Un poco más sobre el capítulo 1 del libro de Strengholt

Juamini 3 Pro Preview

November 23, 2025

Para que te vaya bien, no veas esto como una lista de definiciones, sino como una **historia de evolución**. La tecnología de datos cambió porque surgieron **problemas** que la tecnología anterior no podía resolver.

Aquí tienes la explicación conceptual, atando cabos para que entiendas el **porqué** de cada cosa.

La Gran Imagen: ¿Cómo se estructura una Plataforma de Datos?

Antes de hablar de historia, el autor te da el "mapa mental" de cómo se ve cualquier arquitectura de datos moderna. Imagínatelo como una fábrica.

1. El Diseño de Tres Capas (The 3-Layer Design)

No importa si usas tecnología de 1990 o de 2025, siempre hay tres partes:

1. **Proveedores (Input):** La materia prima. Bases de datos de apps, sensores IoT, archivos externos. Son un caos.

2. **Distribución (El Proceso):** La maquinaria. Aquí es donde ocurre la magia de limpiar y transformar.

3. **Consumidores (Output):** El cliente final. Puede ser un humano (mirando un Dashboard de PowerBI) o una máquina (un algoritmo de ML haciendo predicciones).

El "Techo": Por encima de todo esto está la **Capa de Gobernanza y Metadatos**. No es un paso más, es el "gerente" que vigila que nada se rompa, que los datos sean seguros y que sepamos qué hay en cada lugar.

El Objetivo Final: La Arquitectura Medallion

El autor te dice: *"Mira, hoy en día, la mejor forma de organizar esa 'Capa de Distribución' es usando el patrón Medallion"*. ¿Por qué? Porque pone orden en el caos.

Imagina la calidad del dato como el agua:

1. **Bronze (Crudo):** Agua del río. Sucia, con barro, pero es **toda** el agua que hay. No se tira nada. Es tu copia de seguridad histórica.

2. **Silver (Limpio):** Agua filtrada. Le quitaste el barro (limpieza), mataste bacterias (validación) y quitaste duplicados. Ya se puede usar, pero quizás no está embotellada para vender.

3. **Gold (Negocio):** Agua embotellada y etiquetada. Está lista para consumo masivo. Aquí los datos están agregados (ej: "Ventas totales por mes") para que el gerente los lea rápido.

La Historia: ¿Cómo llegamos hasta aquí?

Para entender por qué usamos Medallion y Lakehouse hoy, tienes que entender qué falló antes.

Era 1: El Data Warehouse (El Almacén Rígido)

En los 90s, las empresas querían informes. Inventaron el Data Warehouse.

- **La Filosofía:** Orden absoluto. Antes de guardar un dato, tienes que definir exactamente dónde va (Tablas, Columnas).

- **El Problema Técnico:** Las bases de datos de las aplicaciones (OLTP) están hechas para escribir rápido transacción por transacción. Si un analista entra a hacer consultas gigantes, tumba el sistema.

- **La Solución:** Mover los datos a un **Data Warehouse (OLAP)**, diseñado para leer rápido.

Conceptos de Examen (Era DW):

1. **Normalización vs. Desnormalización:**

- *Normalizar* es dividir todo en muchas tablas pequeñas para no repetir datos (bueno para sistemas operativos).

- *Desnormalizar* es juntar todo en tablas grandes y repetitivas para leer rápido (bueno para Analytics/Warehouse).

2. **Inmon vs. Kimball:**

   - *Inmon:* Construye un almacén gigante y centralizado primero (Top-down). Es lento y caro de hacer.

   - *Kimball:* Construye pequeños almacenes por departamento (Data Marts) usando el **Modelo Estrella** (Tablas de Hechos y Dimensiones). Es más rápido de implementar.

3. **SCD (Dimensiones Cambiantes):** ¿Qué pasa si un cliente se muda?

   - *SCD1:* Sobrescribes el dato. Pierdes la historia. (Malo para análisis histórico).

   - *SCD2:* Creas una fila nueva y marcas la vieja como "pasada". (El estándar de oro, guardas toda la historia).

¿Por qué murió esta era? Porque llegó el Internet y el Big Data. Los Warehouses eran carísimos y no podían guardar videos, logs o textos libres. Eran demasiado rígidos.

Era 2: El Data Lake (El Pantano Caótico)

En los 2000, Google y Yahoo crearon **Hadoop**. La idea era: *"Guardemos todo en hardware barato y preocupémonos después"*.

- **La Tecnología:**

  - **HDFS:** El disco duro distribuido (guarda archivos en muchas máquinas baratas).

  - **MapReduce:** La forma de procesar (lenta, escribía mucho en disco).

  - **Hive:** Puso SQL encima de Hadoop para que no fuera tan difícil de usar.

- **Spark:** Reemplazó a MapReduce porque procesa en memoria (RAM) y es 100 veces más rápido.

El Concepto Clave (Examen): Schema-on-Read

A diferencia del Warehouse (que te obliga a definir la tabla antes de escribir), en el Lake tiras los archivos y defines la estructura **recién cuando los lees**.

- *Ventaja:* Flexibilidad total.
- *Desventaja:* **Data Swamp**. Si tiras basura, cuando vas a leer, encuentras basura. Además, no tenía transacciones (ACID). Si fallaba una carga, te quedaban archivos corruptos.

El problema de los "Dos Niveles": Como el Data Lake era barato pero lento y desordenado, las empresas terminaron teniendo los dos: Un Data Lake para guardar todo + un Data Warehouse para los reportes importantes. **Mantener esto sincronizado era una pesadilla.**

Era 3: El Lakehouse (La Convergencia)

Aquí es donde entra **Databricks** y **Delta Lake**. Se preguntaron: *¿Por qué no le damos al Data Lake (barato) los superpoderes del Data Warehouse (confiable)?*

El Habilitador Técnico: Delta Lake

Delta Lake es lo que hace posible el Lakehouse. ¿Cómo? Agrega un **Registro de Transacciones (Transaction Log)** a los archivos del Data Lake.

1. **Transacciones ACID:** Ahora el Lake sabe si una escritura terminó bien o mal. No más datos corruptos.
2. **Time Travel:** Como guarda un historial de cambios en el log, puedes consultar los datos "como estaban ayer".
3. **Schema Enforcement:** Aunque son archivos, Delta Lake impide que guardes datos con el formato incorrecto.

Conclusión: ¿Por qué Medallion + Lakehouse?

El libro cierra uniendo todo.

- Usamos **Lakehouse** (tecnología) para tener una sola plataforma barata y rápida.
- Sobre ella, aplicamos la arquitectura **Medallion** (diseño) para ordenar los datos progresivamente (Bronze -> Silver -> Gold).

Glosario Rápido para el Examen ("Cheat Sheet")

- **ETL:** Extraer, Transformar, Cargar (Modelo Warehouse clásico).
- **ELT:** Extraer, Cargar, Transformar (Modelo moderno/Lakehouse: cargo crudo al Bronze y transformo después).
- **OLTP:** Base de datos de la App (rápida para transacciones, mala para análisis).
- **OLAP:** Base de datos Analítica (rápida para leer mucho datos).
- **SCD2:** La mejor forma de guardar historia (crear filas nuevas para cambios).
- **Schema-on-Read:** Definir la estructura al leer (típico de Data Lake).
- **Data Swamp:** Un Data Lake sin gobernanza ni calidad.
- **Delta Log:** El cerebro de Delta Lake que permite transacciones ACID en archivos.

Si entiendes que pasamos de **"Orden Rígido y Caro" (Warehouse)** a **"Caos Barato" (Lake)** y finalmente a **"Orden Flexible y Eficiente" (Lakehouse)**, tienes el capítulo dominado.