ETL/ELT

O objetivo de realizar ETL (Extract, Transform, Load) ou ELT (Extract, Load, Transform) é integrar e preparar dados provenientes de diversas fontes, garantindo que estejam organizados, limpos e acessíveis para análises e tomada de decisão. Esses processos permitem centralizar informações em um único repositório, como um Data Warehouse ou Data Lake, melhorando a qualidade e consistência dos dados, além de otimizar a performance de consultas e análises. Com isso, empresas podem consolidar suas informações e apoiar estratégias baseadas em dados, criando pipelines escaláveis que atendam às demandas crescentes de volume e diversidade de fontes.

ETL (Extract, Transform, Load)

1. Extração (Extract)

A extração é o primeiro passo do ETL e tem como objetivo coletar dados brutos de diversas fontes. Essas fontes podem ser sistemas transacionais, bancos de dados, APIs, arquivos, ou qualquer outro local onde os dados estejam armazenados.

Identificação das fontes:

- Você precisa saber de onde os dados virão, como: sistemas ERP (ex.: SAP), bancos de dados transacionais (ex.: MySQL, PostgreSQL), arquivos CSV ou APIs.

2. Transformação (Transform)

Depois que os dados são extraídos, eles são preparados e **transformados** para atender aos requisitos da análise ou do sistema de destino. Essa etapa é onde acontece a maior parte do trabalho de "limpeza" e adaptação dos dados.

3. Carga (Load)

Após os dados estarem preparados, a última etapa é transferi-los para o sistema de destino, como um **Data Warehouse** (Snowflake, BigQuery, Redshift) ou arquiteturas mais modernas.

ELT (Extract, Load, Transform)

No processo ELT (Extract, Load, Transform), a ordem das etapas é alterada em relação ao ETL. Aqui, os dados brutos são extraídos e carregados diretamente no sistema de armazenamento de dados, onde a transformação acontece posteriormente. Essa abordagem é ideal para ambientes modernos com grande capacidade de processamento, especialmente em nuvem.

1. Extração (Extract)

Assim como no ETL, a extração no ELT envolve coletar dados brutos de várias fontes, como bancos de dados, APIs, arquivos CSV, sistemas ERP, entre outros. O objetivo é extrair rapidamente grandes volumes de dados para serem processados posteriormente no destino. Não há limpeza nem transformação nesta etapa, garantindo que os dados sejam transferidos na íntegra.

2. Carga (Load)

No ELT, os dados são carregados no sistema de destino logo após a extração, sem nenhuma transformação prévia. O destino geralmente é um **Data Lake** ou um **Data Warehouse em nuvem**, como Snowflake, Google BigQuery, ou Amazon Redshift.

3. Transformação (Transform)

Depois que os dados estão no destino, começa a transformação, que geralmente é feita diretamente no sistema de armazenamento usando SQL ou ferramentas especializadas, como dbt (Data Build Tool). Isso elimina a necessidade de transformar os dados antes de carregá-los, permitindo que os sistemas modernos lidem com grandes volumes de dados.

Quando usar ETL ou ELT?

A escolha entre ETL (Extract, Transform, Load) e ELT (Extract, Load, Transform) depende de vários fatores, incluindo o tipo de dados, o ambiente tecnológico, os requisitos do projeto e as preferências da equipe.

O ETL é mais adequado em cenários onde os dados precisam ser transformados antes de serem carregados no sistema de destino. Ele é uma escolha tradicional em ambientes locais (on-premise) e para sistemas que não suportam grandes volumes de dados brutos. Exemplos: Ambientes legados que utilizam Data Warehouses tradicionais, como Teradata ou Oracle, onde o processamento de dados brutos pode ser caro e demorado no destino. Quando há cenários com requisitos de qualidade imediata, onde dados limpos e estruturados devem estar disponíveis imediatamente após o carregamento, como em relatórios diários ou sistemas que não podem processar dados brutos, etc.

Já o ELT é ideal para ambientes modernos baseados em nuvem, onde a escalabilidade e a capacidade de processamento distribuído são fundamentais. Ele é mais flexível e eficiente para grandes volumes de dados e processos iterativos. O ELT é indicado para lidar fluxos de dados em tempo real, em que mover os dados brutos para o destino é mais eficiente antes de realizar as transformações necessárias. Além disso, ao reduzir a complexidade inicial do processo e permitir o carregamento de dados brutos diretamente no destino, o ELT ajuda a reduzir custos, tornando-o uma escolha eficiente para projetos em que as transformações podem ser adiadas para o ambiente de destino, garantindo maior agilidade e capacidade de adaptação às necessidades analíticas.

Fonte de dados

As fontes de dados são os sistemas, plataformas ou dispositivos que geram ou armazenam informações que podem ser utilizadas para análise e tomada de decisões. Essas fontes podem ser de diversos tipos e formatos, dependendo do contexto organizacional e das necessidades do projeto. Alguns exemplos:

- Os **bancos de dados** são uma das fontes mais comuns de dados. Eles podem ser relacionais ou não relacionais. Ex.: MySQL, PostgreSQL, MongoDB.

- **Sistemas de CRM (Customer Relationship Management)** é uma plataforma usada para gerenciar interações com clientes e leads, oferecendo dados sobre comportamento, vendas e suporte. Ex.: Salesforce, Pipedrive
- **ERP (Enterprise Resource Planning)** integram os processos de negócios em uma única plataforma, oferecendo acesso aos dados de estoques, finanças, logística, etc. Ex.: SAP, Oracle ERP.

APIs (Application Programming Interfaces)

API é a sigla para **Application Programming Interface** (Interface de Programação de Aplicações). Trata-se de um conjunto de regras e protocolos que permite que diferentes softwares se comuniquem entre si, compartilhando funcionalidades e dados. **As APIs funcionam como uma ponte entre sistemas, facilitando a integração e a troca de informações**. Sua utilização facilita a forma que sistemas compartilham dados entre si e boa parte dos sites, aplicativos ou sistemas usam esse recurso.

Um tipo muito conhecido de API são as chamadas **REST API** (Representational State API). Elas fornecem uma forma padronizada de fazer as principais operações entre sistemas. São elas:

Tipos de requisição:

- **GET**: Utilizado quando o requisitante solicita dados do servidor.
- **POST**: Utilizado quando o requisitante envia/cria dados no servidor.
- **PUT**: Utilizado quando o requisitante atualiza dados no servidor.
- **DELETE**: Utilizado quando o requisitante remove dados do servidor.

Geralmente o dado recebido ou postado segue a estrutura de arquivos **JSON**.

Estrutura de uma chamada de API:

1. **Header**: Utilizado para passar informações gerais da requisição, como por exemplo as credenciais de acesso (segurança).
2. **Tipos de Requisição**: Informa se vamos obter dados (GET), enviar dados (POST) e assim por diante.
3. **End Point**: Tipo de dado que o requisitante busca.
4. **Parameters/Body**: São os parâmetros que cada End Point necessita para filtrar as informações buscadas. Dependendo do End Point, nenhum parâmetro é necessário.