



INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
ACADEMIA INGENIERÍA DE SOFTWARE



Unidad de Aprendizaje: Big Data

Profesor: Ing. Tania Rodriguez Sarabia

Práctica: ELT

ALUMNO:

GRUPO:

FECHA DE ENTREGA:

¿Qué es ELT?

ELT (Extract, Load, Transform) es un enfoque moderno para la integración y procesamiento de datos que extrae los datos de diversas fuentes, los carga en un destino centralizado (como un data warehouse o un data lake) y luego aplica transformaciones sobre los datos dentro del mismo destino.

Este paradigma se diferencia del tradicional ETL (Extract, Transform, Load) en que la transformación ocurre después de la carga de datos, permitiendo aprovechar la capacidad de procesamiento del sistema de almacenamiento para realizar cálculos y modificaciones.

Fases de ELT

1. Extracción (Extract)

Consiste en obtener datos de diferentes fuentes como bases de datos relacionales, APIs, archivos CSV, JSON, logs, sensores IoT, redes sociales, etc.

Ejemplo de fuentes:

- Bases de datos relacionales (MySQL, PostgreSQL)
- Bases de datos NoSQL (MongoDB, DynamoDB, Cassandra)
- Archivos planos y estructurados (CSV, JSON, XML, Parquet)
- Data lakes (Amazon S3, Google Cloud Storage, Azure Data Lake)
- Aplicaciones SaaS (Salesforce, Google Analytics)

La extracción se puede hacer en lotes o en tiempo real (streaming) usando herramientas como Apache Kafka o AWS Kinesis

2. Carga (Load)

Los datos extraídos se cargan sin modificaciones significativas en el destino de almacenamiento centralizado, como un data warehouse o data lake,

Ejemplos de destinos:

- Data warehouses: Amazon Redshift, Google BigQuery, Snowflake, Microsoft Synapse
- Data lakes: AWS S3, Azure Data Lake, Google Cloud Storage

Dependiendo de la arquitectura, la carga puede ser incremental o en lotes completos

3. Transformación (Transform)

Una vez en el almacenamiento, los datos son transformados mediante consultas SQL, script Python o frameworks como Apache Spark y dbt.

Ejemplos de transformaciones:

- Limpieza de datos (eliminación de valores nulos, eliminación de duplicados)
- Enriquecimiento de datos (unión con otras tablas)
- Agregaciones y cálculos (sumas, promedios, conteos)
- Normalización o desnormalización de datos
- Aplicación de reglas de negocio y segmentación

¿Cuándo se usa ELT?

ELT es ideal cuando se trabaja con grandes volúmenes de datos y se tiene acceso a plataformas de almacenamiento y procesamiento con gran capacidad, como los data warehouses en la nube. Se utiliza en casos como:

1. Análisis de Big Data: ELT permite almacenar datos brutos y procesarlos de manera flexible según las necesidades del análisis
2. Procesamiento en la nube: Aprovecha la capacidad de cómputo de servicios como Redshift, BigQuery y Snowflake
3. Data Lakes: Ideal para almacenar datos sin una estructura rígida y luego transformarlos según los requisitos del usuario
4. Machine Learning y Data Science: Permite que los científicos de datos trabajen con datos en su forma original y los transformen según el modelo que necesiten.
5. ETL tradicional con volúmenes altos de datos: ELT es útil cuando las transformaciones previas ralentizan el proceso de carga, ya que permite almacenar primero y transformar después

Diferencia entre ELT y ETL

Característica	ETL (Extract, transform, load)	ELT (Extract, Load, Transform)
Orden de procesos	Extrae -> Transforma -> Carga	Extrae -> Carga -> Transforma
Ubicación de la transformación	Se hace en un servidor ETL antes de cargar los datos	Se hace en el data warehouse o data lake
Velocidad	Más lento con grandes volúmenes de datos debido a la transformación previa a la carga	Más rápido, ya que aprovecha la infraestructura de almacenamiento para transformar
Escalabilidad	Puede ser limitado por la capacidad de los servidores ETL	Altamente escalable gracias a la potencia de procesamiento de la nube
Costo	Puede ser costoso si requieres hardware potente para transformar los datos antes de cargarlos	Más económico en la nube, ya que los servicios de almacenamiento y procesamiento escalan dinámicamente
Flexibilidad	Menos flexible, ya que la estructura de los datos debe definirse antes de la carga	Más flexible, ya que se pueden almacenar datos en bruto y transformarlos después
Casos de uso	Procesos de integración de datos tradicionales, migración de bases de datos pequeñas	Análisis de Big Data, data lakes, procesamiento en la nube

Un ejemplo

Vamos a hacer un proceso de ELT utilizando el servicio de S3 y AWS.

Amazon Web Services (AWS) es una de las plataformas de computación en la nube más populares del mundo. Ofrece una amplia gama de servicios, desde almacenamiento y bases de datos hasta inteligencia artificial y computación sin servidores. AWS permite a empresas y desarrolladores desplegar aplicaciones sin necesidad de administrar infraestructura física, optimizando costos y escalabilidad.

Amazon S3 es un servicio de almacenamiento de objetos altamente escalable, seguro y duradero. Se utiliza para almacenar cualquier tipo de datos, como imágenes, videos, copias de seguridad y archivos de registro. Algunas características clave de S3 incluyen:

- Alta disponibilidad y durabilidad: Los datos en S3 están replicados en múltiples ubicaciones
- Seguridad: Soporta cifrado y controles de acceso detallados
- Versionado: Permite rastrear y recuperar versiones antiguas de archivos
- Integración con otros servicios AWS: Se usa con Lambda, Athena, CloudFront y más

S3 es ideal para almacenamiento de datos en la nube, hosting de sitios estáticos, backup y recuperación, y análisis de big data.

En este ejemplo utilizaremos un bucket de S3 para hacer la carga y transformación de los datos, para ello lo primero que haremos será crear una cuenta de AWS, para ello necesitarás añadir un

método de pago, pero no se te realizarán cargos al menos que actives un servicio que tenga algún costo, en nuestro caso como solo ocuparemos S3, no se generarán costos. También deberás elegir el plan gratuito (Basic Support) para evitar cargos adicionales. Además deberás instalar algunas dependencias necesarias como boto3:

```
pip install pyspark boto3
```

Sin mencionar que necesitas descargar el JAR que habilita la compatibilidad con S3:

1. Hadoop_aws JAR (<https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-aws/3.3.4/hadoop-aws-3.3.4.jar>)
2. aws-java-sdk-bundle JAR (<https://repo1.maven.org/maven2/com/amazonaws/aws-java-sdk-bundle/1.12.262/aws-java-sdk-bundle-1.12.262.jar>)

Deberás pegar los dos archivos en la carpeta donde hayas descargado spark, dentro de jars, en mi caso es la carpeta "C:\spark-3.5.4-bin-hadoop3\jars".

Ahora vamos a acceder a nuestra cuenta de AWS y vamos a ir a S3, donde vamos a crear un bucket S3 para almacenar los datos crudos y transformados.

1. Crear un bucket en S3
 - Ve a Amazon S3
 - Crea un bucket llamado, por ejemplo: "mi-bucket-elt"
 - Crear carpetas dentro del bucket
S3://mi-bucket-elt/datalake/raw/ -> Datos crudos
S3://mi-bucket-elt/datalake/processed/ -> Datos transformados
2. Configurar Permisos en IAM
 - Crear una política en IAM (o usar AmazonS3FullAccess)
 - Ir a AWS IAM -> Policies -> Create Policy
 - Agregar la siguiente política JSON:

The screenshot shows the AWS IAM console interface. On the left is a sidebar with navigation links like 'Panel', 'Administración del acceso', 'Personas', 'Roles', 'Políticas', 'Proveedores de identidad', 'Configuración de cuenta', 'Administración del acceso raíz', 'Novedad', 'Informes de acceso', 'Access Analyzer', 'Acceso externo', 'Acceso no utilizado', and 'Configuración del analizador'. The main area is titled 'Identity and Access Management (IAM)' and has a search bar. Below the search bar are tabs: 'Permisos' (selected), 'Entidades asociadas', 'Etiquetas', 'Versiones de la política (2)', and 'Último acceso'. The 'Permisos' tab shows a section 'Permisos definidos en esta política' with a description: 'Los permisos definidos en este documento de política especifican qué acciones se permiten o deniegan. Para definir permisos para una identidad de IAM (persona, grupo de personas o rol), asóciela a una política'. Below this is a JSON policy document with line numbers 1 through 21. The JSON defines two statements: one for 's3:ListBucket' and another for 's3:GetObject', 's3:PutObject', and 's3:DeleteObject'. Both statements are allowed on the resource 'arn:aws:s3::mi-bucket-elt/*'. At the top right of the policy editor are buttons for 'Copiar', 'Editar', 'Resumen', and 'JSON'.

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "s3:ListBucket"
8       ],
9       "Resource": "arn:aws:s3::mi-bucket-elt"
10    },
11    {
12       "Effect": "Allow",
13       "Action": [
14         "s3:GetObject",
15         "s3:PutObject",
16         "s3:DeleteObject"
17       ],
18       "Resource": "arn:aws:s3::mi-bucket-elt/*"
19    }
20  ]
21 }
```

- Guardar como S3AccessPolicy

3. Crear un usuario IAM con acceso a S3:

- Ir a IAM -> Users -> Create User
- Asignarle la política que acabas de crear S3AccessPolicy
- Guardar las credenciales (AWS_ACCESS_KEY_ID y AWS_SECRET_ACCESS_KEY)

Identity and Access Management (IAM)

Buscar en IAM

Panel

Administración del acceso

Grupos de personas

Personas

Roles

Políticas

Personas (3) Información

Un usuario de IAM es una identidad con credenciales válidas a largo plazo que se utiliza para interactuar con AWS en una cuenta.

Buscar

<input type="checkbox"/>	Nombre de usuario	Ruta	Grupo	Última actividad	MFA	Ant
<input type="checkbox"/>	developer	/	0	-	-	-
<input type="checkbox"/>	javie	/	0	-	-	-
<input type="checkbox"/>	usuario-s3-elt	/	0	✓ hace 1 hora	-	-

Identity and Access Management (IAM)

Buscar en IAM

Panel

Administración del acceso

Grupos de personas

Personas

Roles

Políticas

Proveedores de identidad

Configuración de cuenta

Administración del acceso raíz

Novedad

Informes de acceso

Access Analyzer

Acceso externo

Acceso no utilizado

Configuración del analizador

usuario-s3-elt Información

Resumen

ARN
arn:aws:iam::339712950707:user/usuario-s3-elt

Acceso a la consola
Desactivada

Clave de acceso 1
AKIAU6GDYIGZZ7U5LE
Usado hoy. Ayer anti

Creado
March 11, 2025, 12:34 (UTC-06:00)

Último inicio de sesión en la consola
-

Clave de acceso 2
Crear clave de acceso

Permisos

Grupos

Etiquetas (1)

Credenciales de seguridad

Último acceso

Políticas de permisos (1)

Los permisos se definen mediante políticas asociadas a la persona directamente o a través de grupos.

Buscar

Filtrar por Tipo

Todos los tipos

<input type="checkbox"/>	Nombre de la política	Tipo	Adjuntado a través
<input type="checkbox"/>	S3AccessPolicy	Administrada por el cliente	Directamente

No hay dispositivos MFA. Asigne un dispositivo MFA para mejorar la seguridad del entorno de AWS.
Asignar dispositivo MFA

Claves de acceso (1)

Utilice las claves de acceso para enviar llamadas mediante programación a AWS desde AWS CLI, Herramientas de AWS para PowerShell, AWS SDK o llamadas directas a la API de AWS. Puede tener un máximo de dos claves de acceso (activas o inactivas) a la vez. Más información

Crear clave de acceso

Acciones

Descripción acceso s3 editor	Estado Active
Último uso hace 1 hora	Creado Ayer
Última región utilizada us-east-1	Último servicio utilizado s3

Ahora deberás configurar apache spark para acceder a S3 y reemplazar ACCESS_KEY y SECRET_KEY con las credenciales que te dio el usuario IAM en AWS.

```
[1]: from pyspark.sql import SparkSession
```

Configuramos Apache Spark para acceder a S3

```
•[2]: spark = SparkSession.builder \
      .appName("ELT-S3") \
      .config("spark.hadoop.fs.s3a.impl", "org.apache.hadoop.fs.s3a.S3AFileSystem") \
      .config("spark.hadoop.fs.s3a.access.key", "ACCESS_KEY") \
      .config("spark.hadoop.fs.s3a.secret.key", "SECRET_KEY") \
      .config("spark.hadoop.fs.s3a.endpoint", "s3.amazonaws.com") \
      .getOrCreate()
```

Ahora vamos a proseguir con la parte de extracción de los datos, vamos a leer los datos de un archivo csv, además vamos a activar la opción de header y que infiera el esquema de los datos.

▼ Extracción

```
[4]: orders_df = spark.read.csv("orders.csv", header=True, inferSchema=True)
```

```
[5]: customers_df = spark.read.csv("customers.csv", header=True, inferSchema=True)
```

```
[6]: products_df = spark.read.csv("products.csv", header=True, inferSchema=True)
```

```
: orders_df.show()
```

```
+-----+-----+-----+-----+-----+-----+
|order_id|customer_id|product_id|quantity|price|order_date|
+-----+-----+-----+-----+-----+-----+
|      1|      101|      201|      2|   50|2025-03-10|
|      2|      102|      202|      1|   20|2025-03-11|
|      3|      103|      203|      3|   15|2025-03-11|
|      4|      101|      201|      1|   50|2025-03-12|
|      5|      104|      204|      5|   30|2025-03-12|
+-----+-----+-----+-----+-----+-----+
```

```
: customers_df.show()
```

```
+-----+-----+-----+-----+
|customer_id|  name|          email|signup_date|
+-----+-----+-----+-----+
|      101| Alice| alice@example.com| 2025-01-01|
|      102|  Bob|  bob@example.com| 2025-01-05|
|      103|Charlie|charlie@example.com| 2025-02-01|
|      104| David| david@example.com| 2025-02-10|
+-----+-----+-----+-----+
```

```
: products_df.show()
```

```
+-----+-----+-----+-----+
|product_id|product_name|  category|price|
+-----+-----+-----+-----+
|      201|      Laptop|Electronics| 1000|
|      202|       Mouse|Accessories|   20|
|      203|   Keyboard|Accessories|   50|
|      204|     Monitor|Electronics|  300|
+-----+-----+-----+-----+
```

Ahora vamos a cargar los datos en S3, es decir en la nube de AWS

Cargar datos en S3 ¶

Subimos los datos sin procesar al Data Lake en S3, ahora los datos están almacenados en S3 en formato Parquet

```
: orders_df.write.mode("overwrite").parquet("s3a://mi-bucket-elt/datalake/raw/orders")
```

```
: customers_df.write.mode("overwrite").parquet("s3a://mi-bucket-elt/datalake/raw/customers")
```

```
: products_df.write.mode("overwrite").parquet("s3a://mi-bucket-elt/datalake/raw/products")
```

orders/

Objetos

Propiedades

Objetos (2)





 Copiar URI de S3

Copiar URL

 Descargar

Los objetos son las entidades fundamentales que se almacenan en Amazon S3. Puede utilizar el [inventario de Amazon S3](#) para obtener una lista de sus objetos, tendrá que concederles permisos de forma explícita. [Más información](#)

🔍 *Buscar objetos por prefijo*

<input type="checkbox"/>	Nombre ▲	Tipo ▼	Última modificación ▼	Tamaño ▼	Clase de almacenamiento
<input type="checkbox"/>	 _SUCCESS	-	12 Mar 2025 12:04:23 PM CST	0 B	Estándar
<input type="checkbox"/>	 part-00000-f39b259c-264b-40a6-b7d1-41d8c098840c-c000.snappy.parquet	parquet	12 Mar 2025 12:04:21 PM CST	1.7 KB	Estándar

customers/

Objetos

Propiedades

Objetos (2)





 Copiar URI de S3

 Copiar URL

 Descargar

Los objetos son las entidades fundamentales que se almacenan en Amazon S3. Puede utilizar el [inventario de Amazon S3](#) para obtener los objetos, tendrá que concederles permisos de forma explícita. [Más información](#)

🔍 *Buscar objetos por prefijo*





<input type="checkbox"/>	Nombre ▲	Tipo ▼	Última modificación ▼	Tamaño ▼	Clase de almacenamiento
<input type="checkbox"/>	 _SUCCESS	-	12 Mar 2025 12:06:03 PM CST	0 B	Estándar
<input type="checkbox"/>	 part-00000-b6b42dec-33de-42b2-b233-943d703ed5df-c000.snappy.parquet	parquet	12 Mar 2025 12:06:01 PM CST	1.3 KB	Estándar

aws [Alt+S]

Amazon S3 > Buckets > mi-bucket-elt > datalake/ > raw/ > products/



products/

Objetos | Propiedades

Objetos (2)   Copiar URI de S3  Copiar URL  Descargar

Los objetos son las entidades fundamentales que se almacenan en Amazon S3. Puede utilizar el [inventario de Amazon S3](#) para obtener sus objetos, tendrá que concederles permisos de forma explícita. [Más información](#)

Buscar objetos por prefijo

<input type="checkbox"/>	Nombre	Tipo	Última modificación	Tamaño	Clase de almacenamiento
<input type="checkbox"/>	 _SUCCESS	-	12 Mar 2025 12:08:09 PM CST	0 B	Estándar
<input type="checkbox"/>	 part-00000-08dd3b69-efa4-4789-ac2b-671c83b25c85-c000.snappy.parquet	parquet	12 Mar 2025 12:08:07 PM CST	1.3 KB	Estándar

Ahora vamos a realizar transformaciones de los datos, primero vamos a leer los datos directamente desde S3.

Transformamos datos en Spark desde S3

Leemos los datos directamente desde S3

```
orders_df = spark.read.parquet("s3a://mi-bucket-elt/datalake/raw/orders")
customers_df = spark.read.parquet("s3a://mi-bucket-elt/datalake/raw/customers")
products_df = spark.read.parquet("s3a://mi-bucket-elt/datalake/raw/products")
```

Luego de leer los datos directamente desde S3, vamos a renombrar los dataframes para evitar colisiones de nombres y vamos a hacer un join entre los dataframes.

Realizamos transformaciones

```
from pyspark.sql.functions import col, sum
```

Unimos los datos

Renombramos DataFrames para evitar colisiones de nombres ¶

```
orders_df = orders_df.alias("orders")
customers_df = customers_df.alias("customers")
products_df = products_df.alias("products")
```

```
sales_df = orders_df.join(customers_df, "customer_id").join(products_df, "product_id")
```

Ahora vamos a agrupar y calcular el total gastado por el cliente y calcular las ventas por categoría

Agrupar y calcular el total gastado por cliente

```
sales_by_customer = sales_df.groupBy("customer_id", "customers.name") \
    .agg(sum(col("quantity") * col("products.price")).alias("total_spent"))
```

Calcular ventas por categoría

```
sales_by_category = sales_df.groupBy("category") \
    .agg(sum(col("quantity") * col("products.price")).alias("total_sales"))
```

Mostramos el total calculado por cliente y el total por categoría

```
sales_by_customer.show()
```

```
+-----+-----+-----+
|customer_id|  name|total_spent|
+-----+-----+-----+
|         104| David|         1500|
|         103|Charlie|          150|
|         102|   Bob|           20|
|         101| Alice|        3000|
+-----+-----+-----+
```

```
sales_by_category.show()
```

```
+-----+-----+
| category|total_sales|
+-----+-----+
|Electronics|         4500|
|Accessories|          170|
+-----+-----+
```

Ahora vamos a guardar los datos transformados en s3, en la carpeta

“datalake/processed/sales_by_customer” y “datalake/processed/sales_by_category”

Guardamos los datos transformados en S3



```
sales_by_customer.write.mode("overwrite").parquet("s3a://mi-bucket-elt/datalake/processed/s
```

```
sales_by_category.write.mode("overwrite").parquet("s3a://mi-bucket-elt/datalake/processed/s
```

Amazon S3 > Buckets > mi-bucket-elt > datalake/ > processed/

Concesiones de acceso
Puntos de acceso
Puntos de acceso del objeto
Lambda
Puntos de acceso de varias regiones
Operaciones por lotes
Analizador de acceso de IAM para S3

Configuración de bloqueo de acceso público correspondiente a esta cuenta

▼ Storage Lens
Paneles
Grupos de Storage Lens
Configuración de AWS Organizations

processed/

Objetos Propiedades

Objetos (2)

[Copiar URI de S3](#) [Copiar URL](#) [Descargar](#) [Abrir](#) [Eliminar](#) [Acciones](#) [Crear carpeta](#) [Cargar](#)

Los objetos son las entidades fundamentales que se almacenan en Amazon S3. Puede utilizar el [inventario de Amazon S3](#) para obtener una lista de todos los objetos de su bucket. Para que otras personas obtengan acceso a sus objetos, tendrá que concederles permisos de forma explícita. [Más información](#)

Buscar objetos por prefijo

<input type="checkbox"/>	Nombre	Tipo	Última modificación	Tamaño	Clase de almacenamiento
<input type="checkbox"/>	sales_by_category/	Carpeta	-	-	-
<input type="checkbox"/>	sales_by_customer/	Carpeta	-	-	-

Amazon S3 > Buckets > mi-bucket-elt > datalake/ > processed/ > sales_by_category/

Concesiones de acceso
Puntos de acceso
Puntos de acceso del objeto
Lambda
Puntos de acceso de varias regiones
Operaciones por lotes
Analizador de acceso de IAM para S3

Configuración de bloqueo de acceso público correspondiente a esta cuenta

▼ Storage Lens
Paneles
Grupos de Storage Lens
Configuración de AWS Organizations

Características destacadas 11

► AWS Marketplace para S3

sales_by_category/

Objetos Propiedades

Objetos (2)

[Copiar URI de S3](#) [Copiar URL](#) [Descargar](#) [Abrir](#) [Eliminar](#) [Acciones](#) [Crear carpeta](#) [Cargar](#)

Los objetos son las entidades fundamentales que se almacenan en Amazon S3. Puede utilizar el [inventario de Amazon S3](#) para obtener una lista de todos los objetos de su bucket. Para que otras personas obtengan acceso a sus objetos, tendrá que concederles permisos de forma explícita. [Más información](#)

Buscar objetos por prefijo

<input type="checkbox"/>	Nombre	Tipo	Última modificación	Tamaño	Clase de almacenamiento
<input type="checkbox"/>	_SUCCESS	-	12 Mar 2025 12:53:47 PM CST	0 B	Estándar
<input type="checkbox"/>	part-00000-4-dd08c88-a2f0-4fed-bd29-9a181550099f-c000.snappy.parquet	parquet	12 Mar 2025 12:53:45 PM CST	800.0 B	Estándar

The screenshot shows the AWS S3 console interface. The breadcrumb navigation at the top indicates the path: Amazon S3 > Buckets > mi-bucket-elt > datalake/ > processed/ > sales_by_customer/. The left sidebar contains navigation links for access control, IAM, and Storage Lens. The main content area shows the 'sales_by_customer/' bucket with a table of objects. The table has columns for Name, Type, Last Modified, Size, and Storage Class. Two objects are listed: a directory named 'SUCCESS' and a file named 'part-00000-31cc549d-3480-4eba-a7bf-9845474a77cb-c000.snappy.parquet'.

Nombre	Tipo	Última modificación	Tamaño	Clase de almacenamiento
SUCCESS	-	12 Mar 2025 12:47:32 PM CST	0 B	Estándar
part-00000-31cc549d-3480-4eba-a7bf-9845474a77cb-c000.snappy.parquet	parquet	12 Mar 2025 12:47:30 PM CST	1.0 KB	Estándar

Por hacer

- Realizar un proceso de ELT con Spark, primero extraer los datos desde “ventas.csv” y “productos.csv”
- Guardar los datos en Amazon S3 en un bucket llamado “elt-practica” y dentro de una carpeta llamada “bronce”
- Hacer el join entre ventas.csv y productos.csv en Spark
- Agrupar por categoría y calcular las ventas totales
- Guardar en los datos procesados en el bucket “elt-practica” dentro de una carpeta llamada “oro”.