Soha Khalid

BWT – Data Engineering

Task 07

ELT vs ETL

ETL (Extract, Transform, Load):

Process: Data is extracted from various sources, then transformed (cleaned, normalized, aggregated) according to business requirements, and finally loaded into a data warehouse or data mart.

Usage: Typically used when transformations are complex and resource-intensive. Transformation is done in a dedicated ETL server or ETL tool before loading into the target system.

Tools: Informatica PowerCenter, IBM DataStage, Talend.

ELT (Extract, Load, Transform):

Process: Data is extracted from source systems and loaded into the target system (usually a data warehouse or data lake) without significant transformation upfront. Transformation occurs directly within the target system using SQL queries, stored procedures, or data processing frameworks.

Usage: Suitable when the target system (like a data warehouse) has powerful processing capabilities, such as MPP (Massively Parallel Processing) databases or big data platforms. It leverages the computing power of the target system for transformations.

Tools: Amazon Redshift, Google BigQuery, Snowflake.

When to Use Which One:

Use ETL when:
- Transformations are complex and require significant processing.
- Source data needs substantial cleaning, normalization, or aggregation before loading into the target system.
- The target system does not have robust processing capabilities for complex transformations.

Use ELT when:

- Source data does not require extensive transformation and can be directly loaded into the target system.
- The target system (data warehouse or data lake) has powerful processing capabilities to handle transformations efficiently.
- Real-time or near-real-time data processing is required.

Batch Processing:

Processing Model: Data is collected over a period of time and processed in large batches.
Latency: Higher latency as data is processed in chunks or batches, typically scheduled at regular intervals (e.g., hourly, daily).
Use Cases: Suitable for scenarios where latency is acceptable (e.g., daily reports, periodic data updates).

Stream Processing:

Processing Model: Data is processed as soon as it arrives, in real-time or near-real-time.
Latency: Lower latency as data is processed continuously or in small chunks as it streams in.
Use Cases: Suitable for real-time analytics, monitoring, fraud detection, IoT applications.

Demonstration with Use-Case:

Financial Transactions Processing

Batch Processing Scenario:

Description: A financial institution needs to generate daily transaction reports for regulatory compliance and business analysis.
Pipeline Design: Data is extracted from transaction databases every night, transformed to calculate daily summaries and metrics (such as total transactions, average transaction amount), and loaded into a data warehouse.
Tools: ETL tools like Informatica PowerCenter or Talend can be used to schedule and execute batch processing jobs.

Streaming Processing Scenario:

Description: The same financial institution also needs real-time fraud detection on transactions to prevent financial losses.
Pipeline Design: Transaction data is streamed in real-time from transaction systems. As each transaction arrives, it is processed through a streaming pipeline that applies business rules and statistical models for fraud detection.
Tools: Streaming frameworks like Apache Kafka with Apache Flink or Apache Spark Streaming can be utilized for building real-time processing pipelines.

Choosing ELT over ETL:

Scalability: ELT leverages the scalability of modern data warehouses or big data platforms, allowing for efficient processing of large volumes of data without upfront transformation bottlenecks.
Flexibility: It enables a more agile approach where raw data can be stored and transformed on-demand, supporting both traditional BI reporting and ad-hoc querying needs.
Cost Efficiency: ELT reduces the need for a separate ETL infrastructure, utilizing existing processing capabilities of the target data warehouse or lake, potentially lowering

overall infrastructure costs.

Choosing Streaming over Batch:

Real-Time Insights: Streaming processing provides immediate insights into data as it arrives, crucial for applications like fraud detection or IoT monitoring where timely actions are required.
Operational Efficiency: It reduces the need to wait for batch processing intervals, enabling quicker decision-making and response to events.
Continuous Processing: Streaming pipelines handle data continuously, which is essential for applications needing up-to-date information.

Conclusion:

In summary, for modern data processing needs, ELT is favored over ETL due to its scalability, flexibility, and cost-efficiency when leveraging powerful target systems. Similarly, streaming pipelines are preferred over batch for applications requiring real-time or near-real-time data processing capabilities, ensuring timely insights and operational efficiency.