

# Procesos de ETL Modernos con Python

Domina el Flujo de Datos 🦹



# Hola soy... 🙋

¡Hola a todos! Antes de sumergirnos en esta presentación, quiero tomarme un momento para hablar un poco sobre mí. **Flavio Cesar Sandoval**, he trabajado como Data Analyst, Marketing Science Analyst y Data Engineer.

Soy profe en los bootcamps de ciencia e ingeniería de datos.

#DataViz

#SQL

#DataEngineer



# ! Domina tus procesos de Datos! 🐼🚀

En este curso, no solo aprenderás a manejar datos, sino a transformarlos en el combustible que impulsa las decisiones de negocio. Domina el arte de la extracción, transformación y carga (ETL) con Python y herramientas de vanguardia como Pandas, Polars y Mage-AI.



# Estructura del curso

**/01**

## Fundamentos de ETL

Principios clave de la metodología ETL, su arquitectura y flujo de trabajo

**/02**

## Configuración del entorno

Instalación y configuración de las herramientas necesarias para ETL

**/03**

## (E) Extract

Obtener datos desde diversas fuentes

**/04**

## (T) Transform

Limpiar, transformar y enriquecer datos

**/05**

## (L) Load

Obtener datos desde diversas fuentes

**/06**

## Herramientas avanzadas

Mage-AI, herramienta de código abierto para la creación y gestión de pipelines de ETL

**/07**

## Proyecto final

Aplicarás los conocimientos adquiridos a través de un proyecto final desafiante

# Clase 1

## Introducción a ETL

# Introducción a ETL

¿Tus datos son un caos? 🤯 ¡Tranquilo, tendrás la solución!



# ¿Qué es ETL?

## ¡La magia 🧙 de la transformación de datos!

**Extracción (E):** Conectamos con tus datos, sin importar dónde se escondan (bases de datos, archivos, APIs...).

**Transformación (T):** Limpiamos, organizamos y damos forma a tus datos, como un escultor que crea una obra maestra.

**Carga (L):** Entregamos los datos procesados a su destino final, listos para ser analizados y generar insights poderosos.

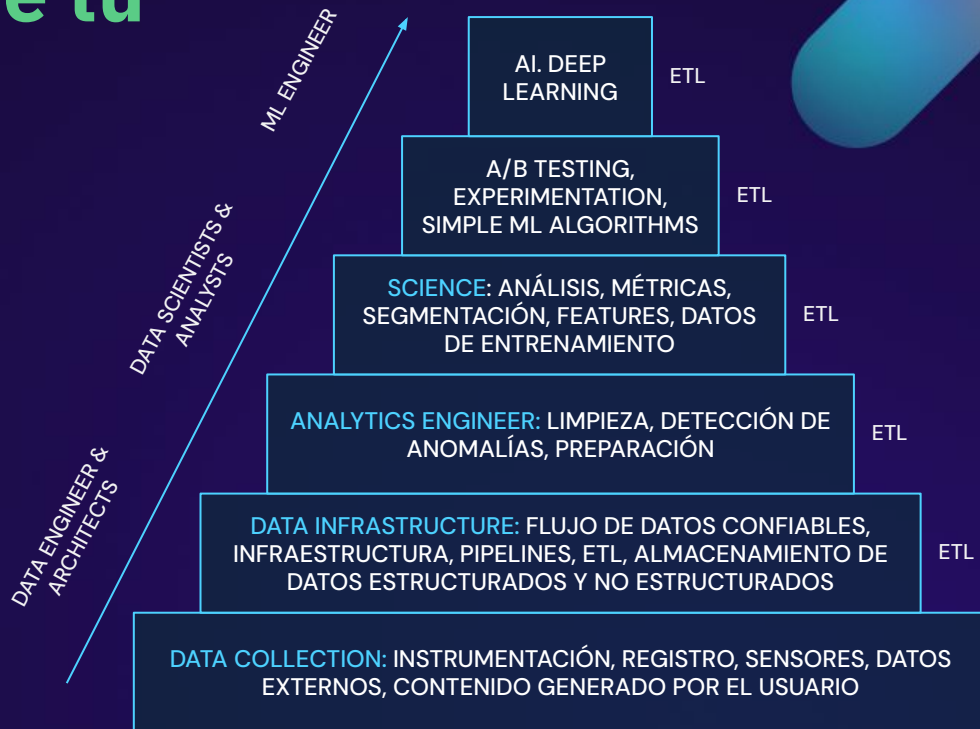


# ¿Por qué ETL es importante? ¡El combustible de tu empresa!

ETL es la base de la inteligencia de negocio. Sin datos limpios y estructurados, no puedes tomar decisiones informadas.

- **Descubre patrones ocultos:** ETL te permite encontrar relaciones y tendencias en tus datos que de otra manera pasarían desapercibidas.
- **Optimiza tus procesos:** Identifica cuellos de botella, ineficiencias y áreas de mejora gracias al análisis de datos.

¡ETL es la clave para impulsar el crecimiento y la innovación en tu empresa!





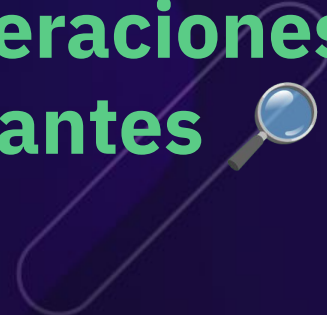
# ETL en diferentes industrias

	Extract	Transform	Load
Finanzas	Sistemas transaccionales bancarios, plataformas de inversión, registros contables, informes de mercado, fuentes de datos externos (índices bursátiles, tasas de interés).	Limpieza de datos (eliminación de duplicados, valores atípicos), enriquecimiento de datos (agregar información de clientes), agregación de datos (cálculo de saldos, rendimiento de carteras), anonimización de datos (para cumplir con regulaciones de privacidad).	Almacenes de datos financieros (Data Warehouses), lagos de datos (Data Lakes), bases de datos relacionales, herramientas de visualización de datos (Tableau, Power BI) para análisis de riesgos, detección de fraudes, informes financieros.
Salud	Historias clínicas electrónicas (EHR), sistemas de facturación médica, dispositivos médicos (monitores de pacientes, wearables), registros de ensayos clínicos, bases de datos de salud pública.	Estandarización de terminología médica (códigos ICD-10, SNOMED CT), conversión de formatos de datos, anonimización de datos de pacientes (eliminación de información de identificación personal - PII), integración de datos de múltiples fuentes.	Almacenes de datos clínicos, registros de salud basados en la población, herramientas de análisis de datos de salud, sistemas de apoyo a la toma de decisiones clínicas, paneles de control de salud pública.

# ETL en diferentes industrias

	Extract	Transform	Load
Marketing	Plataformas de redes sociales (Facebook, Twitter, Instagram), herramientas de seguimiento web (Google Analytics), sistemas de gestión de relaciones con clientes (CRM), plataformas de publicidad online, encuestas a clientes.	Limpieza de datos de redes sociales, seguimiento de campañas publicitarias, segmentación de clientes, cálculo de métricas de marketing (ROI, tasa de conversión, tasa de apertura de correo electrónico)	Bases de datos de marketing, plataformas de automatización de marketing, herramientas de visualización de datos de marketing, paneles de control de rendimiento de campañas.
Ecommerce	Plataformas de comercio electrónico (Shopify, Magento, WooCommerce), sistemas de gestión de pedidos, sistemas de pago, registros de navegación web de los clientes, datos de productos.	Seguimiento del comportamiento de los clientes en el sitio web, análisis de la cesta de la compra, cálculo de métricas de comercio electrónico (valor medio de los pedidos, tasa de abandono del carrito), personalización de recomendaciones de productos.	Almacenes de datos de comercio electrónico, motores de recomendación de productos, herramientas de análisis de comercio electrónico, paneles de control de rendimiento de ventas.

# ETL se utiliza en prácticamente todos los sectores y debes de tener en cuenta algunas consideraciones importantes



- **Privacidad de Datos:** En todas las industrias, es crucial garantizar el cumplimiento de las regulaciones de privacidad de datos (como el GDPR en Europa o la LGPD en México). La anonimización y el cifrado de datos son prácticas esenciales.
- **Calidad de los Datos:** La calidad de los datos es fundamental para obtener información precisa. La limpieza, validación y enriquecimiento de datos son pasos clave en el proceso de transformación.
- **Escalabilidad:** A medida que las empresas crecen, sus necesidades de datos también lo hacen. Es importante diseñar soluciones ETL que puedan escalar para manejar mayores volúmenes de datos.

# Reto fin de clase

**Piensa en como sería el flujo de ETL en las dos industrias que más te interesen y genera la tabla resumen de los procesos**

# Clase 2

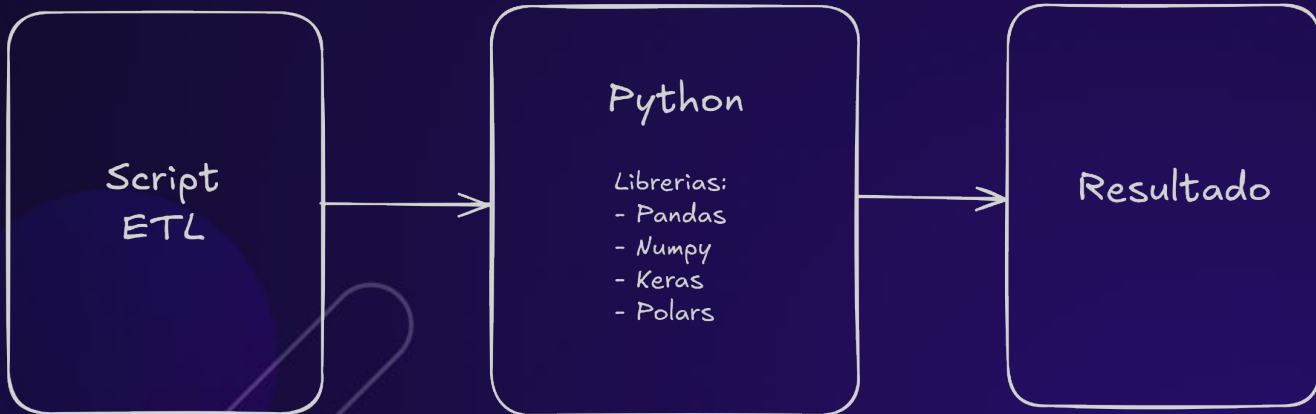
## Configuración del entorno

# ¡Crea el entorno perfecto!

¡Olvídate de errores frustrantes 🤬 y configuraciones interminables! Vamos a construir un entorno perfecto para ETL 😎



## Ambiente de desarrollo



## Ambiente de desarrollo



# Anaconda 🐍 ¡Tu kit de herramientas!

## ANACONDA DISTRIBUTION

Most Trusted Distribution for Data Science

### ANACONDA NAVIGATOR

Desktop Portal to Data Science

### ANACONDA PROJECT

Portable Data Science Encapsulation

### DATA SCIENCE LIBRARIES

#### Data Science IDEs



#### Analytics & Scientific Computing



#### Visualization



#### Machine Learning



...and many more!

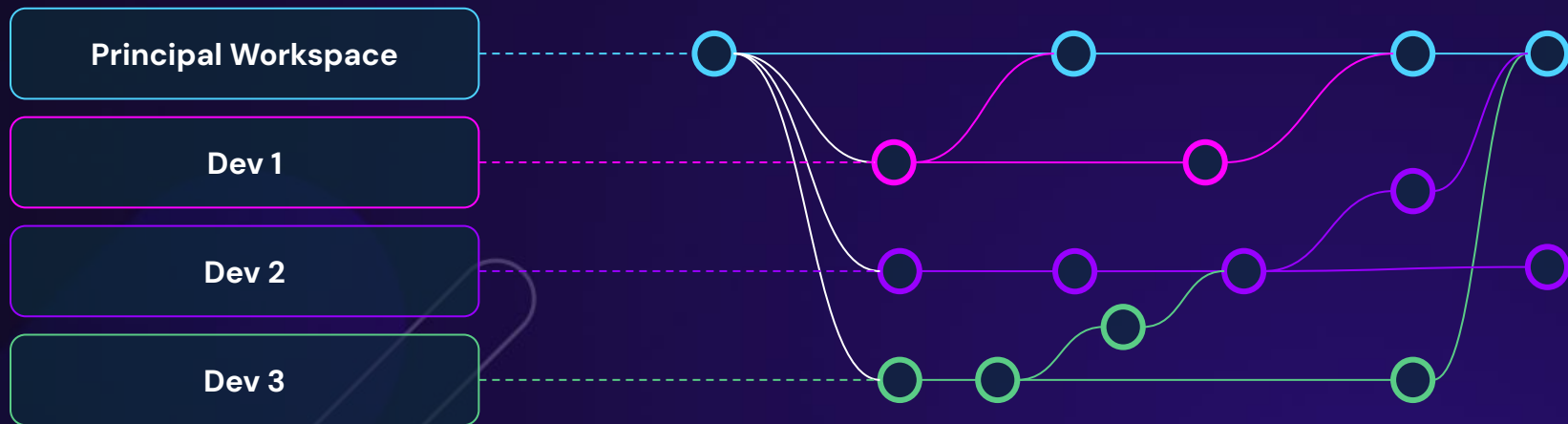
### CONDA®


Data Science Package & Environment Manager


## Pipeline ETL



# Git/GitHub: ¡Estandarización de procesos para todo el equipo!



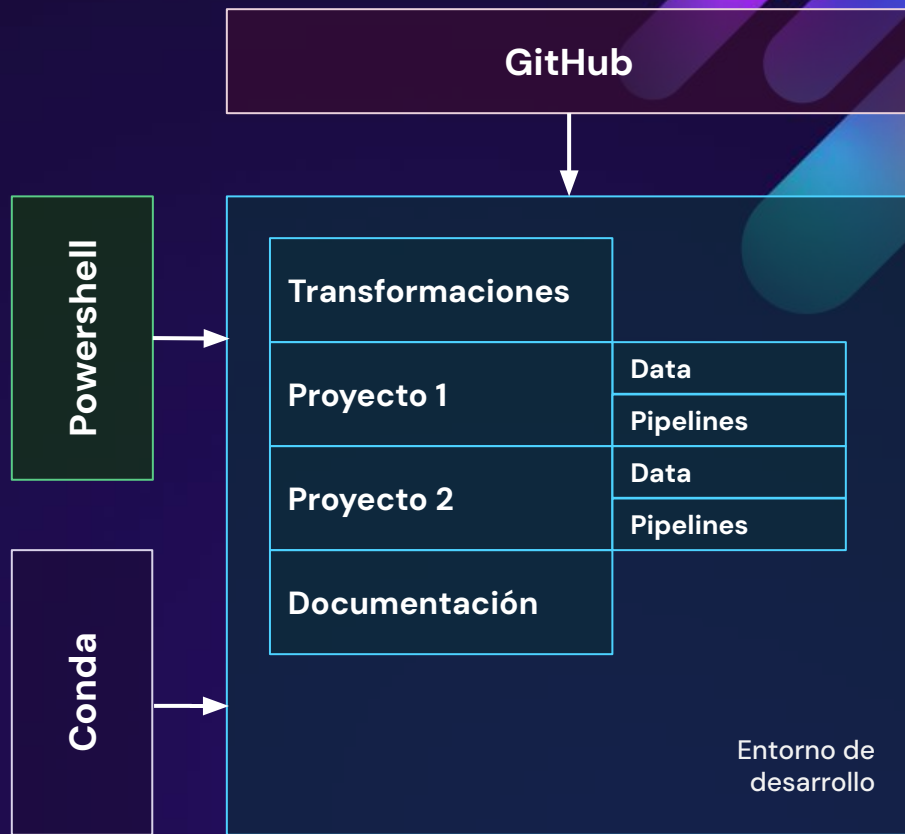
**PowerShell**   
**¡La línea de  
comandos es  
tu aliada!**



```
pwsh in Teaching
@flavi on ~\Teaching
# echo "Hola Codigo Facilito"
Hola Codigo Facilito
@flavi on ~\Teaching
#
```



**Un buen entorno de desarrollo:** te da las herramientas y el orden necesarios para experimentar y crear 🧪.



# Reto fin de clase

**Genera un entorno de desarrollo que se adapte a tus necesidades**

# Clase 3

## Principios de ETL

# Principios de ETL

¿Quieres construir un castillo de datos que dure para siempre?



¡Necesitas una base sólida!

# Arquitectura de un pipeline ETL

¡El mapa  de tu viaje de datos!

Un pipeline de datos construido con una arquitectura sólida es fácilmente mantenible y escalable, pero un pipeline de datos débil corre un alto riesgo de falla, ya sea estructural o analíticamente, produciendo un producto inexacto.

Los siguientes son los atributos de una canalización de datos sólida:

- Expectativas claramente definidas
- Arquitectura escalable
- Reproducible y clara

**Una investigación inicial te ayudará a construir arquitecturas sólidas**

¿De dónde vienen los datos?

- Bases de datos, APIs, archivos, etc.
- Cuáles son los tipos de datos
- Qué datos deben recopilarse
- Cuando los puedo obtener



¿Qué transformación, limpieza, filtrado, agregación deben de aplicarse a los datos?

- Existen inconsistencias en los datos
- Necesitas aplicar estandarizaciones
- Cuáles el stack disponible



¿Dónde deben de entregarse los datos?,

- Cuál es el formato de la entrega
- Cuál es la frecuencia de actualización
- El usuario final puede acceder a todos los datos obtenidos



## Architecture Roadmap

### Sources:

- Reporte Semanal:
- Facebook Ads (API)
- YouTube Ads (API)

### Reporte Mensual:

- Excel Planificación

### Dimensiones:

- Fecha (Facebook, Youtube)
- Nombre de la campaña (Facebook, Youtube, Excel)
- Región (Facebook, Youtube)

### Métricas:

- Clicks (Facebook, Youtube, Excel)
- Impresiones (Facebook, Youtube, Excel)
- Visualizaciones (Facebook, Youtube, Excel)
- Costo (Facebook, Youtube, Excel)

### Transformaciones:

- Las Regiones no están agrupadas en Youtube
- El costo debe de presentarse en dólares
- Cuáles el costo de obtener 1000 impresiones
- Cuál es el costo que tenemos por click
- Cuál es el costo por visualización

### Servicio:

- Una tabla para conectarse desde power BI
- La actualización será de manera mensual

### Stack Tecnológico:

- Python
- PostgreSQL



**Sources:**

- Reporte Semanal:
  - Facebook Ads (API)
  - YouTube Ads (API)

**Reporte Mensual:**

- Excel Planificación

**Dimensiones:**

- Fecha (Facebook, Youtube)
- Nombre de la campaña (Facebook, Youtube, Excel)
- Región (Facebook, Youtube)

**Métricas:**

- Clicks (Facebook, Youtube, Excel)
- Impresiones (Facebook, Youtube, Excel)
- Visualizaciones (Facebook, Youtube, Excel)
- Costo (Facebook, Youtube, Excel)

**Transformaciones:**

- Las Regiones no están agrupadas en Youtube
- El costo debe de presentarse en dólares
- Cuáles el costo de obtener 1000 impresiones
- Cuál es el costo que tenemos por click
- Cuál es el costo por visualización

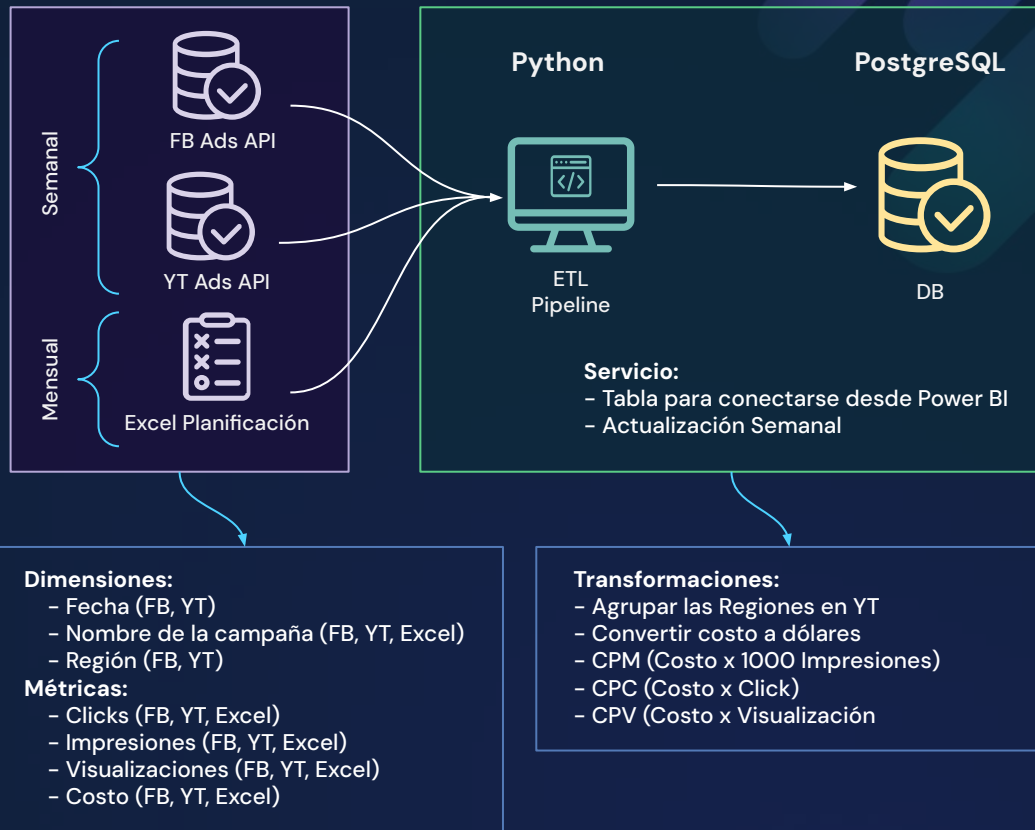
**Servicio:**

- Una tabla para conectarse desde power BI
- La actualización será de manera mensual

**Stack Tecnológico:**

- Python
- PostgreSQL

## Architecture



**Excelente, ya tienes un  
sistema de ETL completo**

¿Esto es todo  ?

**Quizás te preguntes si  
consideramos todas las  
opciones ...**



# Pongamos a prueba nuestro diseño de ETL

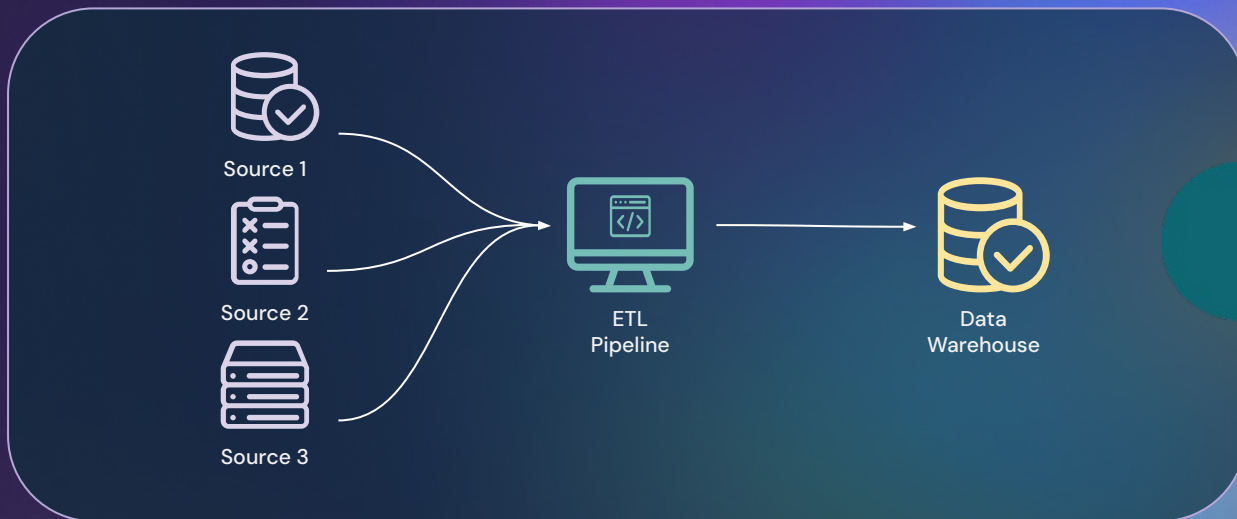


# 1 Patrón de diseño ETL básico

¿Qué sucede cuando ocurre un problema de red en mitad de la operación?

¿Qué sucede con los datos en medio del proceso?

¿Se conserva algún dato?





## 2 Patrón de diseño ETL-P

¿Qué sucede si la conexión a la fuente de datos es inconsistente o tiene una frecuencia diferente a la necesaria para las necesidades comerciales de ingesta de datos?

La capa de PSA por sí sola no puede satisfacer esta necesidad



Source 1



Source 2



Source 3



ETL  
Pipeline



PSA  
Área de preparación  
persistente



ETL  
Pipeline

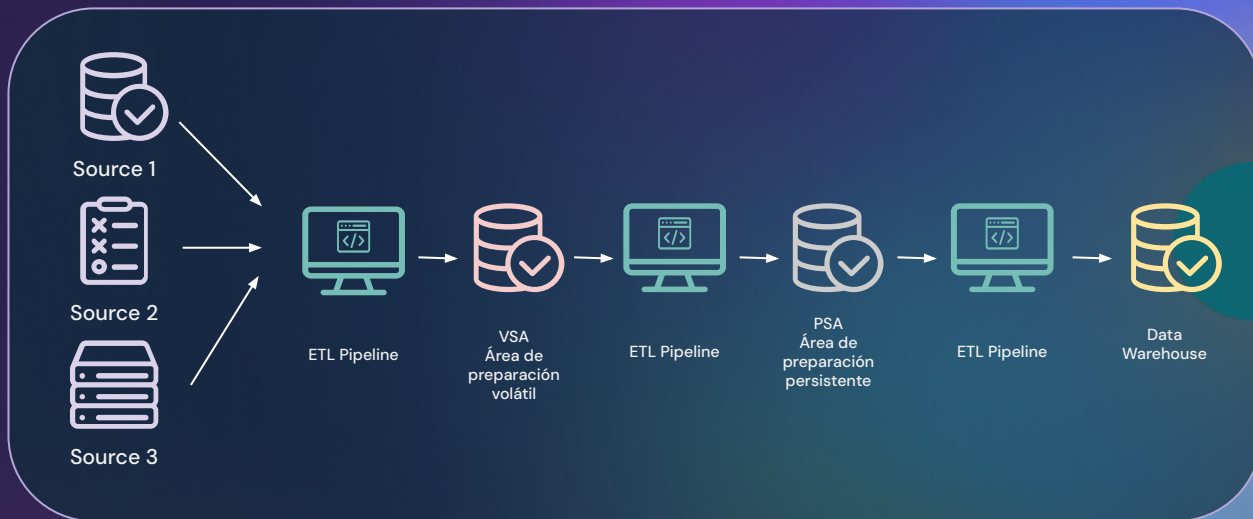


Data  
Warehouse

La capa PSA divide las etapas de importación y exportación de datos para limitar la responsabilidad de los problemas de movimiento de datos derivados de la conexión entre las ubicaciones de datos de origen y destino que se cortan.

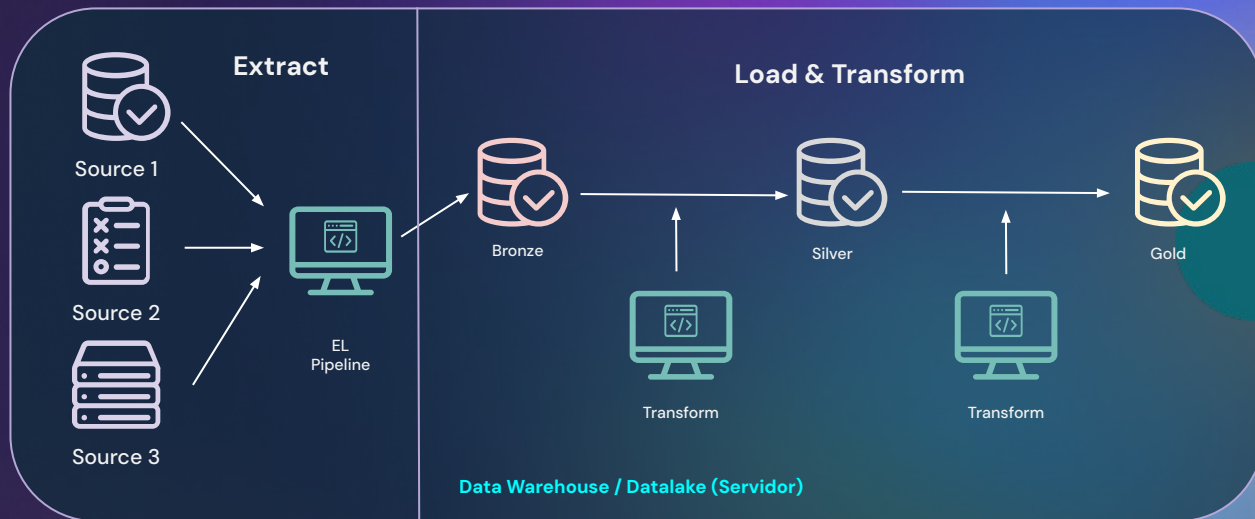
## 3 Patrón de diseño ETL-VP

El ETL-VP puede manejar procesos de importación de datos asincrónicos con facilidad debido a su capacidad de utilizar la capa VSA para cargar datos por lotes en la capa PSA en un cronograma predecible y consistente.



## 4 Patrón de diseño ELT

Resulta que dividir la recopilación de fuentes y la carga de datos en pasos separados es una forma útil de reducir algunos de estos costos evitables. Crear una canalización de datos de esta manera brinda más control sobre la canalización y hace que cada proceso sea más dinámico y reutilizable.



# ELT vs ETL



# Clase 4

## Planificación de pipelines de ETL

# Planificación de pipelines

¿Qué pasa después de crear la arquitectura de tu ETL 🤔?

# Diseño de Pipelines ETL Eficientes



## ¡El arte de la ingeniería de datos!

Construir un pipeline ETL no es una tarea trivial. Es comparable a ensamblar un complejo rompecabezas, donde cada pieza debe encajar a la perfección. Requiere planificación meticulosa, estrategia sólida y las herramientas adecuadas.

### Modularidad:

Dividir el proceso en etapas claras y manejables (extracción, transformación, carga) para facilitar el mantenimiento y la escalabilidad.

### Paralelización:

Identificar oportunidades para ejecutar tareas en paralelo y aprovechar al máximo los recursos computacionales disponibles.

### Tolerancia a fallos:

Implementar mecanismos para manejar errores y recuperarse de interrupciones sin perder datos ni comprometer la integridad del sistema.


### Monitoreo y registro:

Establecer un sistema de monitoreo y registro para rastrear el progreso del pipeline, identificar cuellos de botella y solucionar problemas de manera proactiva.



# Identificación de fuentes y destinos de datos

**Identifica claramente los puntos de partida y llegada** 

Es fundamental comprender los requerimientos, la estructura y el formato de los datos en cada fuente y destino para garantizar una integración fluida y evitar sorpresas desagradables durante el proceso de ETL. 

## Fuentes de datos:

- Accesibilidad a los datos
- Dependencias de límite y frecuencia
- Generación de llaves de acceso
- Características peculiares de la fuente de datos
- Documentación de conexiones

## Destinos de datos:

- Acceso a la base de datos
- Recursos de la fuente de destino
- Esquemas de desarrollo
- Requisitos de gobernanza
- Acceso para los usuarios finales
- Documentación requerida
- Procesos de pase a producción



# Clase 5

## Librerías para ETL



# Librerías para ETL

Explora, experimenta y descubre las herramientas que mejor se adapten a tus necesidades

# Pandas - Polars: Tus aliados en la manipulación de datos

**Pandas:** La herramienta ideal para la manipulación de datos.

- Lectura y escritura de archivos (CSV, Excel, JSON)
- Limpieza y transformación de datos
- Análisis exploratorio de datos (estadísticas, visualizaciones)

**Polars:** El bolido de la velocidad 🚀 para Big Data.

- Diseñado para aprovechar al máximo el hardware moderno.
- Ideal para trabajar con grandes volúmenes de datos.
- Sintaxis similar a Pandas, pero con un rendimiento superior.

```
import pandas as pd

df_csv = pd.read_csv('tu_archivo.csv')
df_filtrado = df_csv[df_csv['Edad'] > 30]
df_agregado = df_csv.groupby('Ciudad')['Ventas'].sum()
df_csv['Edad'].plot(kind='hist')
```



Flavio Cesar Sandoval Muñoz

DSandovalFlavio

```
import polars as pl

df_csv = pl.read_csv('tu_archivo.csv')
df_filtrado = df_csv.filter(pl.col('Edad') > 30)
df_agregado = df_csv.groupby('Ciudad').agg(
    pl.col('Ventas').sum().alias('SumaVentas'))
```



Flavio Cesar Sandoval Muñoz

DSandovalFlavio

# SQLAlchemy y ODBC: El puente entre Python y tus bases de datos

**SQLAlchemy:** ORM (Object-Relational Mapper) más popular.

- Abstrae las diferencias entre diferentes sistemas de bases de datos.
- Permite realizar consultas SQL complejas de forma sencilla y segura.

**ODBC:** El estándar de conexión a bases de datos.

- Open Database Connectivity (ODBC) es una interfaz universal para acceder a diferentes bases de datos.
- SQLAlchemy utiliza ODBC como capa de abstracción para comunicarse con los sistemas de bases de datos.

```
import pandas as pd
import pyodbc
from sqlalchemy import create_engine

# Configuración de la conexión ODBC
connection_string = (
    "DRIVER={ODBC Driver 17 for SQL Server};"
    "SERVER=tu_servidor;"
    "DATABASE=tu_base_de_datos;"
    "UID=tu_usuario;"
    "PWD=tu_contraseña"
)

# Crear el motor de SQLAlchemy utilizando pyodbc
engine = create_engine(
    'mssql+pyodbc:///?odbc_connect={}'.format(
        pyodbc.connect(connection_string)))

query = "SELECT * FROM clientes"
df_clientes = pd.read_sql(query, engine)
nuevos_clientes.to_sql(
    'clientes_nuevos',
    engine,
    if_exists='replace',
    index=False)
```



Flavio Cesar Sandoval Muñoz

 DSandovalFlavio

# Clase 6

## Estrategias de extracción de datos

# Estrategias de extracción

**Nuestros datos pueden residir en una variedad de fuentes,  
cada una con sus propias peculiaridades y desafíos**

# Obtención de datos desde diferentes fuentes

## El mundo de los datos es diverso y complejo

No todos los datos son iguales. Algunos son ordenados y predecibles (estructurados), otros tienen cierta estructura, pero también flexibilidad (semi-estructurados), y otros son caóticos y difíciles de interpretar (no estructurados).

### Datos estructurados:

Son los más fáciles de manejar, ya que se ajustan a un esquema predefinido (tablas, columnas, tipos de datos). Podemos utilizar consultas SQL para extraerlos de manera eficiente, siempre que se pueda las transformaciones deben de realizarse en el servidor.


### Datos semi-estructurados:

Pueden tener variaciones en su estructura (por ejemplo, archivos JSON con campos opcionales). Necesitamos herramientas y técnicas que nos permitan manejar esta flexibilidad, Pandas tiene métodos que te permiten manejar este tipo de datos.

### Datos no estructurados:

Son el mayor desafío, ya que pueden ser texto libre, imágenes, audio o video. Aquí, necesitamos recurrir a técnicas de procesamiento de lenguaje natural (NLP), visión por computadora y aprendizaje automático para extraer información significativa.

# Manejo de grandes volúmenes de datos (Big Data)

En la era del Big Data, es común encontrarnos con conjuntos de datos masivos que superan la capacidad de procesamiento de herramientas tradicionales. 

## Escalabilidad horizontal:

En lugar de depender de un solo servidor potente, distribuimos la carga de trabajo entre varios servidores más pequeños, trabajando en paralelo para procesar grandes volúmenes de datos de manera eficiente.

## Procesamiento distribuido:

Utilizamos frameworks como Apache Hadoop y Apache Spark para dividir el procesamiento de datos en tareas más pequeñas que se ejecutan en un clúster de servidores, aprovechando la potencia de cómputo distribuida.

## Almacenamiento en la nube:

Aprovechamos servicios de almacenamiento en la nube como Amazon S3 y Google Cloud Storage para almacenar y acceder a grandes volúmenes de datos de manera escalable y rentable.



## **Clase 7, 8, 9, 10, 11**

Extracción de datos desde archivos

Extracción de datos desde APIs

Extracción de datos desde bases de datos

Extracción de datos desde almacenamiento  
en la nube

Estandarización de la extracción de datos

# Lectura y Extracción

**Nuestros datos pueden residir en una variedad de fuentes,  
cada una con sus propias peculiaridades y desafíos**

# Lectura de Datos Utilizando:

- Pandas 
- Polars 
- Airbyte 

## Extracción de datos desde archivos:

- Lectura y manipulación de archivos CSV, Excel y otros formatos comunes
- Uso de Pandas y Polars para la extracción eficiente de datos desde archivos

## Extracción de datos desde APIs:

- Conceptos básicos de APIs REST y consumo de APIs en Python
- Extracción de datos JSON desde APIs públicas y privadas

## Extracción de datos desde bases de datos:

- Conexión a bases de datos relacionales con ODBC y SQLAlchemy
- Ejecución de consultas SQL para extraer datos específicos
- Extracción de datos desde bases de datos NoSQL (MongoDB)

## Extracción de datos desde almacenamiento en la nube:

- Acceso y extracción de datos desde plataformas de almacenamiento en la nube como Amazon S3, Google Cloud Storage y Microsoft Azure Blob Storage

## Estandarización de la extracción de datos:

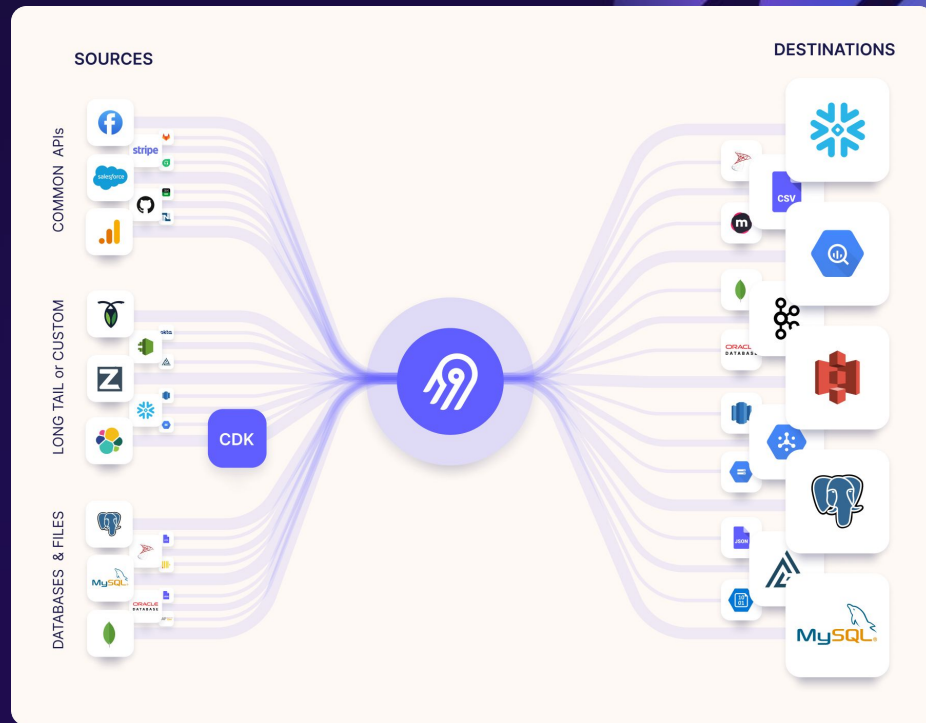
- Uso de Airbyte para automatizar la extracción de datos desde diversas fuentes
- Configuración de pipelines de extracción robustos y escalables

Airbyte 

# Automatiza tus extracciones de datos

**Airbyte:** La herramienta que simplifica la extracción de datos.

- Conecta con más de 150 fuentes de datos (bases de datos, APIs, CRMs, plataformas de marketing...).
- Normaliza los datos en un formato común para facilitar su procesamiento.
- Automatiza la extracción de datos de forma incremental o completa.
- ¡Olvídate de escribir conectores personalizados para cada fuente de datos!



# Clase 12, 13, 14

Técnicas de limpieza y transformación de  
datos Pandas

Transformaciones personalizadas con  
funciones y decoradores

Técnicas de limpieza y transformación de  
datos Polars

# Transformación de datos

¡El arte de esculpir la información!

# Transformación de Datos

Utilizando:

- Pandas 

- Polars 

## Técnicas de limpieza y transformación de datos:

### Pandas:

- Transformaciones para distintos tipos de datos
- Regex para limpieza de datos

### Polars:

- Transformaciones para distintos tipos de datos
- Regex para limpieza de datos

## Transformación de datos desde el servidor utilizando SQL

Como manejar transformaciones que no se pueden ejecutar desde tu PC

## Transformaciones personalizadas con funciones y decoradores:

- Creación de funciones reutilizables para realizar transformaciones específicas
- Aplicación de decoradores para encadenar y automatizar tareas de transformación

# Clase 15

## Estrategias de carga de datos



# Estrategias de carga de datos

¡Has extraído y transformado tus datos, ahora es el momento de llevarlos a su nuevo hogar! 🏠

# Optimización del proceso de carga para minimizar el tiempo y los recursos

La carga de datos puede ser un proceso costoso en términos de tiempo y recursos computacionales, especialmente cuando se trata de grandes volúmenes de información. ⌚

## Carga incremental:

En lugar de cargar todos los datos cada vez, solo cargamos los nuevos o modificados desde la última ejecución, reduciendo significativamente el tiempo y los recursos necesarios.

## Escalabilidad:

Debemos asegurarnos de que nuestro proceso de carga pueda manejar el crecimiento exponencial de los datos sin comprometer el rendimiento.

## Paralelización:

Aprovechamos la potencia de cómputo distribuida para ejecutar múltiples tareas de carga en paralelo, acelerando el proceso y reduciendo el tiempo total de carga.

# **Clase 16, 17, 18, 19**

Carga de datos con ODBC y Python

Carga de datos con SQLAlchemy y Pandas

Carga de datos con Polars

Carga de datos al almacenamiento en la  
nube

# Carga de Datos Utilizando:

- Pandas 
- Polars 

## Carga de datos con Python, ODBC y Pandas:

**Pandas:** Simplifica la carga a DB relacionales mediante `to_sql()`.

**SQLAlchemy:** Un ORM potente que ofrece un mayor control y flexibilidad en la carga de datos.

**PyODBC:** Combina la potencia de SQLAlchemy con la eficiencia de ODBC para optimizar aún más la carga de datos.

## Polars: Carga eficiente a Bases de Datos y Data Lakes:

### Eficiencia:

Su capacidad para manejar grandes volúmenes de datos y su enfoque en la eficiencia lo convierten en una opción ideal para Data Lakes.

### Facilidad:

Utiliza `pl.DataFrame.write_parquet()` o `pl.DataFrame.write_csv()` para exportar tus DataFrames de Polars a diferentes formatos y almacenarlos en tu Data Lake.

## Almacenamiento en la nube:

- **Almacenamiento en la nube:** flexibilidad, escalabilidad y acceso global a tus datos.
- **Amazon S3:** Utiliza `boto3` para cargar y descargar archivos a buckets de S3.
- **Google Cloud Storage:** Utiliza la biblioteca `google-cloud-storage` para gestionar tus datos en la nube de Google.

# Clase 20

## Introducción a Mage

# ETL Soluciones Avanzadas



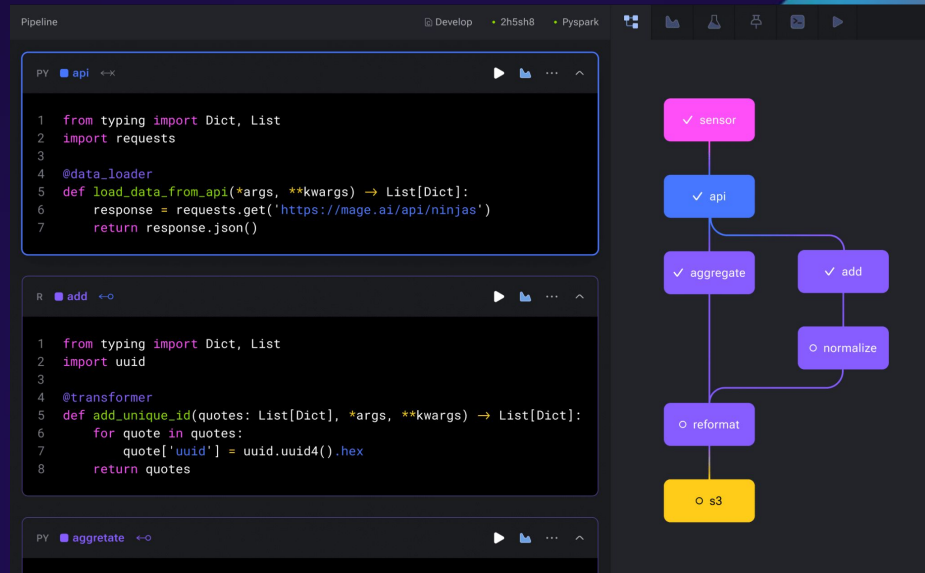
**Mage, la magia de la transformación de datos**

# Introducción a Mage y su aplicación en ETL

Mage es un framework híbrido que combina la flexibilidad de los notebooks con la robustez del código modular, ofreciendo una experiencia de desarrollo intuitiva y eficiente. ✨

Conceptos básicos:

- **Proyectos:** Organizan su código y pipelines, similar a repositorios en GitHub.
- **Pipelines:** Contienen bloques de código, visualizaciones y definen dependencias entre ellos.
- **Bloques:** Código (Python, SQL, R) ejecutables de forma independiente o dentro de un pipeline.
- **Triggers:** Definen cuándo y cómo se ejecuta un pipeline (programación, eventos, etc.).
- **Ejecuciones:** Almacenan información sobre cada ejecución de un pipeline o bloque.



# Clase 21, 22, 23

Extracción de datos con Mage

Transformación de datos con Mage

Carga de datos con Mage



# ETL Con Mage

## Extracción de datos con Mage:

**Conectores pre construidos:**

Conéctate a bases de datos populares, APIs, CRMs, plataformas de marketing y mucho más.

**Sincronización de datos:**

Mantén tus datos actualizados con extracciones incrementales o completas.

## Transformación de datos con Mage:

**Lenguaje de tu elección:**

Escribe código en Python, SQL o R, ¡o combínalos en el mismo pipeline!

**Modularidad:**

Cada paso de tu pipeline es un bloque de código reutilizable y testeable.

## Almacenamiento en la nube:

**Conectores pre construidos:**

Carga datos en tu Data Warehouse o Data Lake favorito con facilidad.

**Escalabilidad:**

Mage se integra con Spark para procesar y cargar grandes volúmenes de datos.

**Monitorización y alertas:**

Mantén el control de tus pipelines con un sistema de monitorización y alertas integrado.

# Clase 24

## Propuesta de proyecto

# Clase 25

## Desarrollo del proyecto

# Repositorio:

[DSandovalFlavio/Curso\\_Codigo-Facilito\\_Procesos-ETL-Modernos-Python: Curso Procesos de ETL Modernos con Python](#)