

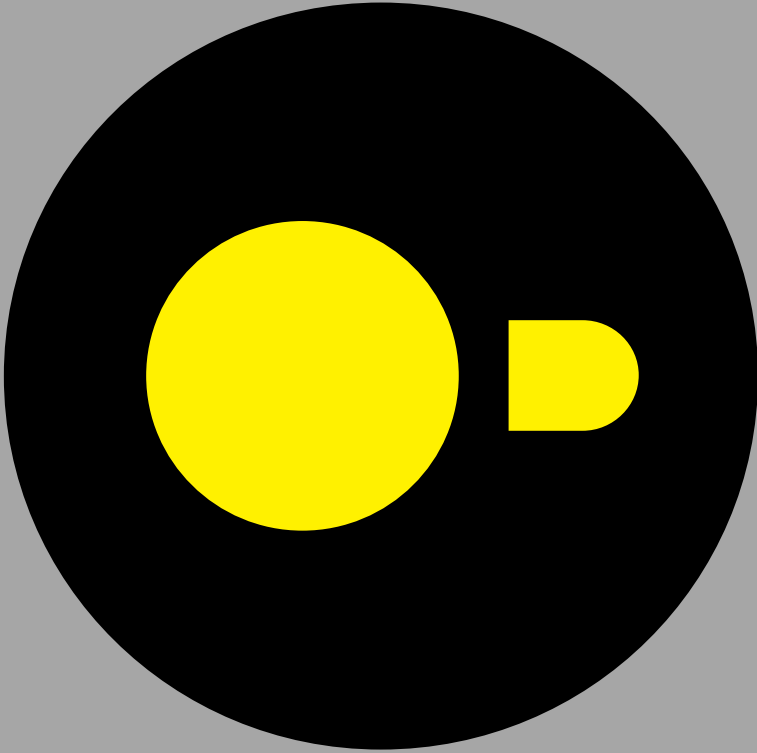
Rioja{dotnet}

DuckDB

Una base de datos analítica in-process con muchas posibilidades



plain
concepts



DuckDB

Una base de datos analítica
in-process con muchas
posibilidades

Eladio Rincón Herrera



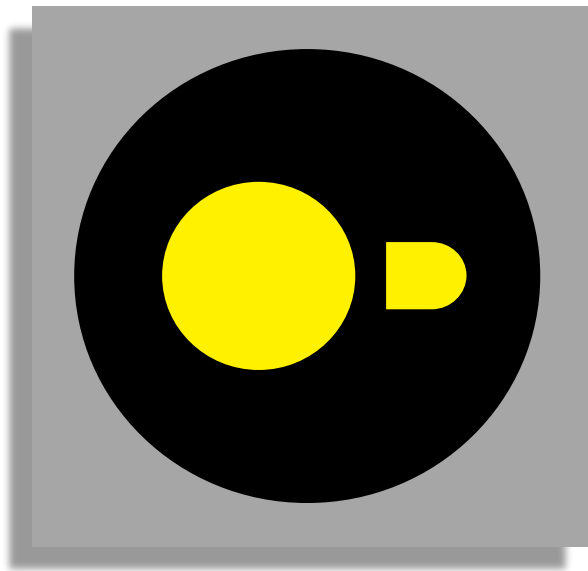
<https://www.linkedin.com/in/erincon/>



<https://github.com/erincon01>



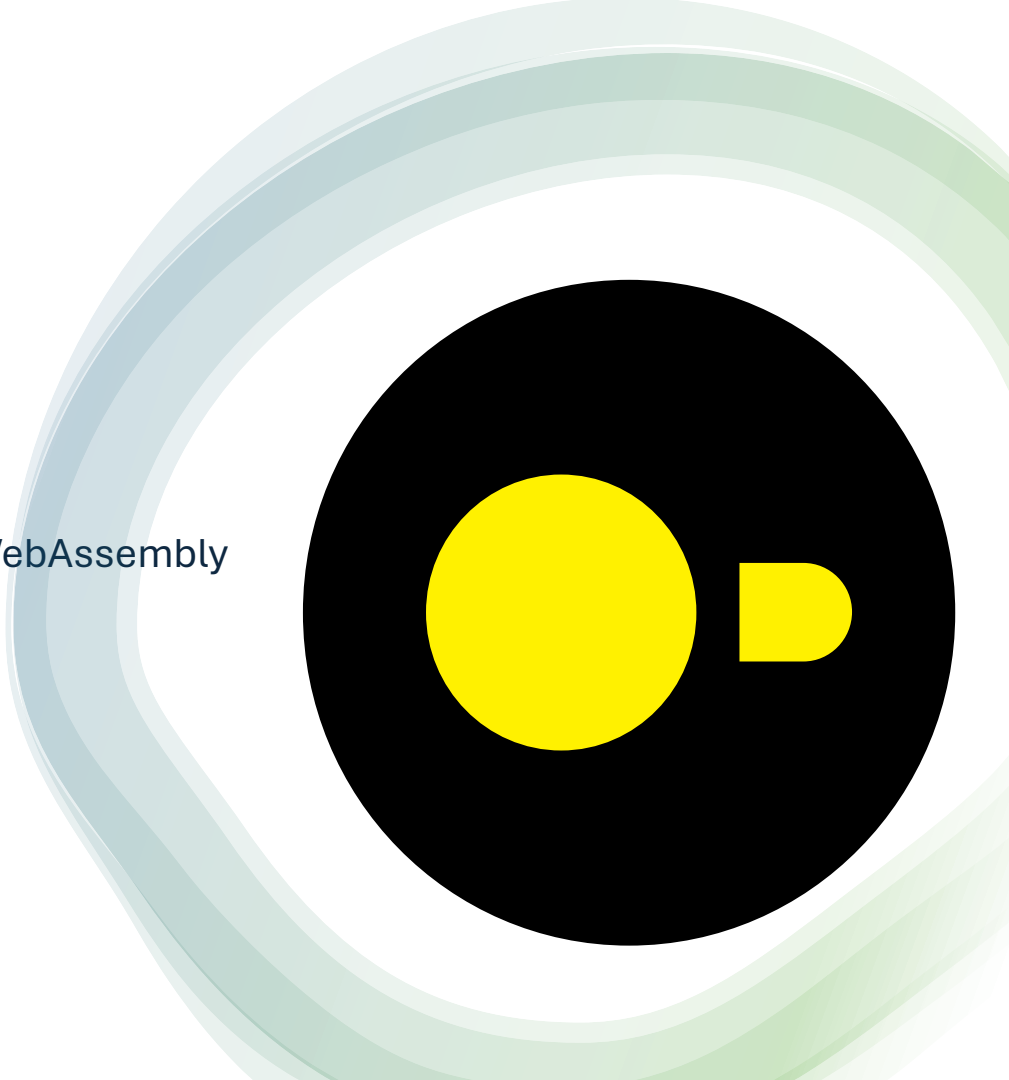
Agenda



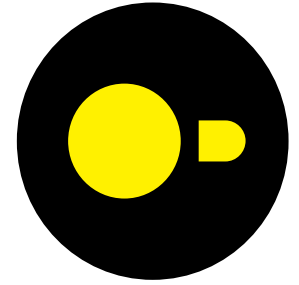
- Qué es DuckDB
- Historia
- Factores ganadores
- Casos de uso:
 - Notebooks
 - DuckDB Cli
 - Azure
 - Fabric

Qué es DuckDB

- SQLite para datos analítico embebido
- Diseñado para consultas OLAP en local
- Sin servidor, embebido, sin dependencias
- Portable: Linux, Windows, mac, x86, arm, WebAssembly
- Extensible
 - azure, aws, excel, json, csv
 - iceberg, vss, csv, parket, arrow, delta
 - mysql, postgres
 - Vss [vectores], spatial
- No es para cargas OLTP



Historia

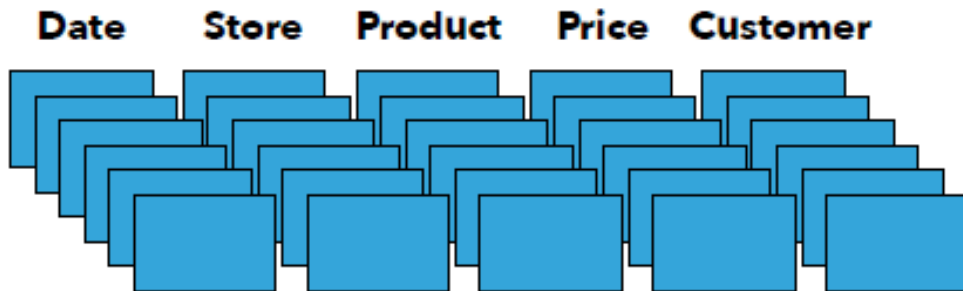




- 2019: Creado en la Universidad de Ámsterdam por Hannes Mühleisen y Mark Raasveldt
- Licencia MIT, inspirado en SQLite
- La clave:
 - Arquitectura columnar,
 - procesamiento vectorizado y
 - ejecución en local sin de servidor.
- Crecimiento rápido: [GitHub](#)
- Versión Cloud: [Motherduck](#) como DBaaS
- Conoce a [Hannes](#) [CEO]



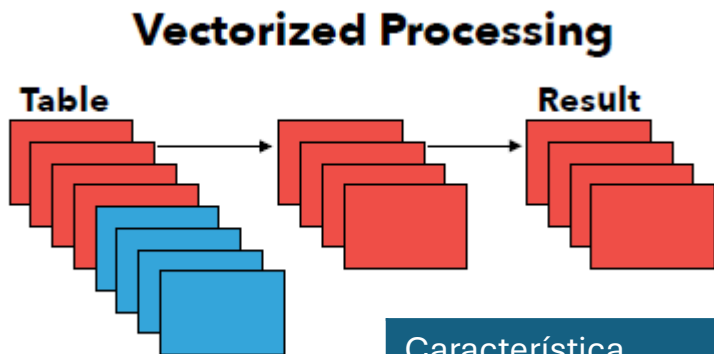
P1. Arquitectura columnar

Column-Store



Característica	DuckDB 	SQL Server 
Almacenamiento Columnar	✓ Sí, nativo	✓ Columnstore Index
Compresión	✓ Sí, optimizada para OLAP	✓ ROW, PAGE
Optimización de CPU	✓ Sí, vectorizado	✓ Batch Mode

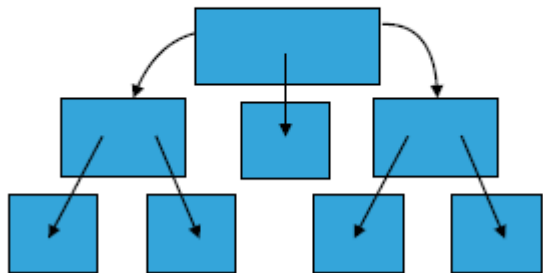
P2. Procesamiento vectorizado



Característica	DuckDB 🦆	SQL Server 🟦
Procesamiento Vectorizado	✅ Sí, nativo y siempre activado	✅ Sí, pero solo en Batch Mode
Tamaño de Bloque (Batch Size)	♦ 1024 valores por vector	♦ Hasta 900 filas en Batch Mode
Ejecución en Lotes (Batch Processing)	✅ Sí, por defecto en todas las consultas	✅ Sí, pero solo en cargas OLAP

P3. Índices ART

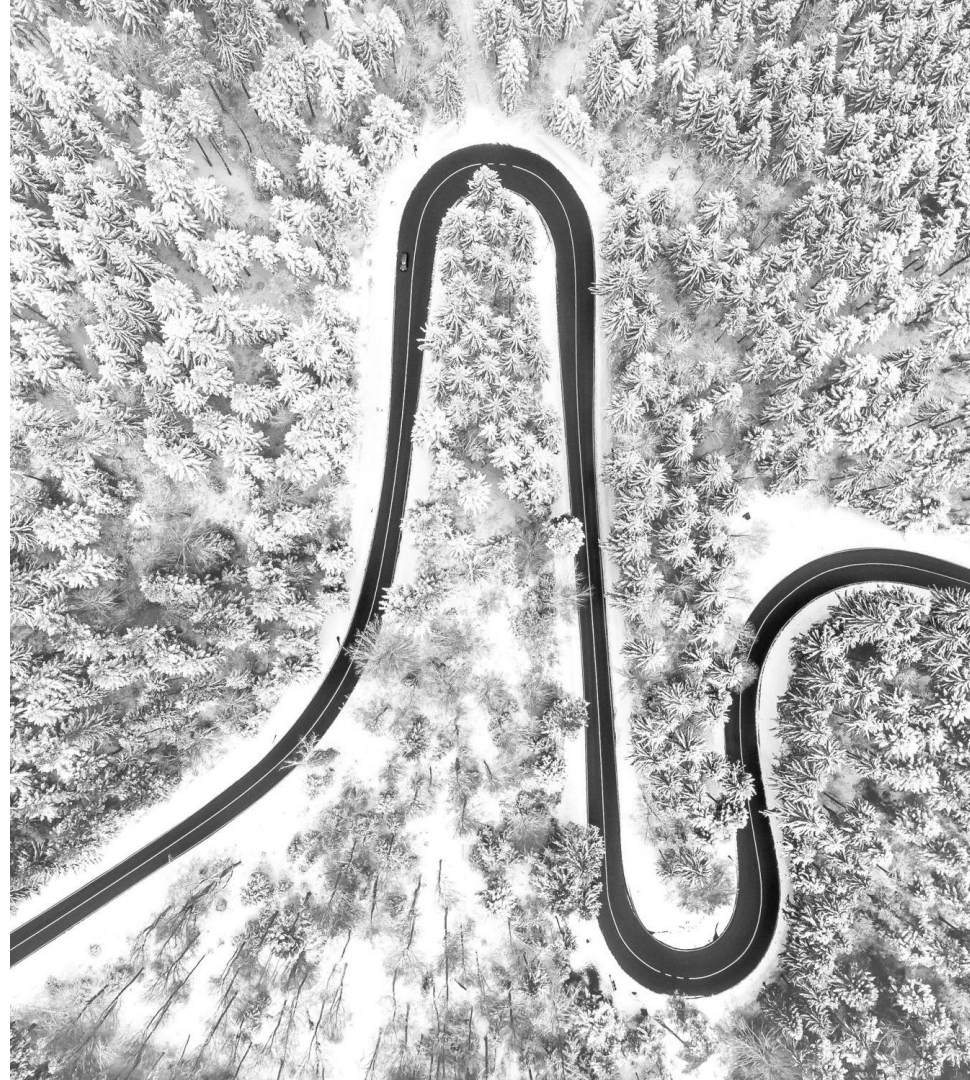
ART Index



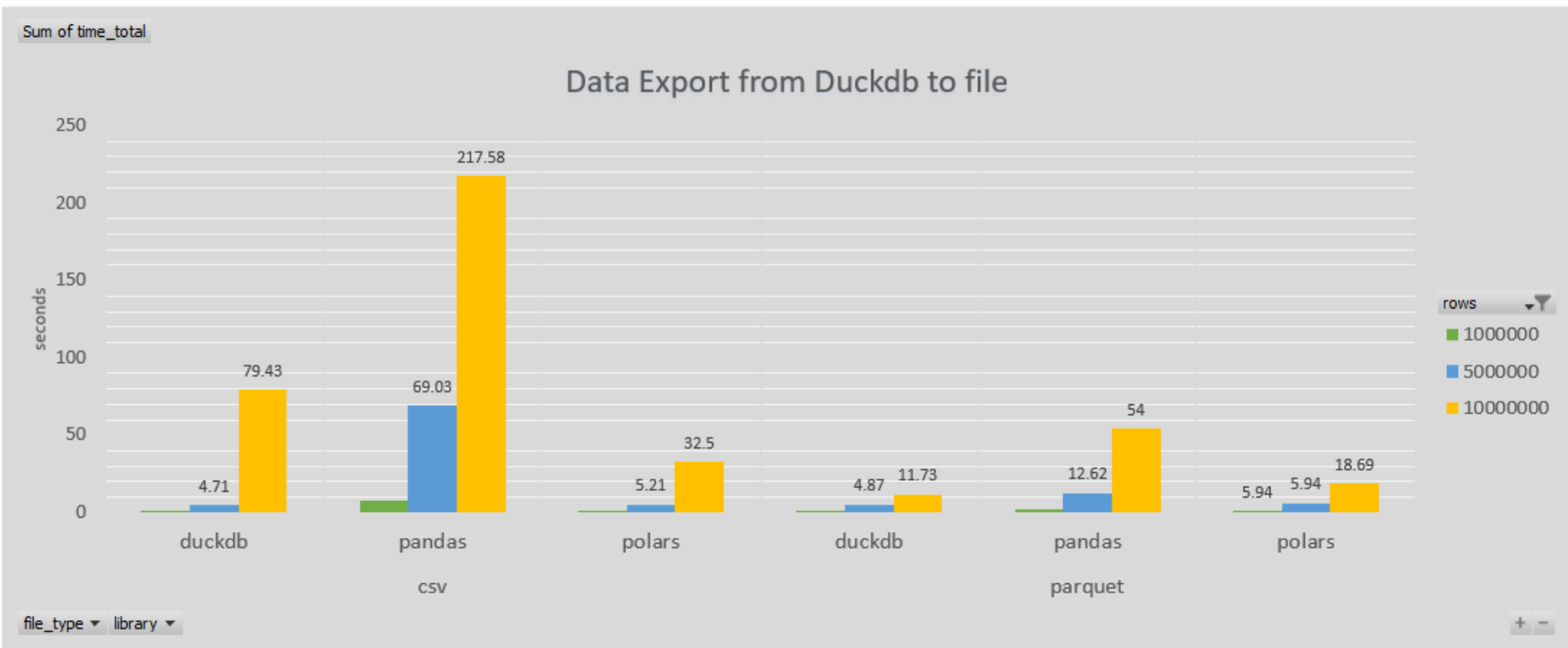
Característica	ART (Adaptive Radix Tree) 	B-Tree 
Velocidad de Búsqueda	✓ Más rápido en seeks al reducir comparaciones innecesarias.	✓ Bueno para búsquedas generales y de rango.
Estructura Adaptativa	✓ Se ajusta dinámicamente al tamaño de las claves	✗ Tamaño de nodo fijo, requiere reconstruir.
Uso de Memoria	✓ Menos punteros, más compacto en memoria.	✗ Mayor consumo de memoria debido a punteros y estructura fija.

Casos de uso: Procesar en Notebook

- Leer, contar, operar, transformar
- Agrupar, exportar
- Pandas vs polars

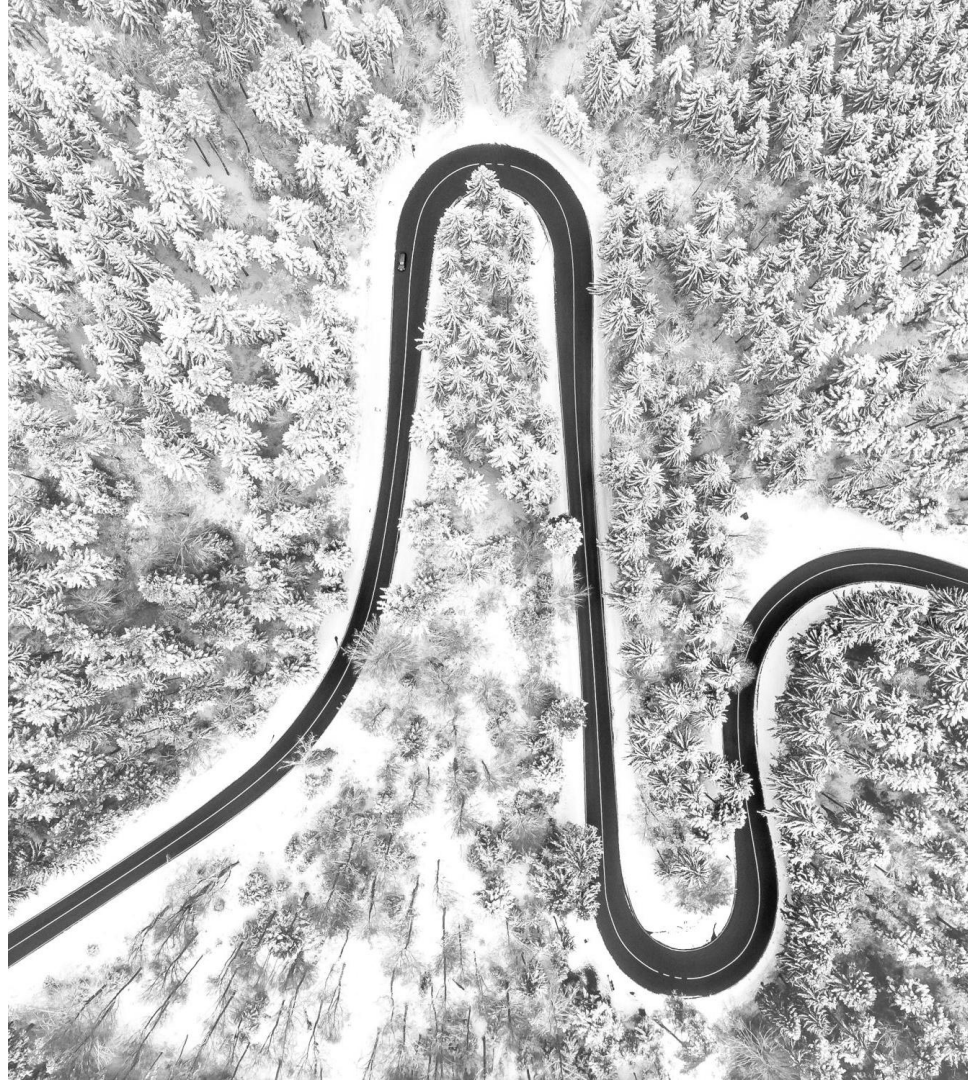


Transformación de datos



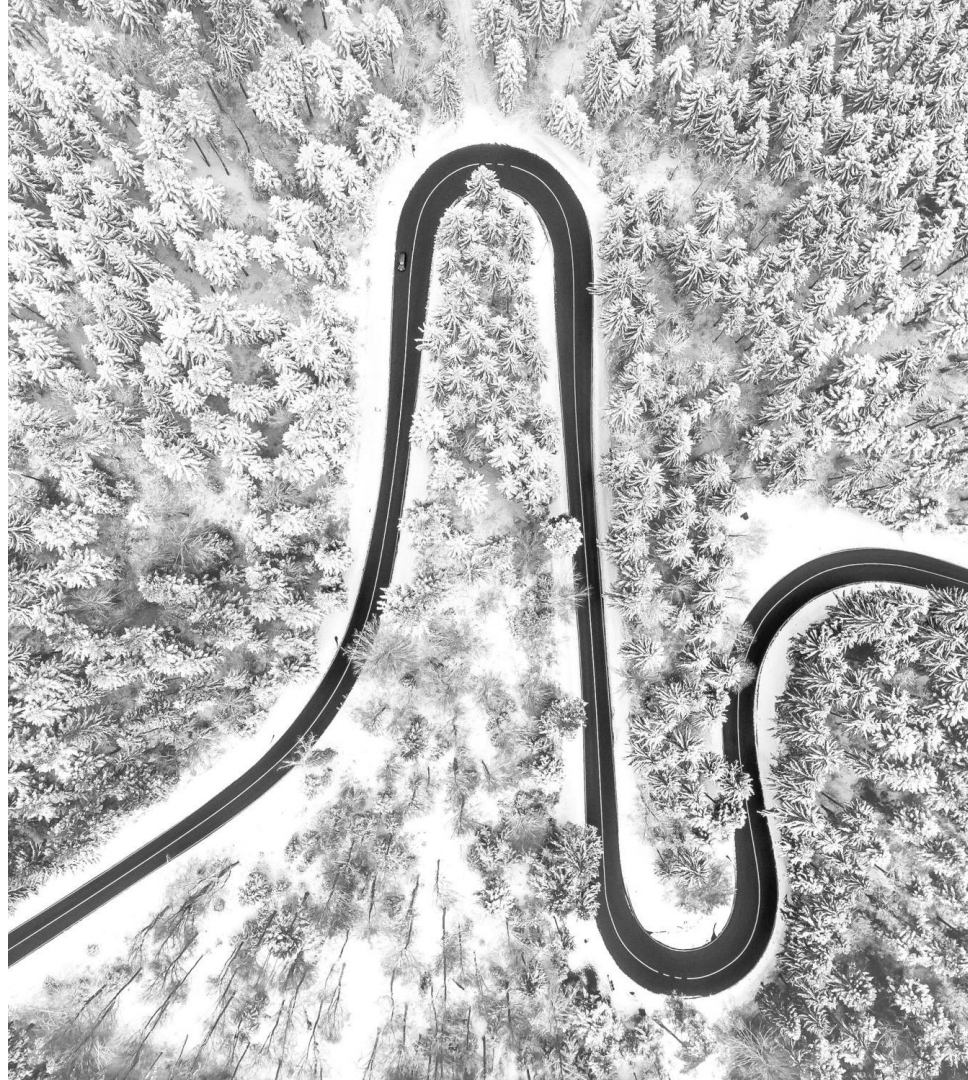
Casos de uso: datos desde Azure

- Azure
 - Autenticar, y bajar
 - Guardar, y transformar
 - Exportar, y volver a subir
- Cosas del lenguaje
 - GROUP BY ALL
 - PIVOT, UNPIVOT
 - FECHAS



Casos de uso: datos desde Fabric

- Coger datos de Fabric
 - Autenticar, y bajar
 - Guardar, y exportar
 - Volver a subir





your next steps...

