BYTEWISE LIMITED

Data Engineering Track

Task: Week – 1 (First Month)

Task No: 4 & 5

Task Date: 16-03-2023

Internee Name: Umer Farooq

Mentor Name: Ahtisham

Task Details:

This task includes the following:

1. What is ETL? in detail.
2. What is ELT? in detail.
3. 3 Tier Architecture in DE
4. ETL Tools (any 3)

   After that, also explain the following:

5. What is Historical Load
6. What is Full Load
7. What is Incremental Load

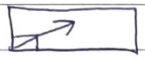
1.ETL:

12410
11100

Date: M T W T F S S

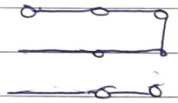
Topic :- ETL , ELT , & Data Pipelines

In this topic, we will learn about diff tools & processes that work to move data from source to destination systems, such as :



ETL

ELT



Data Pipelines

① : ETL :

- ETL is the process that is at the heart of gaining value from data.
- ETL is how raw data is converted into analysis - ready data.
- ETL is an automated process which include:
  - Gathering the raw data.
  - Extracting information needed for reporting & analysis.

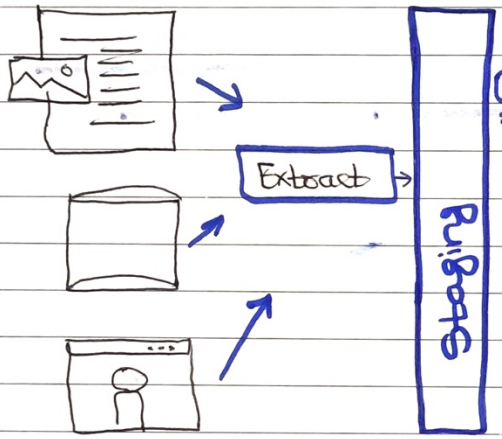
NADEEM

Date: _____ M T W T F S S

- Cleaning, standardizing, & transforming data into usable format.
- Loading data into a data repository.

→ ETL is generic process, actual job can be very different in usage, utility, & complexity.

Extract: (Its a step where data from source locations is collected for transformations)



Extraction can be through:

i) Batch-Processing: Large Chunks of data moved from source to destination at scheduled intervals.

- Tools for Batch-Processing: Stitch & Blendo

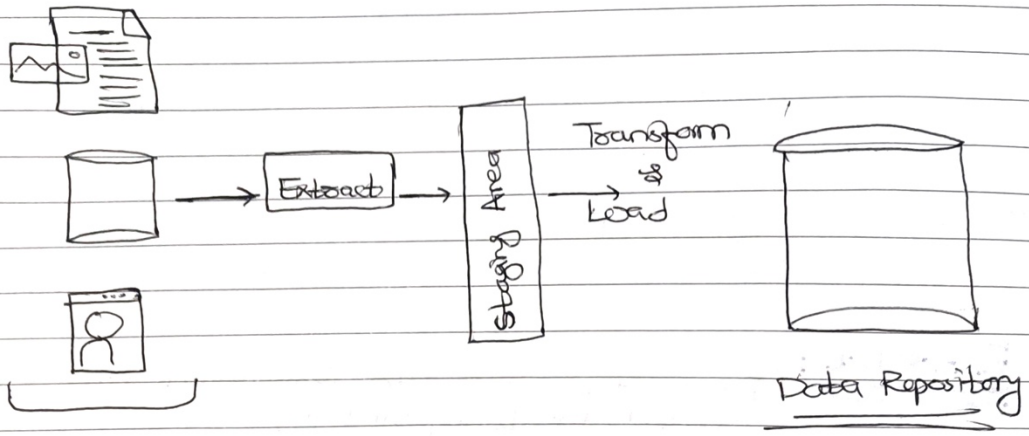
ii) Stream-Processing: Data pulled-in real-time from source, transformed in transit, & loaded into data repository.

- Tools for Stream Processing: Samza, Apache Storm, Apache Kafka.

NADEEM

Date: _____

M T W T F S S



Data Sources

- **Transform** involves the execution of rules & functions that convert "Raw Data" into data that can be "used for Analysis".

Transforming Data:

- Standardizing date formats & units of measurements.
- Removing duplicate data.
- Filtering out data that is not required.
- Enriching Data (e.g: Splitting full name to first, middle & last names.)
- Establishing key relationships across tables.
- Applying business rules & data validation.

NADEEM

Date: _____ M T W T F S S

- **Load** is the step where processed data is transported to a destination system or data repository.
  - It can be:
    - Initial loading: populating all of the data in the repository.
    - Incremental-loading: applying updates & modifications periodically.
    - Full Refresh: erasing data table & reloading fresh data.

- **Load Verification:-**
  Load verification includes checks for:
  - Missing / null values
  - Server Performance
  - Load failures

  are important parts of this process step.
- It is vital to keep an eye on "Load failures" and ensure the right recovery mechanisms are in place.

NADEEM

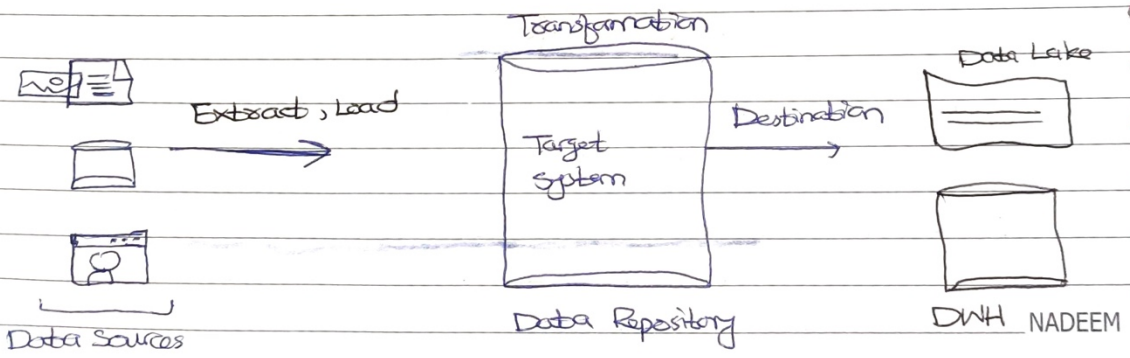
Date: _____ M T W T F S S

- ETL was used for batch workloads on a large scale.
  - However, with the emergence of Streaming ETL, ETL increasingly used for real-time streaming event data as well.
- Popular ETL tools available include:
  - IBM Infosphere Information Server.
  - AWS Glue. Google Cloud Dataflow
  - Informatica.

2.ELT:

ELT (Extract, Load, Transform):

In ELT process, extracted data is first loaded into the target system, & transformation are applied in the target system.



NADEEM

Date: _____ M T W T F S S

- The destination system for an ELT pipeline is most likely a data lake, although it can a data warehouse.

Why We Need ELT :-

- The ELT process :
  - Useful in processing large sets of unstructured & relational data.
  - It is ideal for Data Lakes (where transformations are applied once the raw data is loaded into the data lake).

(Pros of ELT) :-

- It is more suited to work with Big data.
- Shortens the cycle b/w extraction & delivery.
- Allows to ingest volumes of raw data as immediately as the data becomes available.
- Transforms only data that required for particular user-case, so it can be leveraged for multiple use cases.

NADEEM

ELT TOOLS:

There are several tools available for ELT (Extract, Load, Transform) processes. Some popular tools include:

1. Apache Nifi: An open-source data integration tool that provides a web-based interface to design and manage data flows. It supports data ingestion from various sources, transformation, and loading into different destinations.

2. Talend: A data integration platform that provides features for data mapping, transformation, and loading. Talend supports various data sources and destinations, including cloud-based services like AWS, Azure, and Google Cloud.

3. Matillion: A cloud-based data integration platform that provides ELT capabilities for popular cloud data warehouses like Amazon Redshift, Snowflake, and Google BigQuery. Matillion provides a visual interface for designing and managing data pipelines.

4. Stitch: A cloud-based ETL service that supports various data sources and destinations. Stitch provides pre-built connectors for popular data sources and destinations like Salesforce, Google Analytics, and Amazon Redshift.

3.Three-Tier Architecture:

The 3-tier architecture is a common design pattern used in data engineering to organize the flow of data from source systems to the end user. It consists of three tiers: the Staging tier, the Integration tier, and the Presentation tier.

1. Staging Tier: The Staging tier is the first tier in the architecture, where data is initially collected from various sources such as databases, APIs, logs, or files. This tier is responsible for extracting and loading data from the source systems and preparing it for further processing. The data in the Staging tier is typically stored in its raw form without any significant transformations or cleaning.

Example: A company wants to analyze customer behavior on its website. The website generates logs that contain information about the pages visited, products viewed, and time spent on the site. These logs are collected and stored in the Staging tier, where they are prepared for further processing.

2. Integration Tier: The Integration tier is the second tier in the architecture, where data from different sources is combined, transformed, and loaded into a centralized data repository. This tier is responsible for cleaning, enriching, and transforming the data to make it more usable and consistent. The data in the

Integration tier is typically **stored in a structured format**, such as a relational database or a data warehouse.

Example: In the customer behavior analysis example, data from the Staging tier is integrated and transformed to create a more comprehensive picture of customer behavior. This might involve combining website logs with data from other sources, such as customer purchase history or demographic information. The integrated data is then stored in a centralized database for further analysis.

3. Presentation Tier: The Presentation tier is the third tier in the architecture, where data is presented to end-users in a format that is easy to understand and analyze. This tier is responsible for creating dashboards, reports, and other visualizations that enable users to explore and analyze the data. The data in the Presentation tier is typically stored in a format that is optimized for querying and analysis.

Example: In the customer behavior analysis example, the integrated data is presented to business users in the form of a dashboard that provides insights into customer behavior. The dashboard might include metrics such as page views, conversion rates, and customer retention. Users can interact with the dashboard to explore the data and gain insights into customer behavior.

Overall, the 3-tier architecture provides a scalable and modular framework for managing the flow of data through an organization, from raw data to actionable insights.

4.ETL TOOLS:

There are several Cloud-Based and Open-Source ETL (Extract, Transform, Load) tools available. Here are a few examples:

Cloud-Based ETL Tools:

1. **AWS Glue:** This is a fully-managed ETL service that makes it easy to move data between data stores, clean and enrich data, and load it into data warehouses.
2. **Google Cloud Dataflow:** This is a fully-managed service for executing batch and streaming data processing pipelines.
3. **Microsoft Azure Data Factory:** This is a cloud-based data integration service that allows you to create, schedule and orchestrate your ETL/ELT workflows.
4. **Talend Cloud:** This is a cloud-based integration platform that includes ETL and data integration capabilities.

Open-Source ETL Tools:

1. **Apache NiFi:** This is a dataflow system that supports powerful and scalable directed graphs of data routing, transformation, and system mediation logic.
2. **Apache Kafka:** This is a distributed streaming platform that allows you to build real-time streaming data pipelines and applications.
3. **Apache Airflow:** This is a platform to programmatically author, schedule, and monitor workflows.
4. **Pentaho Data Integration (Kettle):** This is an open-source ETL tool that allows you to move data from different sources and transform it to fit your needs.
5. **CloverETL:** This is a data integration platform that allows you to design, automate, and operate data workflows.

Historical load, incremental load, and full load are different types of data loading techniques used in data engineering.

1. What is Historical Load?

A historical load **is a one-time load of all the data** from a source system into a destination system. It is typically used when setting up a new system or when backfilling historical data.

Historical loads are usually performed when migrating data from one system to another, such as when moving data from an old database to a new one, or when setting up a new data warehouse. The historical load involves extracting all the data from the source system, transforming it into the appropriate format, and then loading it into the destination system.

The process of performing a historical load can be **complex and time-consuming**, especially when dealing with large datasets. Some best practices to follow when performing a historical load include:

1. Validate the data: Ensure that the data being loaded is accurate and complete by validating it against the source system.
2. Plan for downtime: The historical load may require downtime for the source or destination system. Plan for this in advance to minimize disruption to business operations.
3. Use staging tables: Use staging tables to temporarily store the data before it is loaded into the destination system. This allows

for data validation and cleansing to be performed before the data is moved into the final destination.

4. Monitor performance: Monitor the performance of the load process to ensure that it completes within the required timeframe.

5. Plan for data dependencies: Ensure that any data dependencies, such as foreign keys or constraints, are maintained during the load process.

In summary, historical loads are used when moving data from one system to another or when setting up a new system. They involve extracting all the data from the source system and loading it into the destination system. Historical loads can be complex and time-consuming, and following best practices can help ensure a successful load.

2. What is Full Load?

A full load is a type of data loading technique that involves loading all the data from a source system into a destination system. It is typically used when setting up a new system or when there is a need to refresh all the data in the destination system. Here's a more detailed explanation of when and how full loads are performed:

When to Use Full Load:

- Initial data loads: When setting up a new system or database, a full load is typically used to move all the data from the source system to the destination system.
- Periodic refresh: Full loads are sometimes used to refresh all the data in the destination system periodically, to ensure that the data is up-to-date with the source system.
- Recovery: In some cases, a full load may be necessary to recover from a system failure or other data loss event.

How Full Load is Performed: Performing a full load involves extracting all the data from the source system, transforming it as necessary, and loading it into the destination system. The exact process will vary depending on the specific systems involved, but generally involves the following steps:

1. Extract: Data is extracted from the source system, typically using an extract-transform-load (ETL) tool or script. The data may be extracted directly from a database, a file system, or an application programming interface (API).
2. Transform: The extracted data is transformed as necessary to ensure that it can be loaded into the destination system. This

may involve converting data types, cleaning data, and performing any necessary calculations or aggregations.

3. Load: The transformed data is loaded into the destination system, typically using a database or data warehouse. The load may be performed in batch mode or in real-time, depending on the specific requirements of the project.

Full loads can be time-consuming and resource-intensive, especially when dealing with large datasets. To minimize the impact on the source and destination systems, it's important to optimize the extraction, transformation, and loading processes, and to schedule full loads during periods of low system activity.

3. What is Incremental Load?

Incremental load is a data loading technique in which only the new or changed data is loaded from the source system to the destination system. It is typically used when there are frequent updates to the source system and it's not necessary to reload all the data each time. Incremental load is also useful when the data is too large to load in a single batch or when real-time data is needed in the destination system.

Here's how incremental load is performed:

1. Identify the changed data: The first step in incremental load is to identify the new or changed data in the source system. This can be done by comparing the source and destination data, or by using a timestamp or a unique identifier to identify the new or changed records.
2. Extract the changed data: Once the new or changed data is identified, it needs to be extracted from the source system. This can be done using SQL queries or APIs depending on the source system.
3. Transform the data: The extracted data may need to be transformed to match the format and structure of the destination system. This can include data cleansing, data mapping, and data conversion.
4. Load the data: Once the data is transformed, it can be loaded into the destination system. This can be done using SQL insert statements, APIs or file uploads.
5. Update metadata: Finally, it's important to update the metadata that tracks the last load date and time, as well as any other relevant information about the load process.

Incremental load is an efficient way to keep the destination system up-to-date with the source system, without having to reload all the data each time. It can be scheduled to run at regular intervals, such

as every hour or every day, depending on the frequency of updates in the source system.