



Proyecto End-to-End Meteo en Azure

Ingesta Streaming y Batch, Lakehouse con
Databricks y Arquitectura Medallion

¿Qué hemos construido?

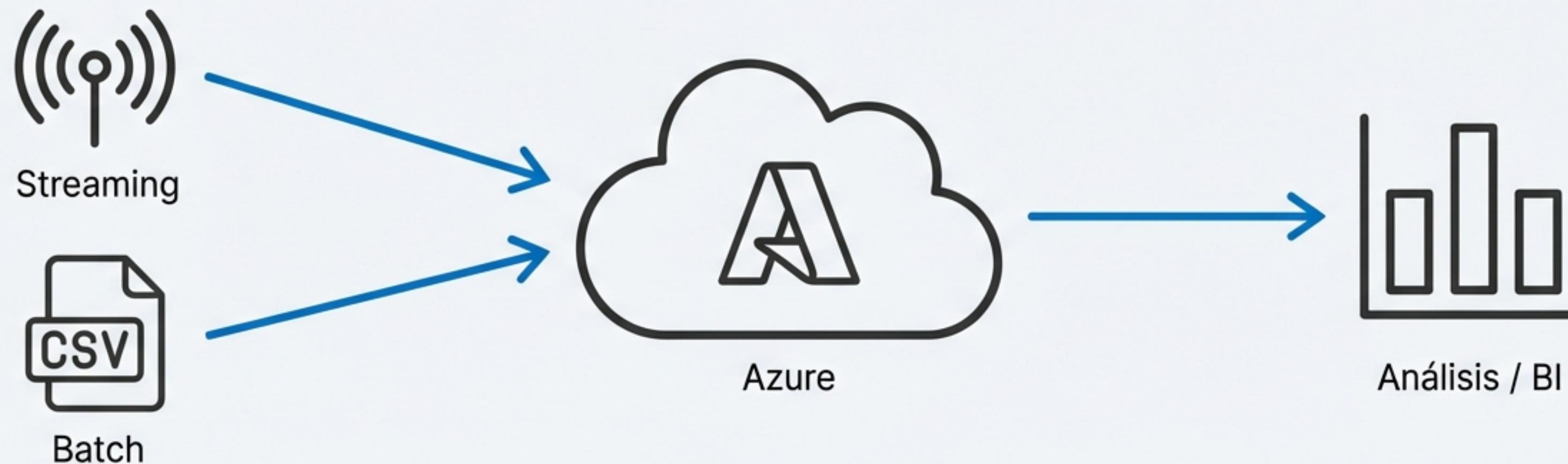
- Un pipeline de datos automatizado que procesa información meteorológica de dos fuentes distintas: una en tiempo real y otra por lotes.
- El objetivo es unificar, limpiar y agregar estos datos para que estén listos para su consumo en herramientas de Business Intelligence.

Fuentes de Datos y Objetivo Final:

Fuente Streaming: Simulación de una estación meteo que envía datos en formato JSON en tiempo real.

Fuente Batch: Ficheros CSV generados cada hora con un resumen histórico.

Destino: Construir una capa de datos "Gold" fiable y optimizada dentro de un Lakehouse en Azure.



El Ecosistema de Nuestro Proyecto



Resource Group (`rg-meteo-end2end`): El contenedor lógico para gestionar todos los recursos del proyecto de forma centralizada.



Scripts Python Externos: Simulan la estación meteorológica, generando nuestros datos de origen. Se ejecutan en local, pero podrían estar en una VM.



Azure Event Hubs (`ehns-meteo`): Nuestro punto de entrada para los datos en tiempo real. Actúa como un búfer masivo y escalable.



Storage Account (ADLS Gen2): El corazón de nuestro almacenamiento, configurado como un Data Lakehouse (`stmeteoend2end`) con el espacio de nombres jerárquico activado.



Azure Databricks (`dbw-meteo`): El cerebro de la operación. Aquí se ejecuta toda la lógica de procesamiento y transformación con PySpark.

Simulando la Generación de Datos con Python

- Dos scripts (`streaming_producer.py` y `batch_generator.py`) simulan una estación meteorológica.

Modo Streaming (tiempo real):

- Genera una lectura (temperatura, humedad, etc.) cada pocos segundos.
- Envía los datos como un mensaje **JSON** directamente al Event Hub `eh-weather`.
- Representa la ingesta de datos desde sensores IoT.

```
{  
    "station_id": "STN001",  
    "timestamp": "2023-10-27T10:00:05Z",  
    "temperature": 15.2,  
    "humidity": 65.5  
}
```

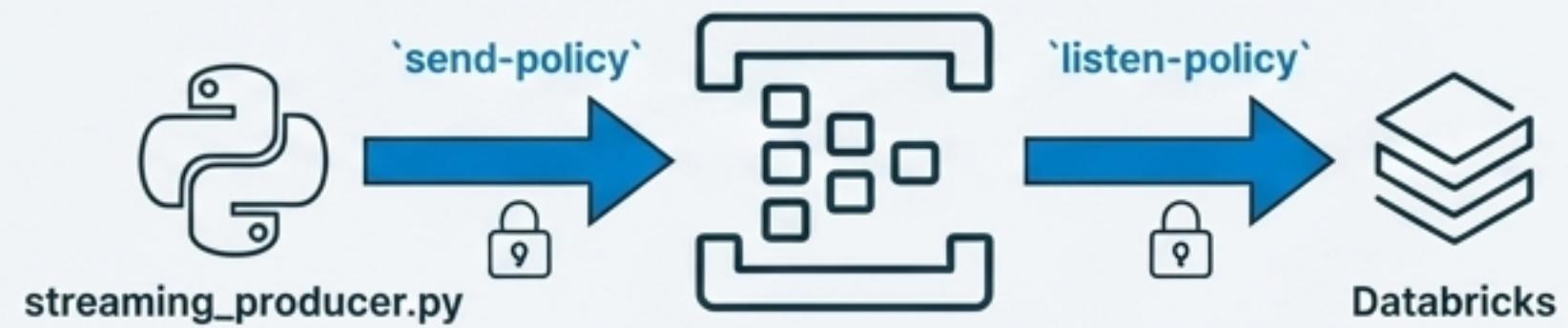
Modo Batch (por lotes):

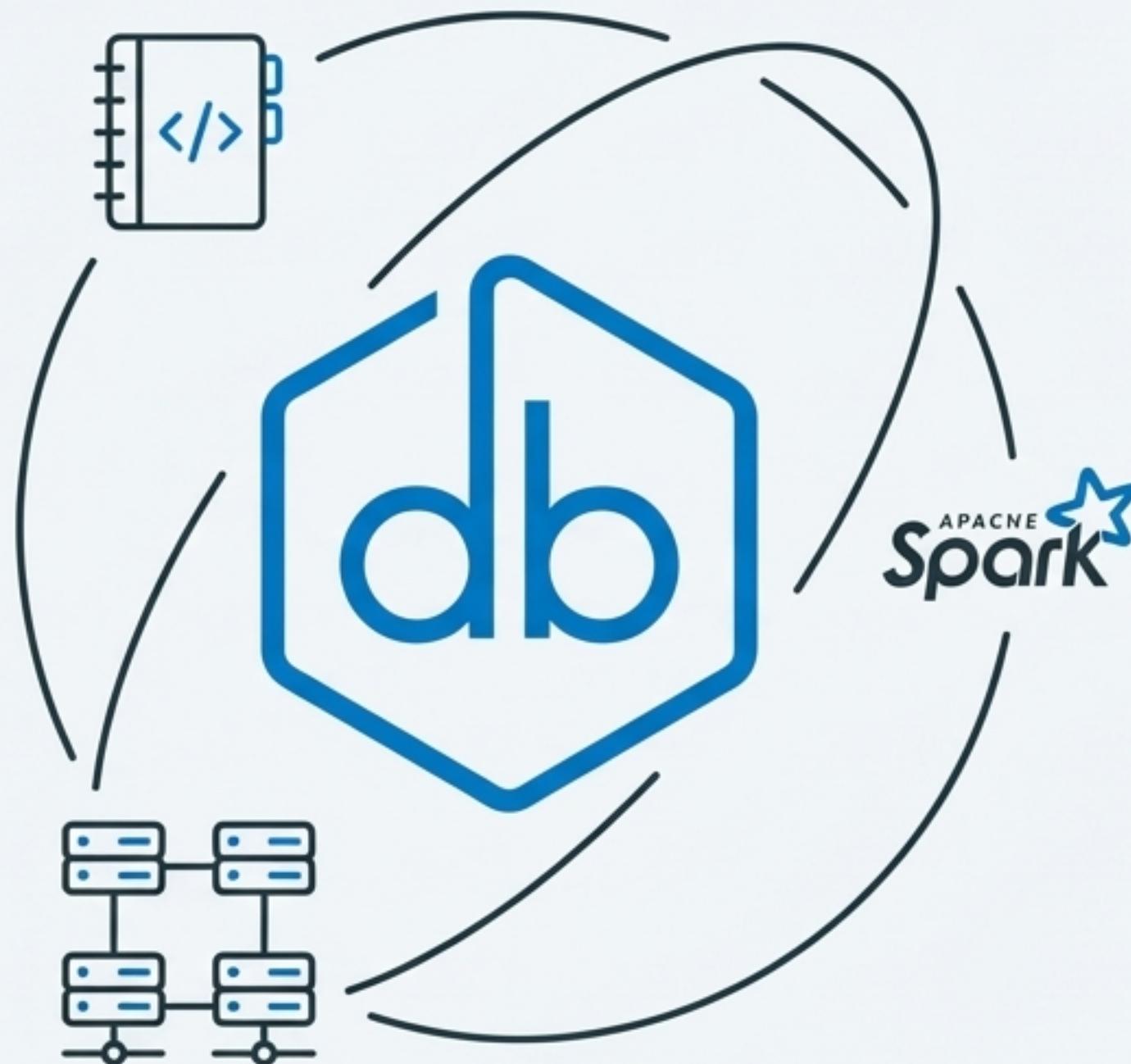
- Acumula lecturas durante una hora y genera un único fichero **CSV**.
- Sube el fichero a la zona de aterrizaje de nuestro Data Lake: `/landing/weather_batch`.

station_id	timestamp	temperature	humidity
STN001	...11:00:00Z	16.1	64.0
STN001	...11:00:05Z	16.2	63.9
STN001	...11:59:55Z	17.5	62.1

Ingesta Streaming: La Puerta de Entrada para el Tiempo Real

- **Azure Event Hubs:** Un servicio de ingestión de datos de alto rendimiento, diseñado para recibir millones de eventos por segundo.
- **Nuestro Event Hub: `eh-weather`**
 - Recibe los mensajes JSON del script de Python en tiempo real.
 - Actúa como un búfer duradero, garantizando que no se pierdan datos aunque el sistema de procesamiento falle temporalmente.
- **Seguridad con Políticas SAS:**
 - **`send-policy`**: Usada por el script Python para tener permiso exclusivo de envío.
 - **`listen-policy`**: Usada por Databricks para tener permiso exclusivo de lectura.





Databricks: El Motor de Procesamiento Central

¿Qué es?

- Una plataforma unificada para datos y IA basada en Apache Spark donde transformamos los datos.

¿Cómo lo usamos?

- Ejecutamos toda nuestra lógica en **Notebooks de PySpark** sobre un **Cluster de Spark**, que proporciona la potencia de cómputo distribuida.

¿Por qué Databricks y no Azure Data Factory (ADF)?

- **Ejecución Nativa:** Toda la lógica está en notebooks de Spark. Databricks los ejecuta de forma más eficiente y con menor latencia que ADF.
- **Simplicidad:** Se evita la sobrecarga de configurar una integración adicional, manteniendo el pipeline en un único entorno.
- **Control Granular:** Ofrece control total sobre el clúster, las dependencias y la gestión de reintentos, lo que es ideal para un escenario centrado en Spark.

Estructurando Nuestro Lakehouse: La Arquitectura Medallion

Un patrón de diseño que organiza los datos en un Lakehouse en capas de calidad progresiva.



Capa BRONZE (Bronce):

- Datos crudos, sin procesar. Una copia exacta de las fuentes originales.
- Estado: Bruto, inmutable.

Capa SILVER (Plata):

- Datos limpios, validados, desduplicados y enriquecidos.
- Estado: Filtrado y estandarizado. La "fuente única de la verdad".

Capa GOLD (Oro):

- Datos agregados y optimizados para el negocio, listos para el análisis.
- Estado: Agregado y optimizado para BI.

Tecnología Clave: **Delta Lake**

Es el formato de almacenamiento que potencia esta arquitectura, aportando fiabilidad (transacciones **ACID**) a nuestro Data Lake.

Capa Bronze: El Aterrizaje de los Datos en Bruto

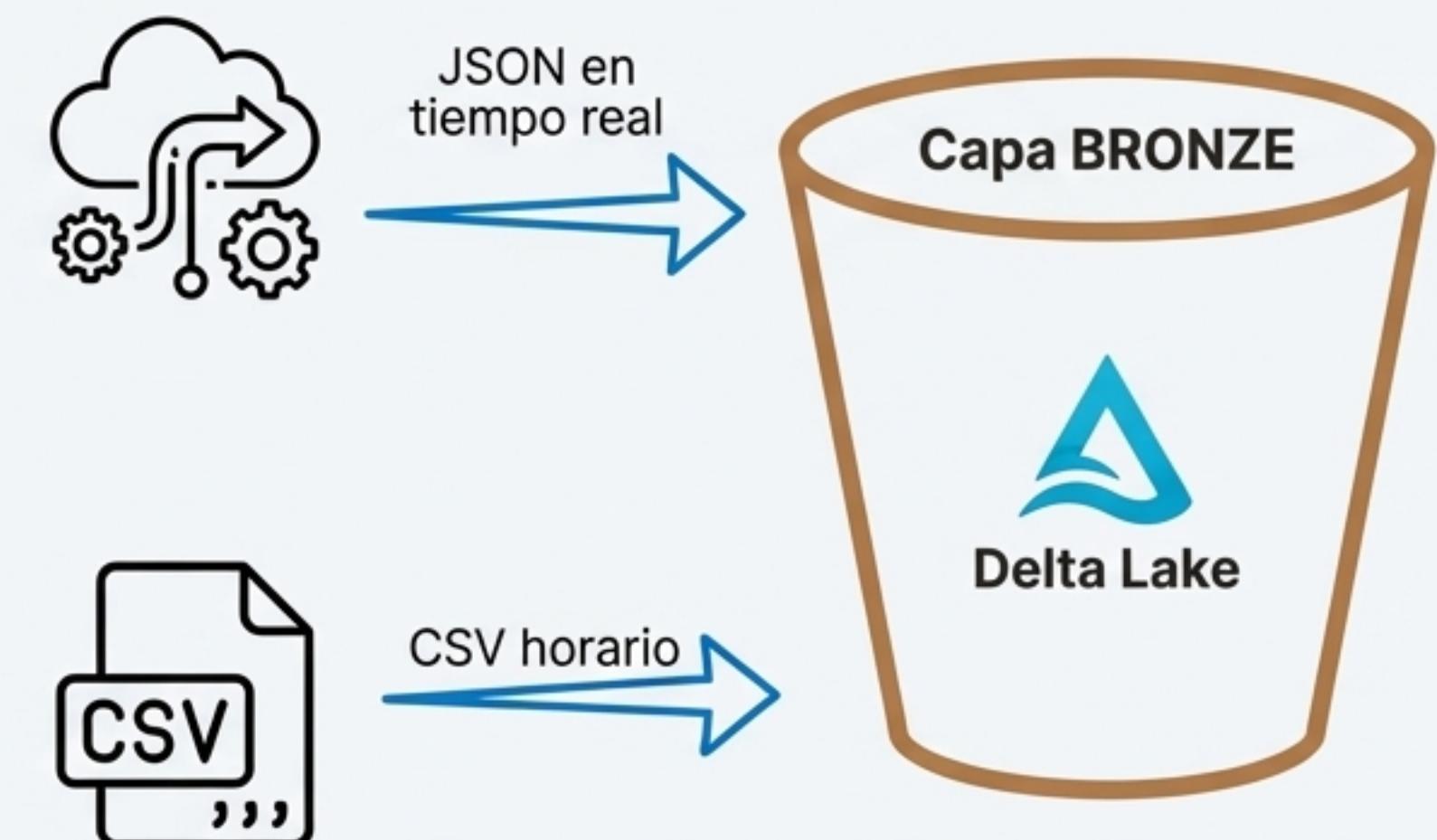
- El objetivo es almacenar una copia fiel y sin procesar de los datos de origen, sirviendo como un registro histórico inmutable.

Ingesta Streaming:

- Un notebook de Databricks lee continuamente de Event Hubs.
- Escribe los datos JSON en la tabla Delta: `datalake/bronze/streaming/weather`.

Ingesta Batch:

- Otro proceso recoge los ficheros CSV desde `/landing/weather_batch`.
- Los convierte y añade a la tabla Delta: `datalake/bronze/batch/weather`.

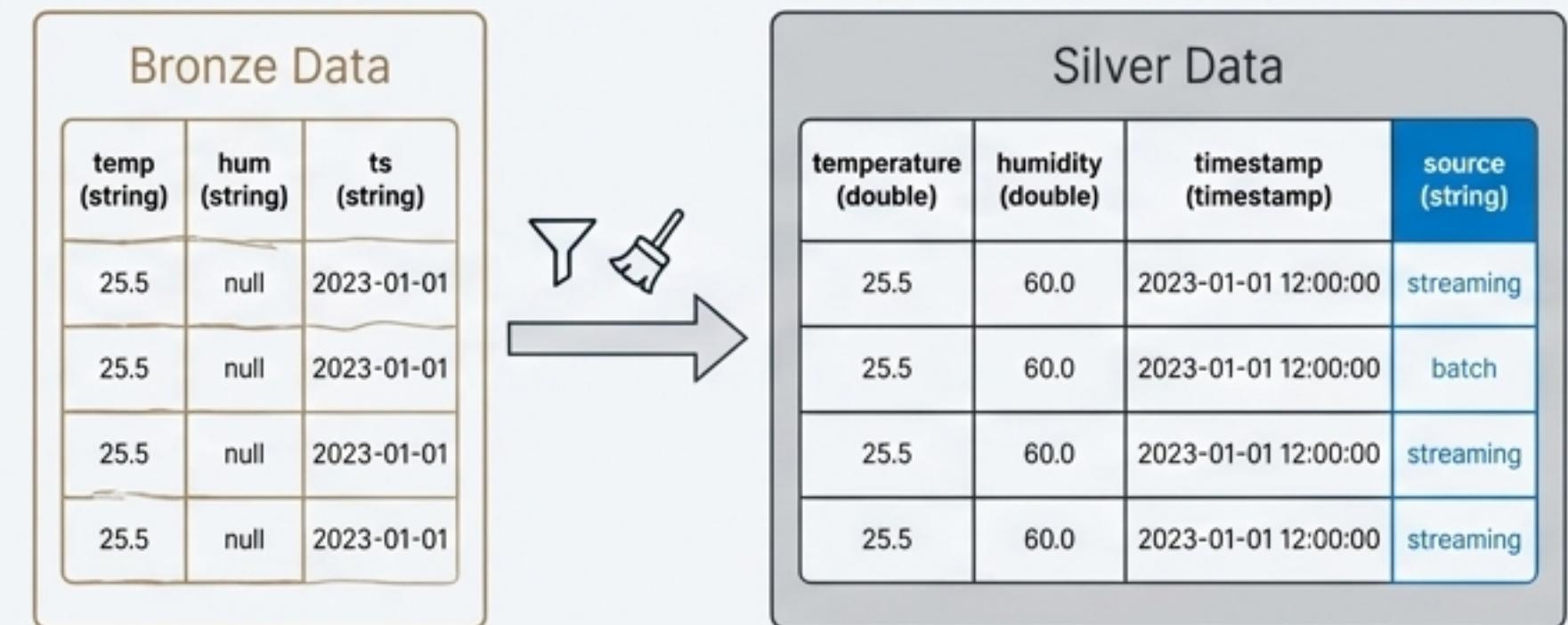


Capa Silver: Aportando Limpieza y Estanderización

- El objetivo es crear una “única fuente de la verdad” (Single Source of Truth). Los datos aquí son fiables, consistentes y uniformes.

Transformaciones Aplicadas:

- **Limpieza:** Eliminación de duplicados y gestión de valores nulos.
- **Conversión de Tipos:** Se asegura de que las fechas sean timestamp, los números double, etc.
- **Normalización:** Se estandarizan los nombres de las columnas.
- **Enriquecimiento:** Se añade una columna source para identificar el origen de cada registro (streaming o batch).



Capa Gold: Datos Agregados y Listos para el Análisis

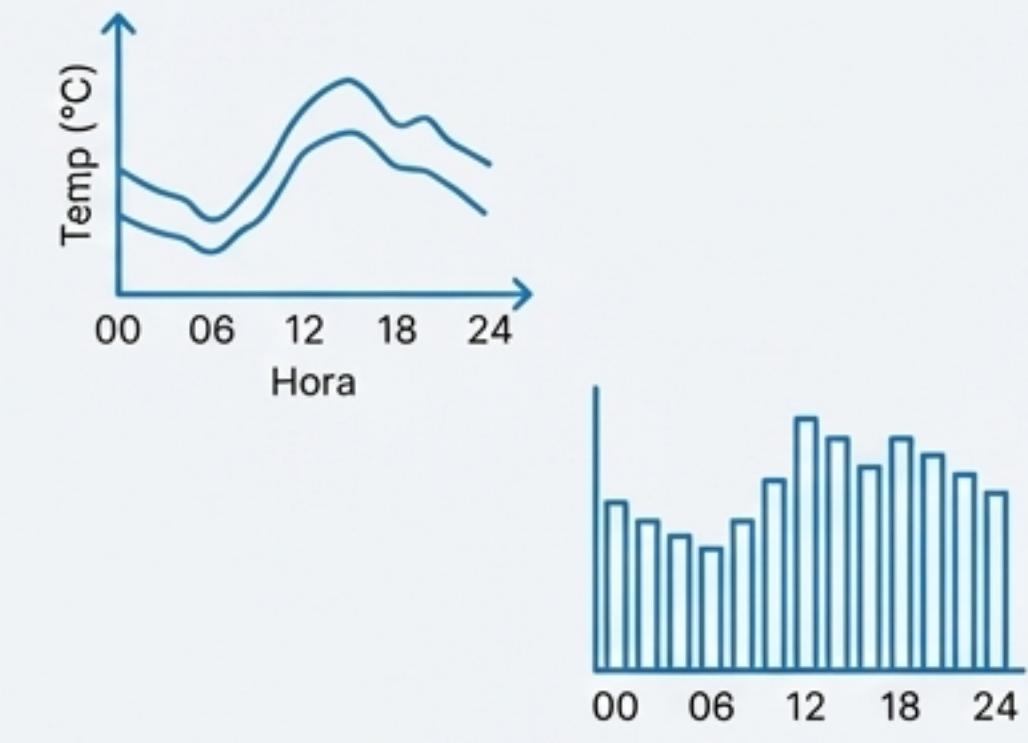
- El objetivo es proporcionar datos pre-agregados y optimizados para responder a preguntas de negocio específicas.

Agregaciones Realizadas:

- **Métricas Horarias:** Calculamos la temperatura media, máxima y mínima por cada estación y hora.
- **Métricas Diarias:** Agregamos los datos para obtener resúmenes por día.

Optimización para Consultas:

- Los datos se guardan en formato Delta, particionados por fecha (año, mes, día).
- Esto acelera drásticamente las consultas en herramientas de BI, que no necesitan escanear toda la tabla para obtener los datos de un día concreto.



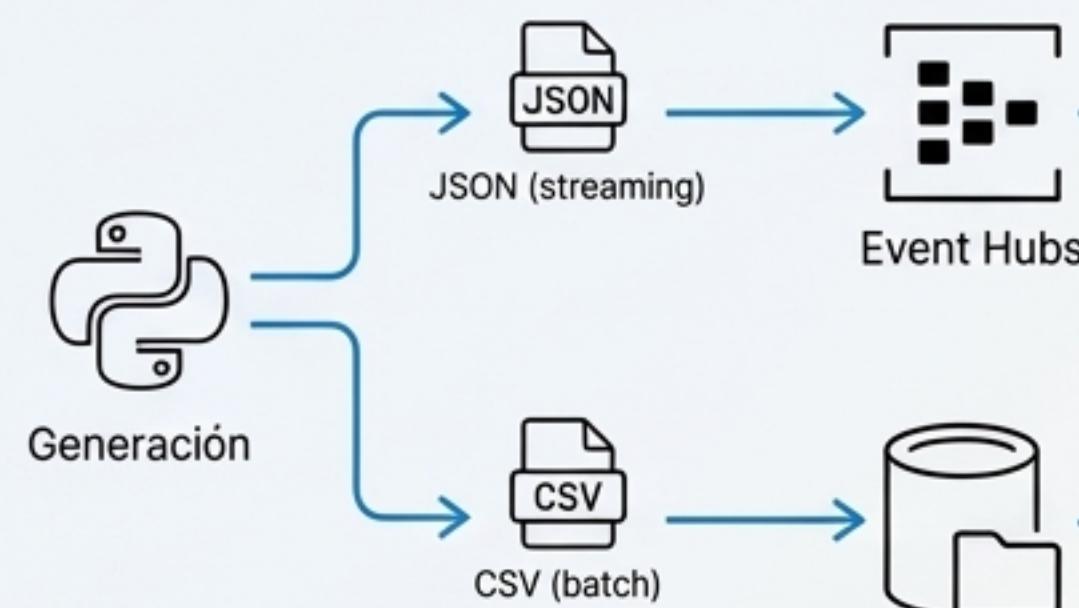
Gold Data			
hora	temp_media	temp_max	temp_min
00	15.5	18.0	13.0
06	12.0	14.0	10.0
12	22.5	25.0	20.0
18	19.0	21.0	17.0

El Flujo de Datos Completo, Paso a Paso

1 Generación: Scripts Python crean datos JSON (streaming) y CSV (batch).

2 Ingesta:

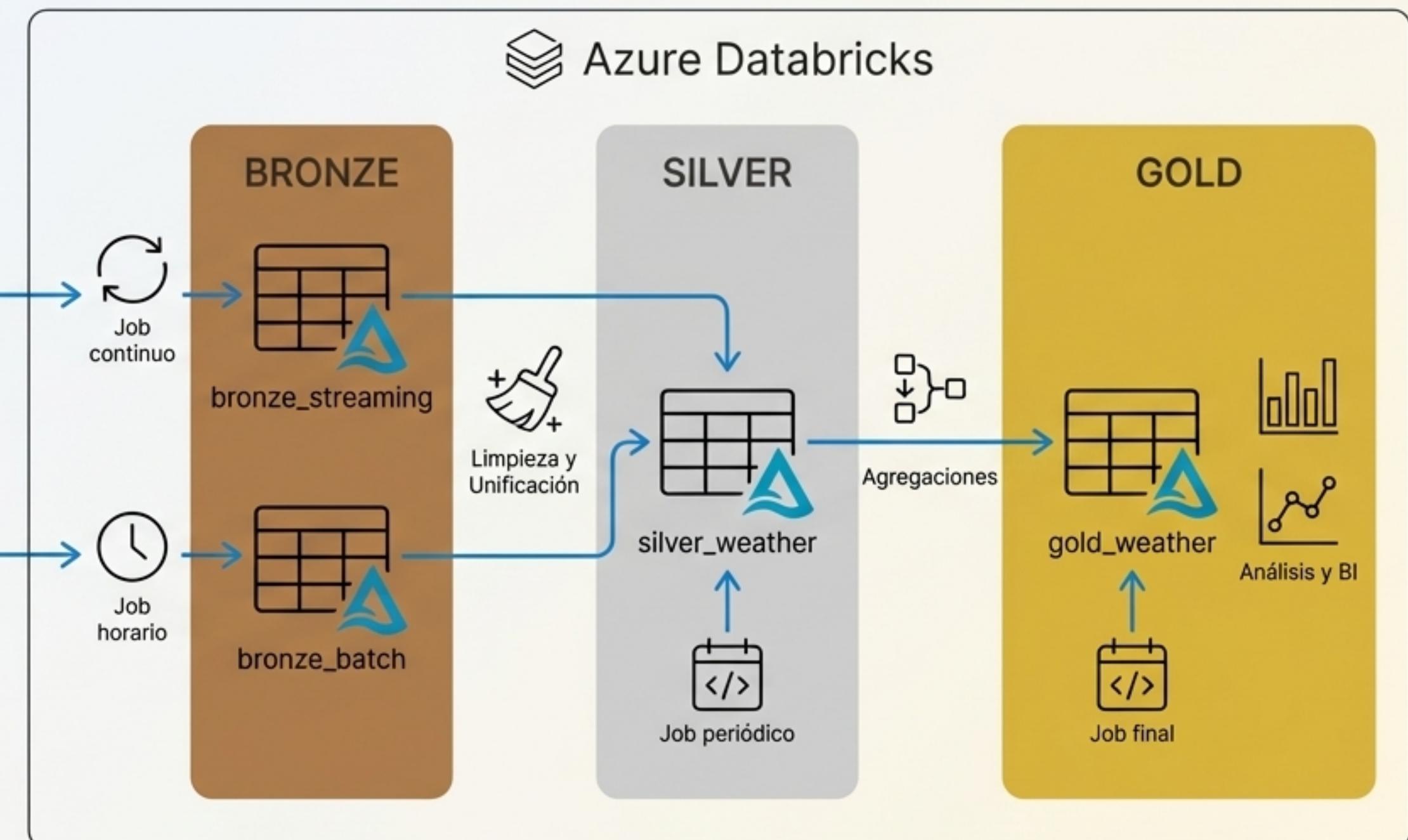
- JSON → Event Hubs en tiempo real.
- CSV → ADLS Gen2 '/landing' cada hora.



3 Procesamiento BRONZE (Databricks):
Jobs continuo leen de Event Hubs y ADLS Gen2 /landing, las escriben en las tablas 'bronze_streaming' y 'bronze_batch'.

4 Procesamiento SILVER (Databricks):
Jobs periódicos leen de ambas tablas Bronze, las limpian, unifican y escriben en la tabla 'silver_weather'.
Job horario

5 Procesamiento GOLD (Databricks):
Un job final lee de Silver, calcula agregaciones horarias/diarias y escribe en las tablas 'gold_weather' optimizadas.



La Lógica Detrás del Flujo: Los Notebooks Clave

- El pipeline se implementa a través de cuatro notebooks principales en Databricks.

“streaming_consumer_event_hubs”

- **Función:** Se conecta a Event Hubs y escribe el flujo de datos crudos en la capa Bronze (streaming).

“bronze_to_silver_streaming_weather”

- **Función:** Lee de Bronze (streaming), aplica limpieza y escribe en Silver en tiempo real.

“bronze_to_silver_batch_weather”

- **Función:** Lee los datos de Bronze (batch), aplica limpieza y escribe en Silver.

“silver_to_gold_weather”

- **Función:** Lee de la capa Silver unificada y genera las agregaciones que conforman la capa Gold.

Orquestación Automatizada con Databricks Jobs

Usamos Databricks Jobs para orquestar la ejecución de los notebooks y crear un pipeline 100% automatizado.

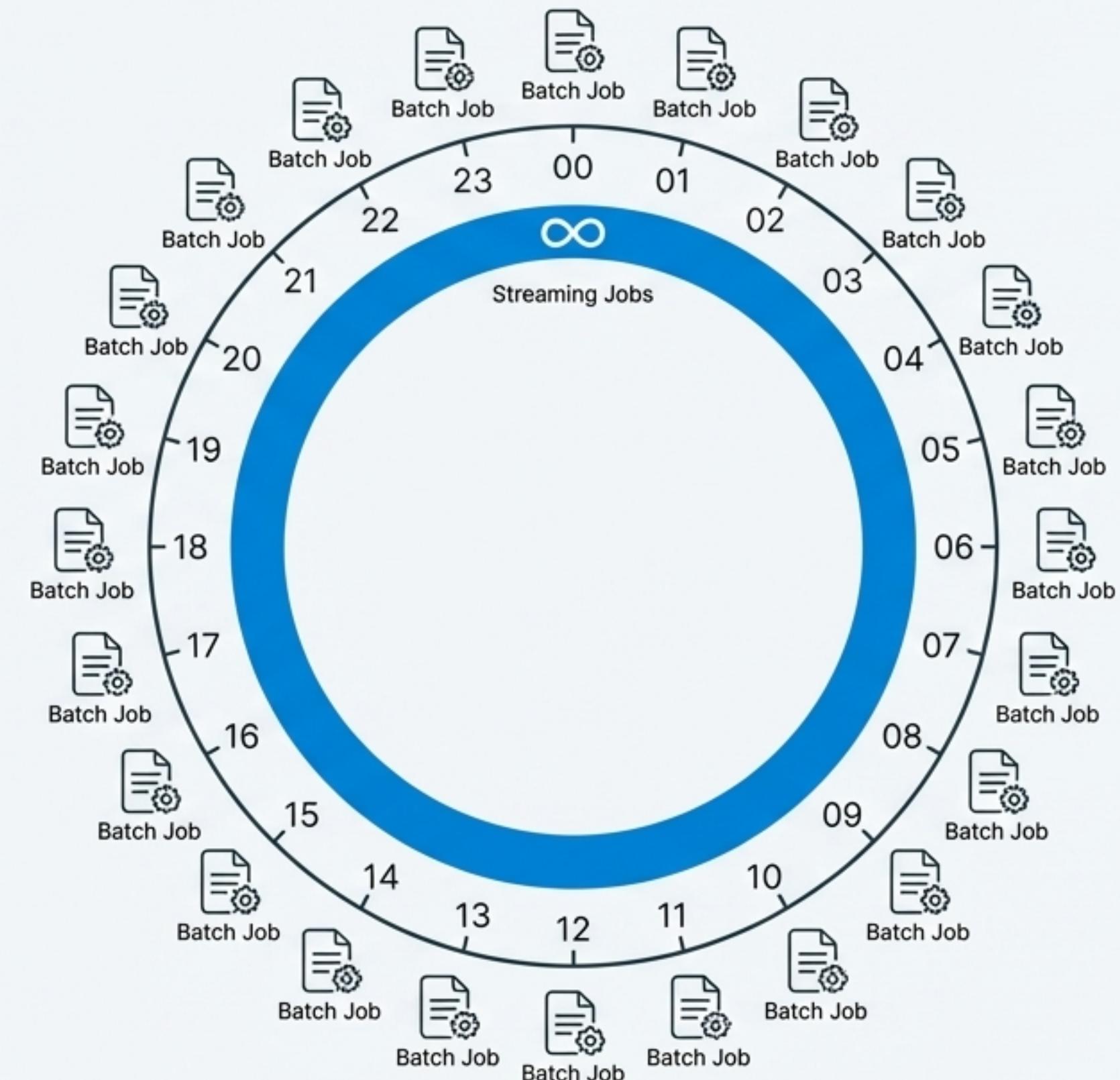
Jobs de Streaming (Ejecución Continua):

- `job_streaming_eventhubs_bronze`: Consumo de Event Hubs a Bronze.
- `job_streaming_bronze_to_silver`: Transforma de Bronze a Silver en tiempo real.

Job de Batch (Programado cada hora):

- `job_batch_etl_weather`: Ejecuta dos tareas en secuencia:
 1. `bronze_to_silver_batch`
 2. `silver_to_gold` (depende de la anterior)

Esto garantiza que el flujo completo permanezca sincronizado y actualizado sin intervención manual.



El Resultado: Una Arquitectura Lakehouse Moderna

Nuestra arquitectura final es un Lakehouse, que combina la flexibilidad y bajo coste de un Data Lake con la fiabilidad y el rendimiento de un Data Warehouse.

¿Por qué es útil?

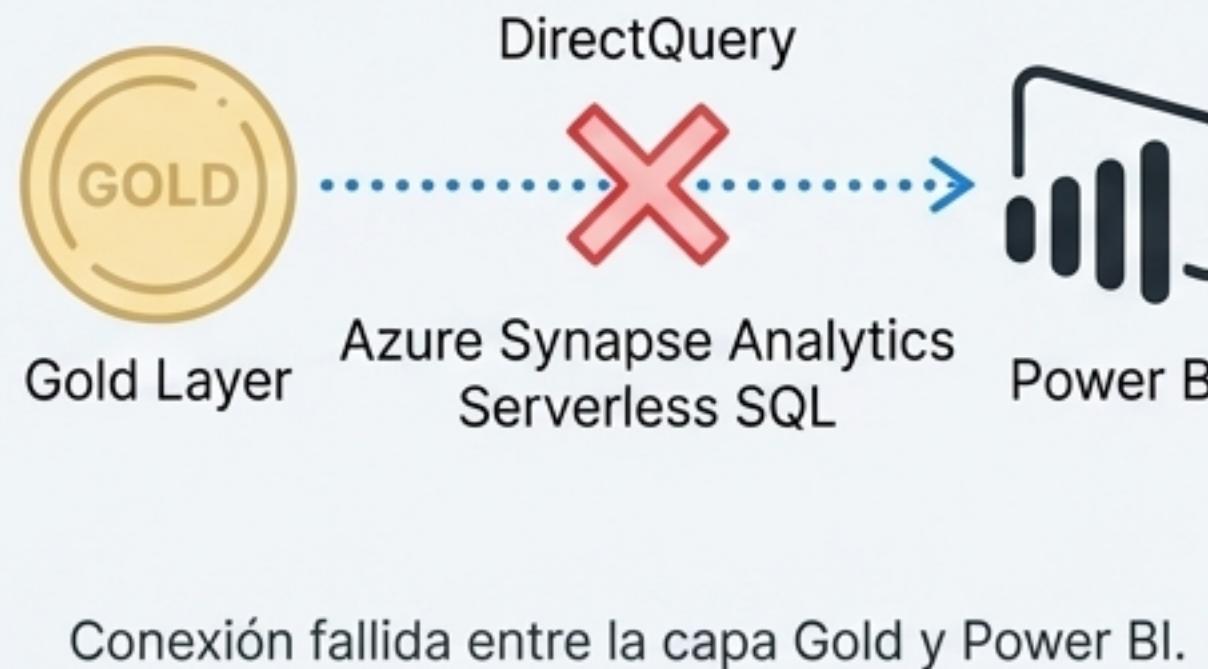
- **Flexibilidad:** Permite almacenar todo tipo de datos (estructurados, no estructurados) a bajo coste.
- **Fiabilidad:** Gracias a Delta Lake, obtenemos características de bases de datos tradicionales.



Ventajas Clave de Delta Lake:

- **Transacciones ACID:** Garantizan que las operaciones o se completan o fallan, evitando la corrupción de datos.
- **Versionado (Time Travel):** Permite consultar versiones anteriores de los datos, ideal para auditorías o corregir errores.

El Último Paso: La Conexión con Power BI (Plan y Desafíos)



- **Plan Original:**

- Conectar Power BI a la capa Gold usando Azure Synapse Analytics Serverless SQL como puente, permitiendo consultas en modo 'DirectQuery'.

- **El Desafío Técnico:**

- Se encontraron problemas persistentes de autenticación entre Power BI Desktop y Synapse SQL mediante Azure Active Directory.
- A pesar de tener los permisos correctos, Power BI devolvía errores de inicio de sesión (OAuth), impidiendo completar la conexión.

- **Opciones Futuras (la arquitectura es flexible):**

- Resolver el problema de autenticación con Synapse.
- Utilizar Microsoft Fabric (Lakehouse o Warehouse).
- Exportar la tabla Gold a una Azure SQL Database.
- Conectar Power BI directamente al Data Lake (ADLS Gen2).

Conclusiones y Logros del Proyecto

¿Qué se ha conseguido?

- ✓ Un **flujo de datos híbrido** y completamente funcional que maneja ingestas en tiempo real (streaming) y por lotes (batch) de forma simultánea.
- ✓ Una **arquitectura Lakehouse profesional** con el patrón Medallion, garantizando la calidad y gobernanza de los datos de extremo a extremo.
- ✓ Un pipeline **100% automatizado** mediante Databricks Jobs, creando un sistema robusto, de bajo mantenimiento y listo para producción.

El Resultado Final:

- ✓ Un sistema **escalable y extensible**, con una base de ingeniería de datos sólida, preparada para conectar herramientas de BI y analítica avanzada en el futuro.