USO

AL EJECUTAR TODAS LAS CELDAS DE **TP_Mateo_Ponce_parte1_bronze** se creará el directorio bronze del delta lake con los datos crudos de los Endpoints.

Luego ejecutar todas las celdas de **TP_Mateo_Ponce_parte2_silver** para cargar los datos de la carpeta bronze, transformar los datos y guardarlos en la capa silver.

Por último, ejecutar todas las celdas de **TP_Mateo_Ponce_parte3_gold** para cargar los datos de la capa silver, hacer transformaciones y cálculos avanzados, visualizar y finalmente guardar los dataframes combinados y las agregaciones en la capa gold.

Las celdas de codigo se ejecutan tal cual estan entregadas se utilizan los parametros por defecto en caso de faltar parametros para funciones.

Resumen y explicación del trabajo

PARTE 1: capa bronze

En primer lugar, vamos a extraer datos de 3 endpoints

-coins/list: endpoint de datos estáticos, devuelve todas las criptomonedas disponibles en coingecko con id, symbol y name. Se almacena en bronze/coingecko_api /coins_list *en modo overwrite.*

-coins/markets: endpoint para datos temporales, información del día actual sobre criptomonedas (current_price, market_cap, total_volume y price_change_percentage_24h, etc) los datos del endpoint se actualizan todos los dias. Los datos se almacenan en en bronze/coingecko_api/coins_markets y se particiona por fecha de extracción, una vez creado un directorio se actualiza o se insertan datos con un **merge** evitando duplicados o inconsistencias

-{coin_id}/market_chart: endpoint para datos historicos, precio, capitalización de mercado y volumen de hasta 365 días atrás de criptomonedas individuales. se particiona por fecha de extracción, una vez creado un directorio se actualiza o se insertan datos con un **merge**.

Cada endpoint cuenta con una funcion “**fetch**” para realizar la petición y extracción a la API y devolver un dataframe y una funcion “**save**” que guarda los datos en la capa bronze del delta lake.

En el caso de **market_chart,** se extraer datos de las 5 criptomonedas mas importantes y se combinado los resultados en un dataframe, luego se utilizarán los datos de estas criptomonedas en la **capa Gold.**

Los datos extraídos serán almacenados en la capa bronze, sin muchas modificaciones.

Ejemplo de estructura endpoint incremental:

bronze/coins_markets/

├── extract_date=2025-05-25/

│ └── archivos.parquet

├── extract_date=2025-05-26/

│ └── archivos.parquet

└── _delta_log/

Detalle: en ***coins/markets*** el predicado del merge utilizando extract_date no produce duplicados ya que el endpoint extrae datos de las monedas en el día actual, todos los dias varia.
Pero **market_chart** extrae datos de los ultimos 30 dias a partir de la fecha de extracción, el predicado del merge "target.coin_id = source.coin_id AND target.extract_date = source.extract_date", al ejecutar por ejemplo el codigo 2 dias seguidos se generan **duplicados** en 29 dias. Esto se soluciona en la capa silver.

PARTE 2: capa silver

Cada endpoint tiene una funcion **load** en donde se cargar los datos brutos mas recientes desde la capa bronce.

TRANSFORMACIONES coins_markets

El dataframe **coins_markets** contiene los datos más recientes del mercado de criptomonedas.

El dataframe contiene 31 columnas y varios valores nulos en max_supplY roi.percentage.

```
Fecha más reciente: 2025-06-16
Registros de la última extracción: 75
                id  symbol              name  \
0          bitcoin     btc           Bitcoin
1         ethereum     eth          Ethereum
2           tether    usdt            Tether
3           ripple     xrp               XRP
4      binancecoin     bnb               BNB
5           solana     sol            Solana
6         usd-coin    usdc              USDC
7         dogecoin    doge          Dogecoin
8             tron     trx              TRON
9     staked-ether   steth  Lido Staked Ether
10         cardano     ada           Cardano
11     hyperliquid    hype       Hyperliquid
12  wrapped-bitcoin    wbtc    Wrapped Bitcoin
13    wrapped-steth  wsteth      Wrapped stETH
14              sui     sui               Sui

                                                image  current_price  \
0   https://coin-images.coingecko.com/coins/images...  107788.000000
1   https://coin-images.coingecko.com/coins/images...    2583.150000
2   https://coin-images.coingecko.com/coins/images...       1.000000
3   https://coin-images.coingecko.com/coins/images...       2.280000
4   https://coin-images.coingecko.com/coins/images...     653.930000
...
13          None             NaN    2025-06-16  2025-06-16T23:15:18.485028
14          None             NaN    2025-06-16  2025-06-16T23:15:18.485028

[15 rows x 31 columns]
```

En la function **df_market_select_column** se reduce el número de columnas en el dataframe original (31), seleccionando las más relevantes para el analisis de datos, el dataframe resultante cuenta con 18 columnas.

La funcion **df_market_fill_nulls** rellena los nulos de las columnas max_supply y roi.percentage. En el caso de max_supply utilizamos la mediana de la columna como valor default y el roi.percentage 0 ya que los nulls se deben a que las monedas son nuevas y no tienen un retorno esperado definido.

TRANFORMACIONES market_chart

DataFrame con los datos de los ultimos 30 dias de las 5 mejores monedas del mercado de criptomonedas.
Tiene las columnas: ['prices', 'market_caps', 'total_volumes', 'extract_date','coin_id'] las 3 primeras columnas tienen una lista con dos elementos: [timestamp, valor del campo].

	prices	market_caps	total_volumes	extract_date	coin_id
0	[1747526400000.0, 103212.36483885496]	[1747526400000.0, 2050318521477.805]	[1747526400000.0, 18950056010.479057]	2025-06-16	bitcoin
1	[1747612800000.0, 106030.6376831359]	[1747612800000.0, 2104889720872.8306]	[1747612800000.0, 30744060179.802963]	2025-06-16	bitcoin
2	[1747699200000.0, 105629.41580436694]	[1747699200000.0, 2098485415895.0496]	[1747699200000.0, 43339734153.75087]	2025-06-16	bitcoin
3	[1747785600000.0, 106786.71995834043]	[1747785600000.0, 2121778617273.821]	[1747785600000.0, 36393687094.25122]	2025-06-16	bitcoin
4	[1747872000000.0, 109665.86371625263]	[1747872000000.0, 2178838967665.505]	[1747872000000.0, 60722883113.84]	2025-06-16	bitcoin
5	[1747958400000.0, 111560.356938144]	[1747958400000.0, 2214712145787.249]	[1747958400000.0, 52218408239.45193]	2025-06-16	bitcoin
6	[1748044800000.0, 107216.66856870624]	[1748044800000.0, 2131595896624.4407]	[1748044800000.0, 49251745837.96465]	2025-06-16	bitcoin

La funcion transform_market_chart_data, realiza 3 transformaciones importantes sobre este dataframe:

- **Se crea una columna date extrayendo el valor timestamp de cada tupla.**
- **Las columnas prices, market_caps, total_volumes originalmente de tuplas se convierten en flotantes redondeados en 2 decimales extrayendo su valor de las listas y eliminando la fecha.**
- **se reordenan las columnas del dataframe**

CAMBIO IMPORTANTE EN EL ALMACENAMIENTO de df_market_chart

Recordemos que cada ejecucion del enpoint y cada extract_date contiene 30 días de datos históricos.

Al almacenar los datos de market_chart en la capa bronze, en la cual no habia columna date solo extract_date utilizaba Merge usando extract_date + coin_id como clave.

Problema clave: La clave del merge (coin_id + extract_date) no previene duplicados en los datos históricos. Si extraes los mismos datos en diferentes días, terminarás con múltiples registros para la misma combinación coin_id + extract_date.

- Duplicación masiva de datos (30 días × N extracciones)

- Consume mucho storage

en la capa silver al procesar los datos obtengo la columna date y en este caso utilizo un Merge date + coin_id como clave.

PROS:

- Sin duplicación: cada combinación (date, coin_id) aparece solo una vez

- Storage eficiente

- Queries más simples y rápidas

- Datos siempre actualizados con la última extracción

PARTE 3: capa Gold

El endpoint ***coins/markets*** *propociona toda la informacion de las monedas al día de hoy (31 columnas) pero sin datos historicos, el endpoint* ***{coin_id}/market_chart*** *tiene datos históricos de los ultimos 30 dias pero con menos columnas. Combinando los dataframes tenemos toda la informacion de las monedas a día de hoy y ademas datos históricos críticos de los ultimos 30 dias.*

En esta etapa del proyecto se aplicaron transformaciones avanzadas combinando datos actuales de las criptomonedas con datos históricos, ademas se realizaron agregaciones y cálculos avanzados; al combinar los datos

históricos con los actuales calculamos el porcentaje de crecimiento de las 5 criptomonedas mas importantes en los ultimos 30 dias.

En la capa **Gold** obtenemos un **insight** importante; a la fecha de hoy 19/06/2025, en el último mes las 5 criptomonedas más importantes **han perdido valor**.

	coin_id	first_price	last_price	extract_date	pct_change_30d
0	binancecoin	650.95	645.19	2025-06-19	-0.884861
1	bitcoin	106786.72	104890.81	2025-06-19	-1.775417
2	ethereum	2524.27	2519.54	2025-06-19	-0.187381
3	ripple	2.36	2.16	2025-06-19	-8.474576
4	tether	1.00	1.00	2025-06-19	0.000000

Resumen del codigo de la capa Gold:

-Se crearon funciones para cargar los datos más recientes y limpios desde la capa Silver

-Se desarrolló una función que integra los datos actuales (última extracción) con los históricos (últimos 30 días), asegurando la homogeneidad de columnas y tipos de datos. Se corrigió el tratamiento de columnas faltantes para evitar errores de tipo al almacenar en Delta Lake.

-Conversión de tipos para almacenamiento:

Se forzaron las columnas problemáticas (como fechas y strings) a tipo string antes de guardar en Delta Lake, previniendo errores de conversión y asegurando la compatibilidad con el formato de almacenamiento.

Almacenamiento incremental y seguro

Se implementó una lógica de merge para evitar duplicados y permitir actualizaciones eficientes en la capa Gold, utilizando claves adecuadas para cada tipo de dato.

Cálculo y visualización de métricas:

Se agregaron funciones para calcular **el cambio porcentual de precio** de las principales criptomonedas en los últimos 30 días y para **visualizar** estos resultados de manera clara.

Estos cambios aseguran que la capa Gold contenga información consolidada, sin duplicados y lista para análisis avanzados y visualización.