



Populate Your Data Warehouse with a Metadata-driven, Pattern-based Approach

Meagan Longoria
Consultant at DCAC

About Me



- Meagan Longoria
- Denver, CO
- Consultant at Denny Cherry & Associates Consulting
- Microsoft Data Platform MVP
- Blogger, Speaker, Author, Technical Editor
- Camper, dog owner

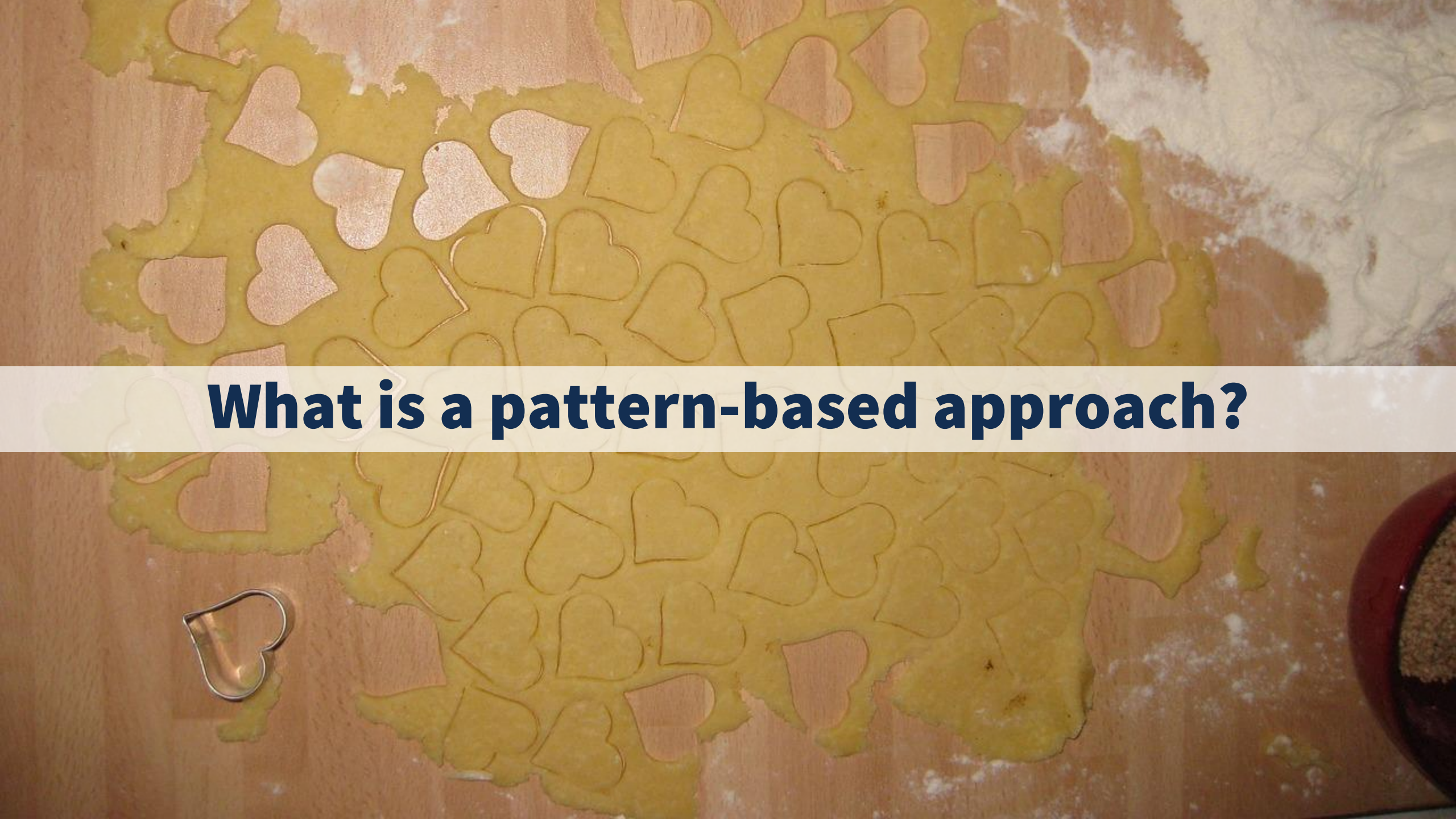


Intro





**Let's
define our
buzzwords**



What is a pattern-based approach?

Patterns



Instead of “Copy Table A from Database X to Database Y”



**Abstract away the details to create a template:
“Copy any table or query result from one database to another”**



Common data warehouse ETL patterns



**DW
Patterns**

Logging

Error Handling

Truncate & Reload

Incremental Load

SCD 1 Load

SCD2 Load

Fact Load

Restartability

Parallel processing

Benefits of a pattern-based approach



Consistency in development and implementation of best practices

Reduced required development time

Less tech debt, implement changes once

Reusable/reduced documentation

Reduced cognitive load for developers and support

Metadata: data that describes other data



File name: IzzyBeach.jpg

File type: JPG

File Size: 4.54 MB

Dimensions: 4640 x 3472

Author: Meagan

Date: 2022-08-23

Location: Long Beach, CA

Camera: Motorola Edge

...

Natural vs synthetic metadata



Natural - exists as an artifact of pre-existing processes and is machine readable and relatively correct.

Hybrid - Enhances natural metadata, usually through key/value pairs or other simple relationships

Synthetic - A largely standalone, sometimes complex metadata store that captures business logic and pattern selection



Common ETL metadata

File properties

Table schemas (name, columns and data types, etc.)

Self-describing web services

Data catalogs

Tags and annotations (e.g., sensitivity labels)

User-created tables and files



Metadata

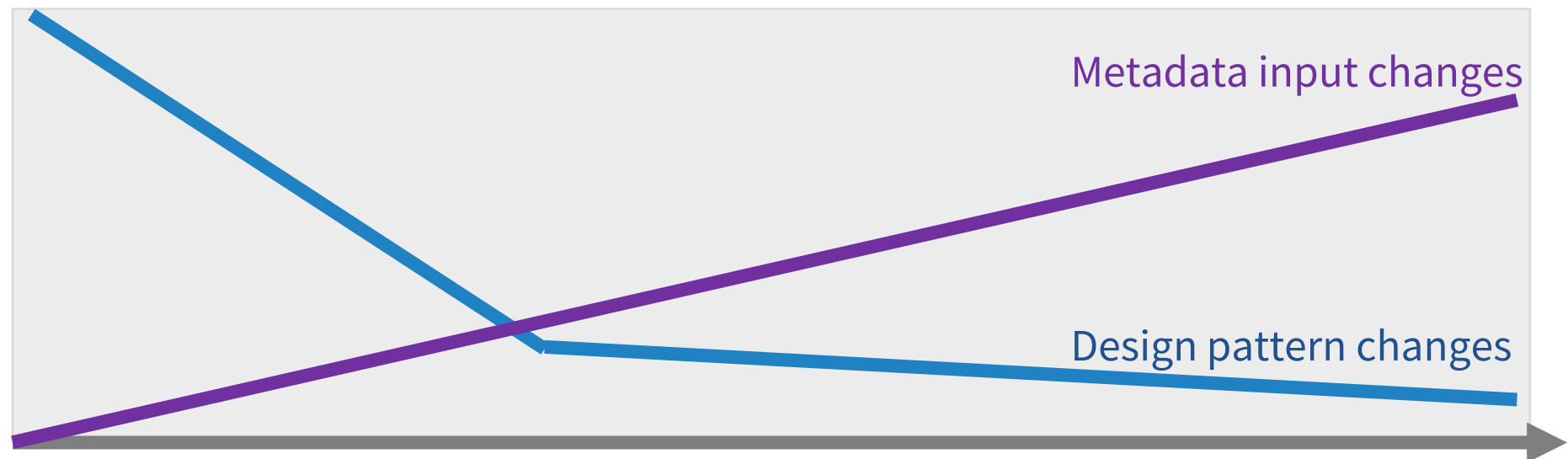
Benefits of metadata-driven ETL



Metadata can act as documentation

Mature implementations can involve data stewards providing synthetic metadata rather than the ETL developers

As time goes on, you rarely change your pattern and more frequently change your data





**Make it work with
Azure Data Factory
and Azure SQL**

Populating a data warehouse



Data Warehouse: Azure SQL DB

Data Orchestration and Copy: Azure Data Factory

Transformation Engine: SQL

Data Sources:

- Azure SQL Database
- Azure Data Lake Storage

The magic ingredients

Control table - determines what gets copied/executed

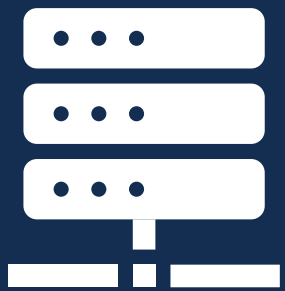
Log table – tracks execution, logs success/failure, execution times

ADF ForEach activity – Orchestrator for patterns, can control parallelism

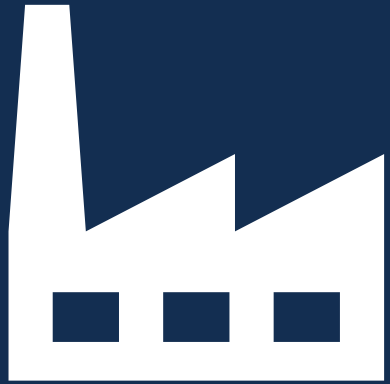
ADF Parameterized pipelines – Accept the metadata passed from the For Each loop



**Make it
work**



Demo/tour of my data mart



Demo/tour of my data factory



Final comments and questions

Tips



Don't make decisions you've already made before. Use patterns!

This works with other technologies, including notebooks and lakehouses.

Determine the appropriate level of abstraction for your projects.

Source control or back up your metadata.

Spend your extra time actually testing your ETL and data!

Automate SQL development using metadata.

Meagan Longoria

DCAC*

**Have a
question?**

Ask me!



Meagan@dcac.com



DataSavvy.me



@Mmarie (Twitter and Bluesky)



/in/meaganlongoria/