UNIT 4 EXTRACT, TRANSFORM AND LOADING

4.0 INTRODUCTION

A data warehouse is a digital storage system that connects and harmonizes large amounts of data from many different sources. Data warehouses store current and historical data in one place and act as the single source for an organization. A typical data warehouse has four main components namely:

Central database: A database serves as the foundation of your data warehouse. Traditionally, these have been standard relational databases running on premise or in the cloud. But because of Big Data, the need for true, real-time performance, and a drastic reduction in the cost of RAM, in-memory databases are rapidly gaining in popularity.

Data integration: Data is pulled from source systems and modified to align the information for rapid analytical consumption using a variety of data integration approaches such as ETL (extract, transform, load) and ELT as well as real-time data replication, bulk-load processing, data transformation, and data quality and enrichment services.

Metadata: Metadata is data about your data. It specifies the source, usage, values, and other features of the data sets in your data warehouse. There is business metadata, which adds context to your data, and technical metadata,

which describes how to access data – including where it resides and how it is structured.

Data warehouse access tools: Access tools allow users to interact with the data in your data warehouse. Examples of access tools include: query and reporting tools, application development tools, data mining tools, and OLAP tools.
All these components are engineered for speed so that you can get results quickly and analyze the data within no time.

In this unit, we will study about Data Integration component approach such as Extract, Transform and Load (ETL) in detail.

4.1 OBJECTIVES

After going through this unit, you shall be able to:

- Understand the purpose ETL;
- Describe the ETL process, benefits and ETL tools;
- Know the complete working of the ETL
- Discuss various layers involved in the ETL implementation;
- To summarize the functionality of ELT, its need and benefits, and
- To compare and contrast the ETL with ELT.

4.2 ETL AND ITS NEED

Extract, Transform, Load (ETL) as shown in Figure 1, is a process of data integration that encompasses three steps - ***extraction, transformation,*** and ***loading***. In a nutshell, ETL systems take large volumes of raw data from multiple sources, convert it for analysis, and load that data into your warehouse.

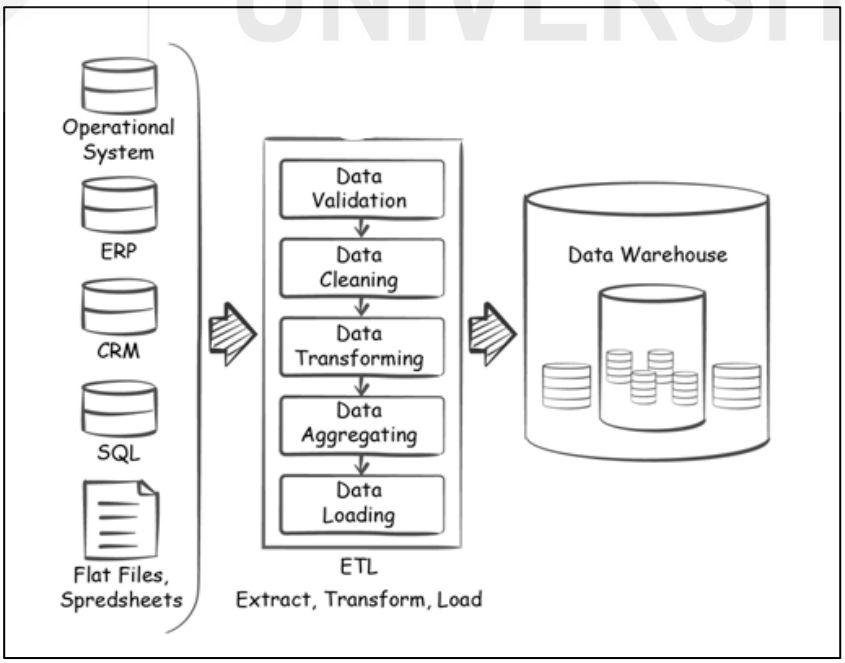

Figure 1: ETL in a Data Warehouse

4.2.1 Why Do You Need ETL?

ETL saves you significant time on data extraction and preparation - time that you can better spend on evaluating your business. Practicing ETL is also part of a healthy data management workflow, ensuring high data quality, availability, and reliability. Each of the three major components in the ETL saves time and development effort by running just once in a dedicated data flow:

Extract: In ETL, the first link determines the strength of the chain. The extract stage determines which data sources to use, the refresh rate (velocity) of each source, and the priorities (extract order) between them — all of which heavily impact your time to insight.

Transform: After extraction, the transformation process brings clarity and order to the initial data swamp. Dates and times combine into a single format and strings parse down into their true underlying meanings. Location data convert to coordinates, zip codes, or cities/countries. The transform step also sums up, rounds, and averages measures, and it deletes useless data and errors or discards them for later inspection. It can also mask personally identifiable information (PII) to comply with GDPR, CCPA, and other privacy requirements.

Load: In the last phase, much as in the first, ETL determines targets and refresh rates. The load phase also determines whether loading will happen incrementally, or if it will require "upsert" (updating existing data and inserting new data) for the new batches of data.

4.3 ETL PROCESS

ETL collects and processes data from various sources into a single data store (a data warehouse or data lake), making it much easier to analyze. The three steps in ETL process are mentioned below:

4.3.1 Data Extraction

Data extraction involves the following four steps:

Identify the data to extract: The first step of data extraction is to identify the data sources you want to incorporate into your data warehouse. These sources might be from relational SQL databases like MySQL or non-relational NoSQL databases like MongoDB or Cassandra. The information could also be from a SaaS platform like Salesforce or other applications. After identifying the data sources, you need to determine the specific data fields you want to extract.

Estimate how large the data extraction is: The size of the data extraction matters. Are you extracting 50 megabytes, 50 gigabytes, or 50 petabytes of data? A larger quantity of data will require a different ETL

strategy. For example, you can make a larger dataset more manageable by aggregating it to month-level rather than day-level, which reduces the size of the extraction. Alternatively, you can upgrade your hardware to handle the larger dataset.

Choose the extraction method: Since data warehouses need to update continually for the most accurate reports, data extraction is an ongoing process that may need to happen on a minute-by-minute basis. There are three principal methods for extracting information:

(a) *Update notifications*: The preferred method of extraction involves update notifications. The source system will send a notification when one of its records has changed, and then the data warehouse updates with only the new information.
(b) *Incremental extraction:* The second method, which you can use when update notifications aren't possible, is incremental extraction. This involves identifying which records have changed and performing extraction of only those records. A potential setback is that incremental extraction cannot always identify deleted records.
(c) *Full extraction:* When the first two methods won't work, a complete update of all the data through full extraction is necessary. Keep in mind that this method is likely only feasible for smaller data sets.

Assess your SaaS platforms: Businesses formerly relied on in-house applications for accounting and other record-keeping. These applications used OLTP transactional databases that they maintained on an on-site server. Today, more businesses use SaaS(software as a service) platforms like Google Analytics, HubSpot, and Salesforce. To pull data from one of these, you'll need a solution that integrates with the unique API of the platform. Xplenty is one such solution.

4.3.2 Data Transformation

In traditional ETL strategies, data transformation that occurs in a staging area (after extraction) is "multistage data transformation". In ELT, data transformation that happens after loading data into the data warehouse is "in-warehouse data transformation". You may need to perform some of the following data transformations:

Deduplication (normalizing): Identifies and removes duplicate information.

Key restructuring: Draws key connections from one table to another.

Cleansing: Involves deleting old, incomplete, and duplicate data to maximize data accuracy - perhaps through parsing to remove syntax errors, typos, and fragments of records.

Format revision: Converts formats in different datasets - like date/time, male/female, and units of measurement - into one consistent format.

Derivation: Creates transformation rules that apply to the data. For example, maybe you need to subtract certain costs or tax liabilities from business revenue figures before analyzing them.

Aggregation: Gathers and searches data so you can present it in a summarized report format.

Integration: Reconciles diverse names/values that apply to the same data elements across the data warehouse so that each element has a standard name and definition.

Filtering: Selects specific columns, rows, and fields within a dataset.

Splitting: Splits one column into more than one column.

Joining: Links data from two or more sources, such as adding spend information across multiple SaaS platforms.

Summarization: Creates different business metrics by calculating value totals. For example, you might add up all the sales made by a specific salesperson to create total sales metrics for specific periods.

Validation: Sets up automated rules to follow in different circumstances. For instance, if the first five fields in a row are NULL, then you can flag the row for investigation or prevent it from being processed with the rest of the information.

4.3.3 Data Loading

Data loading is the process of loading the extracted information into your target data repository. Loading is an ongoing process that could happen through “full loading” (the first time you load data into the warehouse) or “incremental loading” (as you update the data warehouse with new information). Because incremental loads are the most complex, we'll focus on them in this section.

4.3.3.1 Types of Incremental Loads

Incremental loads extract and load information that has appeared since the last incremental load. This can happen in two ways: (a) Batch incremental loads and (b) Streaming incremental loads.

(a) Batch incremental loads: The data warehouse ingests information in packets or batches. If it's a large batch, it's best to carry out a batch load during off-peak hours - on a daily, weekly, or monthly basis - to prevent system slowdowns. However, modern data warehouses can also ingest small batches of information on a minute-by-minute basis with an ETL platform like Xplenty. This allows them to achieve an approximation of real-time updates for the end-user.

(b) Streaming incremental loads: The data warehouse ingests new data as it appears in real-time. This method is particularly valuable when the end-user requires real-time updates (for example: for up-to-the-minute decision-making). Further, streaming incremental loads are only possible when the updates involve a very small amount of data. In most cases, minute-by-minute batch updates offer a more robust solution than real-time streaming.

4.3.3.2 Challenges in Incremental Loading

Incremental loads can disrupt system performance and cause a host of problems, including:

Data structure changes: Data formats in your data sources or data warehouse may need to evolve according to the needs of your information system. However, changing one part of the system could lead to incompatibilities that interfere with the loading process. To prevent problems relating to inconsistent, corrupt, or incongruent data, it's important to zoom out and review how slight changes affect the total ecosystem before making the appropriate adjustments.

Processing data in the wrong order: Data pipelines can follow complex trajectories that result in your data warehouse processing, updating, or deleting information in the wrong order. That can lead to corrupt or inaccurate information. For this reason, it's vital to monitor and audit the ordering of data processing.

Failure to detect problems: Quick detection of any problems with your ETL workflow is crucial: e.g. when an API goes down, when your API access credentials are out-of-date, when system slowdowns interrupt dataflow from an API or when the target data warehouse is down. The sooner you detect the problem, the faster you can fix it, and the easier it is to correct the inaccurate/corrupt data that results from it.

4.4 WORKING OF ETL

In this section, we'll dive a little deeper, taking an in-depth look at each of the three steps in the ETL process.

You can use scripts to implement ETL (i.e. custom *do it yourself* code) or you can use a dedicated ETL tool. An ETL system performs a number of important functions, including:

(a) Parsing/Cleansing: Data generated by applications may be in various formats like JSON, XML, or CSV. The parsing stage maps data into a table format with headers, columns, and rows, and then extracts specified fields.

(b) Data enrichment: Preparing data for analytics usually requires certain data enrichment steps, including injecting expert knowledge, resolving discrepancies, and correcting bugs.

(c) Setting velocity: "Velocity" refers to the frequency of data loading, i.e. inserting new data and updating existing data.

(d) Data validation: In some cases, data is empty, corrupted, or missing crucial elements. During data validation, ETL finds these occurrences and determines whether to stop the entire process, skip the data or set the data aside for human inspection.

4.4.1 Layered Implementation of ETL in a Data Warehouse

When an ETL process is used to move data into a data warehouse, a separate layer represents each phase:

(a) Mirror/Raw layer: This layer is a copy of the source files or tables, with no logic or enrichment. The process copies and adds source data to the target mirror tables, which then hold historical raw data that is ready to be transformed.

(b) Staging layer: Once the raw data from the mirror tables transform, all transformations wind up in staging tables. These tables hold the final form of the data for the incremental part of the ETL cycle in progress.

(c) Schema layer: These are the destination tables, which contain all the data in its final form after cleansing, enrichment, and transformation.

(d) Aggregating layer: In some cases, it's beneficial to aggregate data to a daily or store level from the full dataset. This can improve report performance, enable the addition of business logic to calculate measures, and make it easier for report developers to understand the data.

4.5 ETL AND OLAP DATA WAREHOUSES

Data engineers have been using ETL for over two decades to integrate diverse types of data into online analytical processing (OLAP) data warehouses. The reason for doing this is simple: to make data analysis easier. Normally, business applications use online transactional processing (OLTP) database systems. These are optimized for writing, updating, and editing the information inside them. They are not good at reading and analysis. However, online analytical processing database systems are excellent at high-speed reading and analysis. That's why ETL is necessary to transform OLTP information, so it can work with an OLAP data warehouse.

During the ETL process, information is:

i. Extracted from various relational database systems (OLTP or RDBMS) and other sources.

ii. Transformed within a staging area, into a compatible relational format, and integrated with other data sources.

iii. Loaded into the online analytical processing (OLAP) data warehouse server.

In the past, data engineers hand-coded ETL pipelines in R, Python, and SQL - a laborious process that could take months to complete. Today, hand-coded ETL continues to be necessary in many cases. However, modern ETL solutions like Xplenty allow data teams to skip hand-coding and automatically integrate the most popular data sources into their data warehouses. This has dramatically increased the speed of setting up an ETL pipeline, while eliminating the risk of human error.

In the next section let us focus on the benefits of the ETL tools.

4.6 ETL TOOLS AND THEIR BENEFITS

ETL tools come in a wide variety both in open source or proprietary categories. There are ETL frameworks and libraries that you can use to build ETL pipelines in Python. There are tools and frameworks you can leverage for GO and Hadoop. Really, there is an open-source ETL tool out there for almost any unique ETL need. The downside, of course, is that you'll need lots of custom coding, setup, and man-hours getting the ETL operational. Even then, you may find that you need to tweak your ETL stack whenever you introduce additional tasks. Following are some of the benefits of the ETL tools:

- **Scalability**: Trying to scale-out hand-coded ETL solutions is difficult. As schema complexity rises and your tasks grow more complex and resource-hungry, establishing solid pipelines and deploying the necessary ETL resources can become impossible. With cloud-based ETL tools like Xplenty, you have unlimited scalability at the click of a button.

- **Simplicity**: Going from a hand-coded ETL solution using SQLAlchemy and pandas with rpy2 and parse to something as simple as a cloud-based ETL can be lifechanging. The benefits of having all of your needs layered into one tool saves you time, resources, and lots of headaches.

- **Out-of-the-box**: While open source ETL tools like Apache Airflow require some customization, cloud-based ETL tools like Xplenty work out-of-the-box.

- **Compliance**: The overwhelming nature of modern data compliance can be frightening. Between GDPR, CCPA, HIPAA, and all of the other compliance and privacy nets, using an ETL tool that bakes compliance into its framework is an easy way to skip difficult and risky compliance setups.

- **Long-term costs**: Hand-coded solutions may be cheaper up-front, but they will cost you in the long run. The same thing could be said about open source ETL tools. Since you have to spend time and energy on modification, you're forced to onboard early or risk delaying project launches. Cloud-based ETL tools handle maintenance and back-end caretaking for you.

4.7 IMPROVING THE PERFORMANCE OF ETL

Ultimately tuning is very much required for the ETL to perform better. Following are some of the factors to be considered to improve the ETL performance:

(i) Tackle the Bottlenecks

Before anything else, make sure you log metrics such as time, the number of records processed, and hardware usage. Check how many resources each part of the process takes and address the heaviest one. Usually, it will be the second part, building facts, and dimensions in the staging environment.

(ii) Load Data Incrementally

Loading only the changes between the previous and the new data saves a lot of time as compared to a full load. It's more difficult to implement and maintain, but difficult doesn't mean impossible, so do consider it. Loading incrementally can definitely improve the ETL performance.

(iii) Partition Large Tables

If you use relational databases and you want to improve the data processing window, you can partition large tables. That is, cut big tables down to physically smaller ones, probably by date. Each partition has its own indices and the indices tree is shallower thus allowing for quicker access to the data. It also allows switching data in and out of a table in a quick metadata operation instead of actual insertion or deletion of data records.

(iv) Cut Out Extraneous Data

It's important to collect as much data as possible, but not all of it is worthy enough to enter the data warehouse. To improve the ETL performance, define exactly which data should be processed and leave irrelevant rows/columns out. Better to start small and grow as you go as opposed to creating a giant data that takes much time to process.

(v) Cache the Data

Caching data can greatly speed things up since memory access performs faster than do hard drives. Note that caching is limited by the maximum amount of memory your hardware supports.

(vi) Process in Parallel using Hadoop

Instead of processing serially, optimize resources by processing in parallel. Sort and aggregate functions (count, sum, etc.) block processing because they must end before the next task can begin. Even if you can process in parallel, it won't help if the machine is running on 100% CPU the entire time. You could scale up by upgrading the CPU, but it would scale only to a limit. Hadoop is a much better solution.

Apache Hadoop is designed for the distributed processing of large data over a cluster of machines. It uses HDFS, a dedicated file system that cuts data into small chunks and optimally spreads them over the cluster. Duplicate copies are kept and the system maintains integrity automatically.

MapReduce is used to process tasks (Hadoop 2 or YARN allows more applications). Each MapReduce job works in 2 stages:

(a) Map - filtering and sorting data - tasks are divided into sub-tasks and processed in parallel by the cluster machines.

(b) Reduce - summary operations - data from the previous stage is combined.

Hadoop is optimized for distributed processing analytics. Sort and aggregate functions execute in parallel on an entire cluster.

4.8 ELT AND ITS NEED

Extract/load/transform (ELT) is the process of extracting data from one or multiple sources and loading it into a target data warehouse. Instead of transforming the data before it's written, ELT takes advantage of the target system to do the data transformation. This approach requires fewer remote sources than other techniques because it needs only raw and unprepared data. The process is illustrated in Figure 2.

ELT is an alternative to the traditional extract/transform/load (ETL) process. It pushes the transformation component of the process to the target database for better performance. This capability is very useful for processing the massive data sets needed for business intelligence (BI) and big data analytics.

Because it takes advantage of the processing capability already built into a data storage infrastructure, ELT reduces the time data spends in transit and boosts efficiency.

It is becoming increasingly common for data to be extracted from its source locations, then loaded into a target data warehouse to be transformed into actionable business intelligence. ELT process consists of three steps:

a. Extract - This step works similarly in both ETL and ELT data management approaches. Raw streams of data from virtual infrastructure, software, and applications are ingested either in their entirety or according to predefined rules.

b. Load – The ELT differs here with the ETL. Rather than deliver this mass of raw data and load it to an interim processing server for transformation, ELT delivers it directly to the target storage location. This shortens the cycle between extraction and delivery.

c. Transform - The database or data warehouse sorts and normalizes the data, keeping part or all of it on hand and accessible for customized reporting. The overhead for storing this much data is higher, but it offers more opportunities to mine it for relevant business intelligence in near real-time.

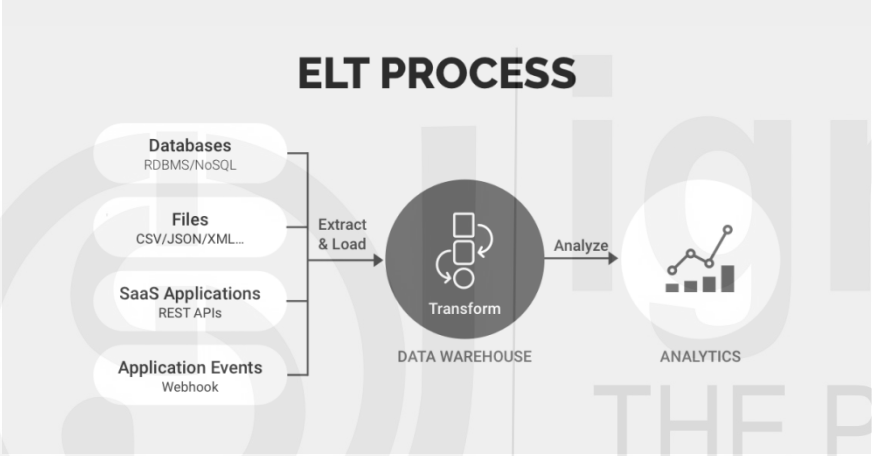

Figure 2: ELT Process

4.8.1 Why Do You Need ELT?

Transforming data after uploading it to modern cloud ecosystems is most effective for:

- Large enterprises with vast data volumes
- Businesses that collect data from multiple source systems or in dissimilar formats
- Companies that require quick or frequent access to integrated data
- Data scientists who rely on business intelligence
- IT departments and data stewards interested in a low-maintenance solution

The ELT process improves data conversion and manipulation capabilities due to parallel load and data transformation functionality. This schema allows data to be accessed and queried in near real time.

However, you might want to stick with ETL if you have dirty data (e.g., duplicate records, incomplete or inaccurate data) that will require data engineers to clean and format it prior to data loading.

4.8.2 Benefits of ELT

With traditional ETL, relevant data is transformed before it is uploaded to a data warehouse, and then it must be pushed out of the warehouse for analysis or processing. This data pipeline works, but it can take more time to migrate data from the source to the target system.

The ELT process saves you steps and time. Data is first loaded into the target ecosystem, such as a data warehouse, and then transformed. Authorized users can securely access the data without returning it to source systems. No downloading is necessary for it. There are reasons to continue using ETL tools. For example, some companies want to keep all their data on-premises. If there is a small amount of data, and it is relational and structured, traditional ETL is effective for businesses that favor hands-on data integration. However, the ELT approach has several benefits for most industries which are listed below:

a) Get better results with more efficient effort

ELT allows you to integrate and process large amounts of data, both structured and unstructured from multiple servers. And, both raw and cleansed data can be accessed with artificial intelligence (AI) and machine learning (ML) tools in addition to SQL and NoSQL processing.

b) Transform your data faster

ELT doesn't have to wait for the data to be transformed and then loaded. The transformation process happens where the data resides, so you can access your data in a few seconds, a huge benefit when processing time-sensitive data.

c) Combine data from different sources and formats

Larger enterprises typically have multiple, disparate data sources like onsite servers, cloud warehouses and log files. Using ELT means you can combine data from various data sets regardless of the source or whether it is structured or unstructured, related or unrelated.

d) Manage data at scale

Technological advances allow organizations to collect petabytes (a million gigabytes!) of data. ELT streamlines the management of massive amounts of data by allowing raw and cleansed data to be stored and accessed. If you're planning to use cloud-based data warehousing or high-end data processing engines like Hadoop, ELT can take advantage of the native processing power for greater scalability.

e) Save time and money

ELT reduces the time data spends in transit and doesn't require an interim data system or additional remote resources to transform the data outside the cloud. Plus, there's no need to move data in and out of cloud ecosystems for analysis. The more your data moves around, the more the costs add up. The scalability of ELT makes it cost-effective for businesses of any size.

4.8.3 ETL Vs ELT

The primary differences between ETL and ELT are how much data is retained in data warehouses and where data is transformed. With ETL, the transformation of data is done before it is loaded into a data warehouse. This enables analysts and business users to get the data they need faster, without building complex transformations or persistent tables in their business intelligence tools. Using the ELT approach, data is loaded into the warehouse as is, with no transformation before loading. This makes jobs easier to configure because it only requires an origin and a destination.

The ETL and ELT approaches to data integration differ in several key ways as listed below:

- **Load time**

  It takes significantly longer to get data from source systems to the target system with ETL.

- **Transformation time**

  ELT performs data transformation on-demand, using the target system's computing power, reducing wait times for transformation.

- **Complexity**

  ETL tools typically have an easy-to-use GUI that simplifies the process. ELT requires in-depth knowledge of BI tools, masses of raw data, and a database that can transform it effectively.

- **Data warehouse support**

  ETL is a better fit for legacy on-premise data warehouses and structured data. ELT is designed for the scalability of the cloud.

- **Maintenance**

  ETL requires significant maintenance for updating data in the data warehouse. With ELT, data is always available in near real-time.

Both ETL and ELT processes have their importance in the data warehouse architecture as per the understanding of business unique needs and strategies which is key to determining which process will deliver the best results.

Check your Progress 1

1. Define Data Extraction process of ELT along the variety of sources of data accounted for this process. Also, mention the challenge(s) for an ETL tool during the extracting process.

.....
.....
.....
.....

2. Describe the Data Transformation process involved in the ELT.

.....
.....
.....
.....

3. Discuss briefly the Data Loading process of ELT.

.....
.....
.....
.....

4.9 SUMMARY

The process of extracting data from source systems and bringing it into the data warehouse is commonly called ETL process, which stands for extraction, transformation, and loading. In this unit, we had studied that ETL refers to a broad process. The methodology and tasks of ETL have also been studied.

Apart from the ELT, we had studied a new approach known as ELT and its benefits too, in this unit.

In the next unit, we will be study about Online Analytical Processing (OLAP).

4.10 SOLUTIONS / ANSWERS

Check your Progress 1:

1. Extracting data is the act of pulling data from one or more data sources. During the extraction phase of ETL, you may handle a variety of sources with data, such as:

   - Relational and non-relational databases
   - Flat files (e.g. XML, JSON, CSV, Microsoft Excel spreadsheets, etc.)

- SaaS applications, such as CRM (customer relationship management) and ERP (enterprise resource planning) systems
- APIs (application programming interfaces)
- Websites
- Analytics and monitoring tools
- System logs and metadata

In the first step, extracted data sets come from a source (*e.g., Salesforce, Google AdWords, etc.*) into a staging area. The staging area acts as a buffer between the data warehouse and the source data. Since data may be coming from multiple different sources, it's likely in various formats, and directly transferring the data to the warehouse may result in corrupted data. The staging area is used for data cleansing and organization.

A big challenge during the data extraction process is how your ETL tool handles structured and unstructured data. All of those unstructured items (e.g., emails, web pages, etc.) can be difficult to extract without the right tool, and you may have to create a custom solution to assist you in transferring unstructured data if you chose a tool with poor unstructured data capabilities.

2. It's rarely the case that your extracted data is already in the exact format that you need it to be. For example, it may requires the following:

   - Rearrange unstructured data into a structured format.
   - Limit the data you've extracted to just a few fields.
   - Sort the data so that all the columns are in a certain order.
   - Join multiple tables together.
   - Clean the data to eliminate duplicate and out-of-date records.

   The data cleaning and organization stage is the transformation stage. All of that data from multiple source systems will be normalized and converted to a single system format — improving data quality and compliance. ETL yields transformed data through these methods:

   - Cleaning
   - Filtering
   - Joining
   - Sorting
   - Splitting
   - Deduplication
   - Summarization

3. Once the process has transformed, sorted, cleaned, validated, and prepared the data, you need to load it into data storage somewhere and the most common target database is a data warehouse. Depending upon your business needs, data can be loaded in batches or all at once. The exact nature of the loading will depend upon the data source, ETL tools, and various other factors.

ETL architecture often consists of a diagram that outlines the flow of information from start to finish in your ETL pipeline. Data originates from sources files and databases, where it's taken up and entered into the ETL transformation engine. From here, the data may be loaded into any or all of the following locations:

- **Landing area:** The landing area is where data first arrives after being extracted from a source location. Few, if any, transformations are applied to the data in the landing area. If you are performing ETL batch processing, the landing area may store multiple batches of data before moving it through the ETL pipeline.
- **Staging area:** The staging area is a temporary, intermediate location for performing ETL transformations. This area may take the form of a relational database, or binary or text files. The transformations performed in this area include joining and consolidating multiple data sources, cleansing and validating source data, and standardizing and aligning information.
- **Data warehouse area:** The data warehouse area is the final destination for data in an ETL pipeline. From here, your data can easily be queried and analyzed in order to obtain valuable insights and make better business decisions. This area may consist of an enterprise data warehouse spanning the entire organization, or a data mart that has been set up to serve the needs of a single team or department.

2.11 FURTHER READINGS

1. William H. Inmon, Building the Data Warehouse, Wiley, 4th Edition, 2005.
2. Data Warehousing Fundamentals, Paulraj Ponnaiah, Wiley Student Edition, 2001.
3. Data Warehousing, Reema Thareja, Oxford University Press, 2009.