the source data to a target system. Data virtualization functionality enables an organization to create virtual data warehouses, data lakes and data marts from the same source data for data storage without the expense and complexity of building and managing separate platforms for each. While data virtualization can be used alongside ETL, it is increasingly seen as an alternative to ETL and to other physical data integration methods.
* **Stream Data Integration (SDI)** is just what it sounds like—it continuously consumes data streams in real time, transforms them, and loads them to a target system for analysis. The key word here is continuously. Instead of integrating snapshots of data extracted from sources at a given time, SDI integrates data constantly as it becomes available. SDI enables a data store for powering analytics, machine learning and real-time applications for improving customer experience, fraud detection and more.

**Architecture OF DATA PLATEFORM IN DATA ENGINEERING**

There are 5 layers of Data platform architecture:

1. Data Collection Layer or Data ingestion Layer
2. Data Storage Layer or Integration Layer
3. Data Processing Layer
4. Analysis and User Interface Layer
5. Data Pipeline Layer



1. **DATA INGESTION LAYER**

This is the first layer of the data platform architecture. The Data Collection layer as the name suggests is responsible for connecting to the source systems and bringing data into the data platform in a periodic manner. This layer performs the following tasks:

1. This layer is responsible for connecting to the data sources.
2. This layer is responsible for transferring data from data sources to the data platform in streaming mode or batch mode or both.
3. Moreover, this layer is responsible for maintaining the information about the data collected in the metadata repository. For example, how much data is gobbled into the data platform and other descriptive information?

There are various tools that are available in the market, but some of the popular tools include Google Cloud Data Flow, IBM Streams, Amazon Kinesis and Apache Kafka are some of the tools that are used for data ingestion that supports both batch and streaming modes.

## 2. Data Storage and Data Integration Layer

This is the second layer of the data platform architecture. The Data collection layer as the name suggests is responsible for storing data for processing and long-term use. Moreover, this layer is also responsible for making data available for processing in both streaming and batch modes. As this layer is responsible for making data available for processing, it needs to be reliable, scalable, high-performing and cost-efficient. IBM DB2, IBM DB2, Microsoft SQL Server, MySQL, Oracle Database, and PostgreSQL are some of the popular relational databases. But nowadays, cloud-based relational databases gained popularity over the recent years, some cloud-based relational databases are IBM DB2, Google Cloud SQL and SQL Azure. In the NoSQL or non-relational database systems on the cloud, we have IBM Cloudant, Redis, MongoDB, Cassandra, and Neo4J. Tools for integration includes IBM’s cloud Pak for Data, IBM’s cloud Pak for Integration and Open Studio. Once the data has been ingested, stored and integrated, it needs to be processed. So with this, we move forward to Data Processing Layer.

## 3. Data Processing Layer

## This is the third layer of the data platform architecture. As the name suggests, this layer is responsible for a processing task. The processing includes data validations, transformations and applying business logic to the data. The processing layer should be able to perform some tasks that include:

## Read data in batch or streaming modes from storage and apply transformations.

## Support popular querying tools and programming languages.

## Scale to meet the processing demands of a growing dataset.

## Provide a way for analysts and data scientists to work with data in the data platform.

## The transformation task that usually occurs in this layer include:

## Structuring: These are the actions that change the structure of the data. This change can be simple or complex in nature. The simple one can also be like changing the arrangement of fields within the record or dataset or complex as combining fields complex structures using joins and unions.

## Normalization: This part focuses on reducing redundancy and inconsistency. It also focuses on cleaning the database of unused data.

## Denormalization: Denormalization is the task of combining data from multiple tables into a single table so that it can be queried more efficiently for reporting and analysis purposes.

## Data Cleaning: Data Cleaning, which fixes irregularities in data to provide credible data for downstream applications and uses.

## There are numerous amount of tools available in the market for performing these operations on the data includes Such as spreadsheets, OpenRefine, Google DataPrep, Watson Studio Refinery, and Trifacta Wrangler. Python and R also offer several libraries and packages that are explicitly created for processing data. It’s very important to know that storage and processing are not always been performed in separate layers. For example, in relational databases, storage and processing are both occur in the same layer while in Big data systems, data is first stored in the Hadoop File Distributed system and then processed in the data processing engine such as Spark.

## 4. Analysis and User Interface Layer

This is the fourth layer of the data platform architecture. This layer is responsible for delivering the process data to the end-users that including business intelligence analysts and business stakeholders who consume these data with the help of interactive dashboards and reports, moreover, data scientists and data analysts fall under this end-user category that further process this data for the specific use case. This layer needs to support querying tools such as SQL tools and No-SQL tools and programming languages like Python, R and Java and moreover, these layers need to support API’s that can be used to run reports on data for both online and offline processing.

## 5. Data Pipeline Layer

This is the last layer of this architecture, this layer is responsible for implementing and maintaining a continuous flow of data through this data pipeline. It is the layer that has the capability to extract, transform and load tools. There are a number of data pipeline solutions available, most popular among them being Apache Airflow and DataFlow.

**ETL TOOLS**

Many organizations today want to use data to guide their decisions but need help managing their growing data sources. More importantly, when they can't transform their raw data into usable formats, they may have poor data availability, which can hinder the development of a data culture.

ETL (Extract, Transform, Load) tools are an important part of solving these problems. There are many different ETL tools to choose from, which gives companies the power to select the best option. However, reviewing all the available options can be time-consuming.

**TOP 3 ETL TOOLS**

### **1. APACHE AIRFLOW**

Apache Airflow is an open-source platform to programmatically author, schedule, and monitor workflows. The platform features a web-based user interface and a command-line interface for managing and triggering workflows.

Workflows are defined using directed acyclic graphs (DAGs), which allow for clear visualization and management of tasks and dependencies. Airflow also integrates with other tools commonly used in data engineering and data science, such as Apache Spark and Pandas.

Companies using Airflow can benefit from its ability to scale and manage complex workflows, as well as its active open-source community and extensive documentation. You can learn about Airflow in the following DataCamp course.

### **2. AWS GLUE**

AWS Glue is a serverless ETL tool offered by Amazon. It discovers, prepares, integrates, and transforms data from multiple sources for analytics use cases. With no requirement to set up or manage infrastructure, AWS Glue promises to reduce the hefty cost of data integration.

Better yet, when interacting with AWS Glue, practitioners can choose between a drag-and-down GUI, a Jupyter notebook, or Python/Scala code. AWS Glue also offers support for various data processing and workloads that meet different business needs, including ETL, ELT, batch, and streaming.

### **3. AZURE DATA FACTORY**

Azure Data Factory is a cloud-based ETL service offered by Microsoft used to create workflows that move and transform data at scale.

It comprises a series of interconnected systems. Together, these systems allow engineers to not only ingest and transform data but also design, schedule, and monitor data pipelines.

The strength of Data Factory lies in the sheer number of its available connectors, from MySQL to AWS, MongoDB, Salesforce, and SAP. It is also lauded for its flexibility; users can choose to interact with either a no-code graphical user interface or a command-line interface.

**TASK 5 DATA ENGINEERING BASICS 16-03-23**

**TYPES OF ETL LOADS**

Data Load is the process that involves taking the transformed data and loading it where the users can access it.

There are 3 types of ETL Loads

**1. HISTORICAL LOAD**

The Data Warehouse/Mart is expected to house historical data. Based on the duration for how long the end users want to perform the analysis, we keep the data for that long. In other words, we would have users want to compare the Stores Monthly sales and compare it with the monthly sales of for the last 3 years. Here we would have to keep at least 3 years of data so that end users can perform their analysis. When we build and implement data warehouse/mart, it is empty. We would not want to start building the history from the day it is implemented. In this case the end user would have to wait for 3 years from the day of implementation to perform this particular analysis. Hence we identify the source where we can find the history data, and perform a onetime ETL to extract the required history data and load it to the warehouse.

**2. FULL LOAD**

With a full load, the entire dataset is dumped, or loaded, and is then completely replaced (i.e. deleted and replaced) with the new, updated dataset. No additional information, such as timestamps, is required.

For example, take a store that uploads all of its sales through the ETL process in data warehouse at the end of each day. Let’s say 5 sales were made on a Monday, so that on Monday night a table of 5 records would be uploaded. Then, on Tuesday, another 3 sales were made which need to be added. So on Tuesday night, assuming a full load, Monday’s 5 records as well as Tuesday’s 3 records are uploaded – an inefficient system, although relatively easy to set up and maintain. While this example is overly simplified, the principle is the same.

**3. INCREAMENTAL LOAD**

Only the difference between the target and source data is loaded through the ETL process in data warehouse. There are 2 types of incremental loads, depending on the volume of data you’re loading; streaming incremental load and batch incremental load.

Following the previous example, the store that made 3 sales on Tuesday will load only the additional 3 records to the sales table, instead of reloading all records. This has the advantage of saving time and resources, but increases complexity.

Incremental loading is of course much faster than a full load. The main drawback to this type of loading is maintainability. Unlike a full load, with an incremental load you can’t re-run the entire load if there’s an error. In addition to this, files need to be loaded in order, so errors will compound the issue as other data queues up.