Best Practices of Implementing Azure Data Factory

Eddy Djaja, CDMP

www.sqlgrease.com

Technical Advisor, SQLGrease

Blogger: https://dbaworld.blogspot.com

Twitter: @edwdjaja

Email: eddy.djaja@gmail.com

Agenda

- Introduction to Azure Data Factory
- Key Concepts of ADF
- Best Practices for ADF Implementation
- Demos on Best Practices
- Additional Recommendations
- Conclusion

Introduction to Azure Data Factory

Overview of ADF

Different Flavors of ADF

ADF Stand Alone

Data Factory in Azure Synapse Analytics

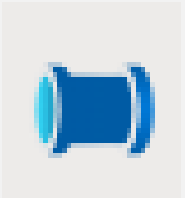
Data Factory in MS Fabric

Overview

- Azure Data Factory is a cloud-based data integration service.
- It allows for the creation, scheduling, and management of data pipelines.
- ADF connects various data sources for data movement and transformation.
- It supports both code-free and code-based development options.
- Provides a user-friendly interface for designing data workflows.

Different ADF Deployments

Standalone Data Factory operates independently for data integration tasks.



Synapse Data Factory integrates seamlessly within Azure Synapse Analytics.

Fabric Data Factory utilizes ADF capabilities in Microsoft Fabric environments.

Key Concepts of ADF

- Pipeline
- Dataset
- Link Services
- Integration Runtime

Best Practices for ADF Implementation

- Adopt Standard Naming Conventions
- Reusable Linked Services, Datasets and Pipelines
- Parameterize As Much As Possible
- Use and Standardize the Configuration Tables
- Integrate with Azure Key Vault
- CI/CD: GitHub and Azure DevOps

ADF Development Naming Standards

Use	Use clear and descriptive names for pipelines.
Implement	Implement consistent naming conventions for datasets.
Avoid	Avoid using special characters in names.
Use	Use separator/prefixes to categorize resources effectively.
Keep	Keep names concise yet informative.

Example of ADF Naming Standard

Type	Naming	Linked Service	Dataset
ADLS Gen2	ADLS_	LS_ADLS_MI	DS_ADLS_MI
Azure SQL Database	ASQL_	LS_ASQL_MI	DS_ASQL_MI
Azure SQL Database	ASQL_	LS_ASQL_SQLAUTH	DS_ASQL_SQLAUTH
MS SQL	MSSQL_	LS_MSSQL_WINAUTH	DS_MSSQL_WINAUTH
MS SQL	MSSQL_	LS_MSSQL_SQLAUTH	DS_MSSQL_SQLAUTH
SFTP	SFTP_	LS_SFTP_BASIC	DS_SFTP_BASIC
SFTP	SFTP_	LS_SFTP_BASIC_FP	DS_SFTP_BASIC_FP
SFTP	SFTP_	LS_SFTP_SSH	DS_SFTP_SSH

Parameterization in ADF

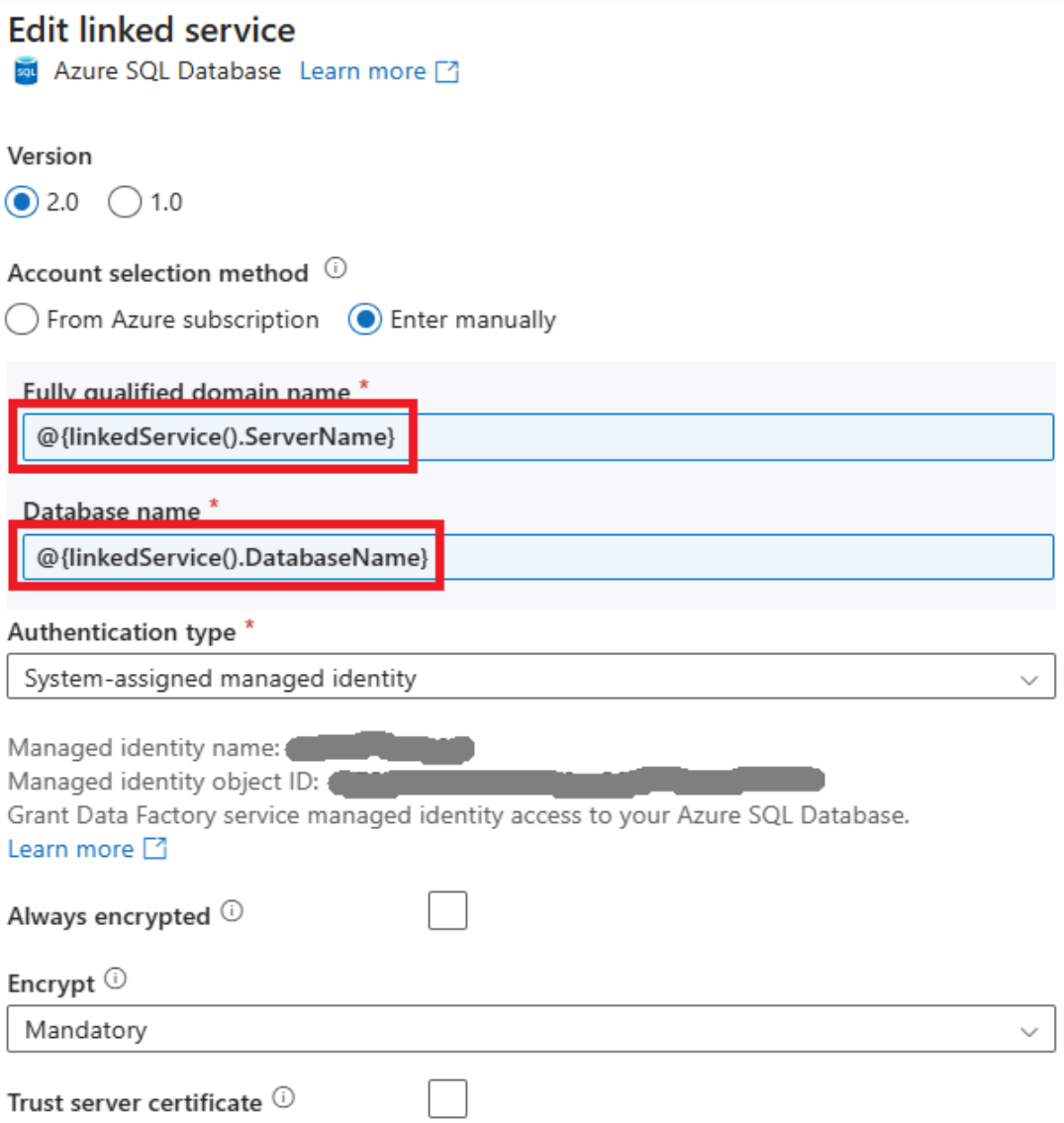
Promotes reuse of components across multiple pipelines.

Simplifies management of linked services and datasets.

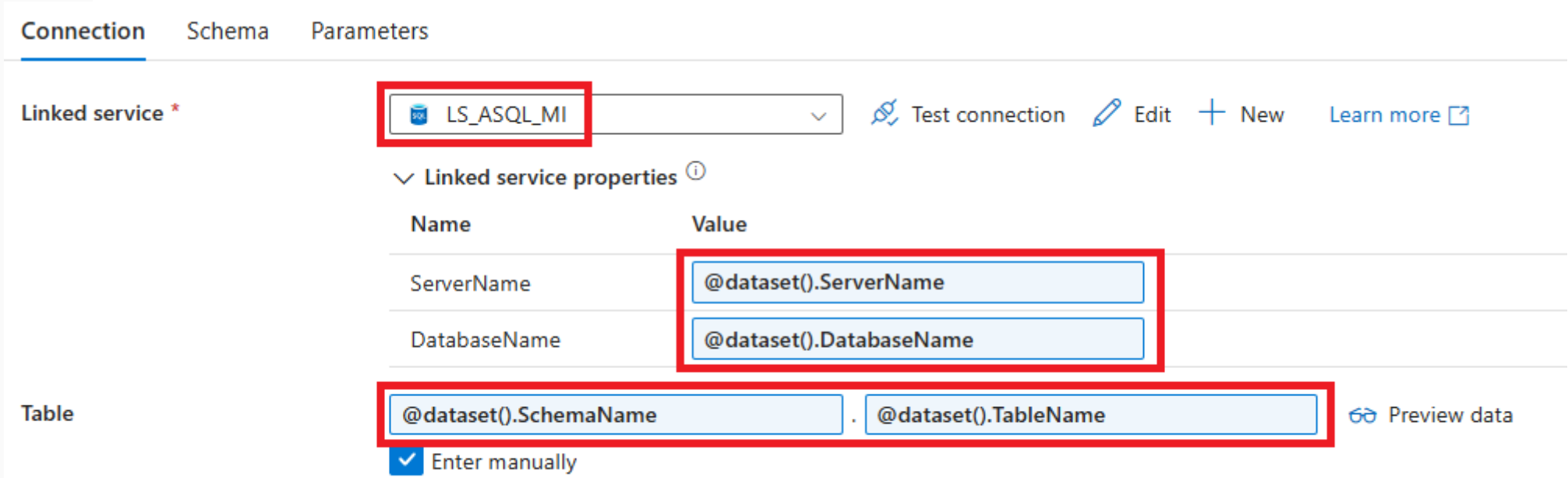
Enhances flexibility by allowing dynamic values.

Facilitates easier updates without extensive code changes.

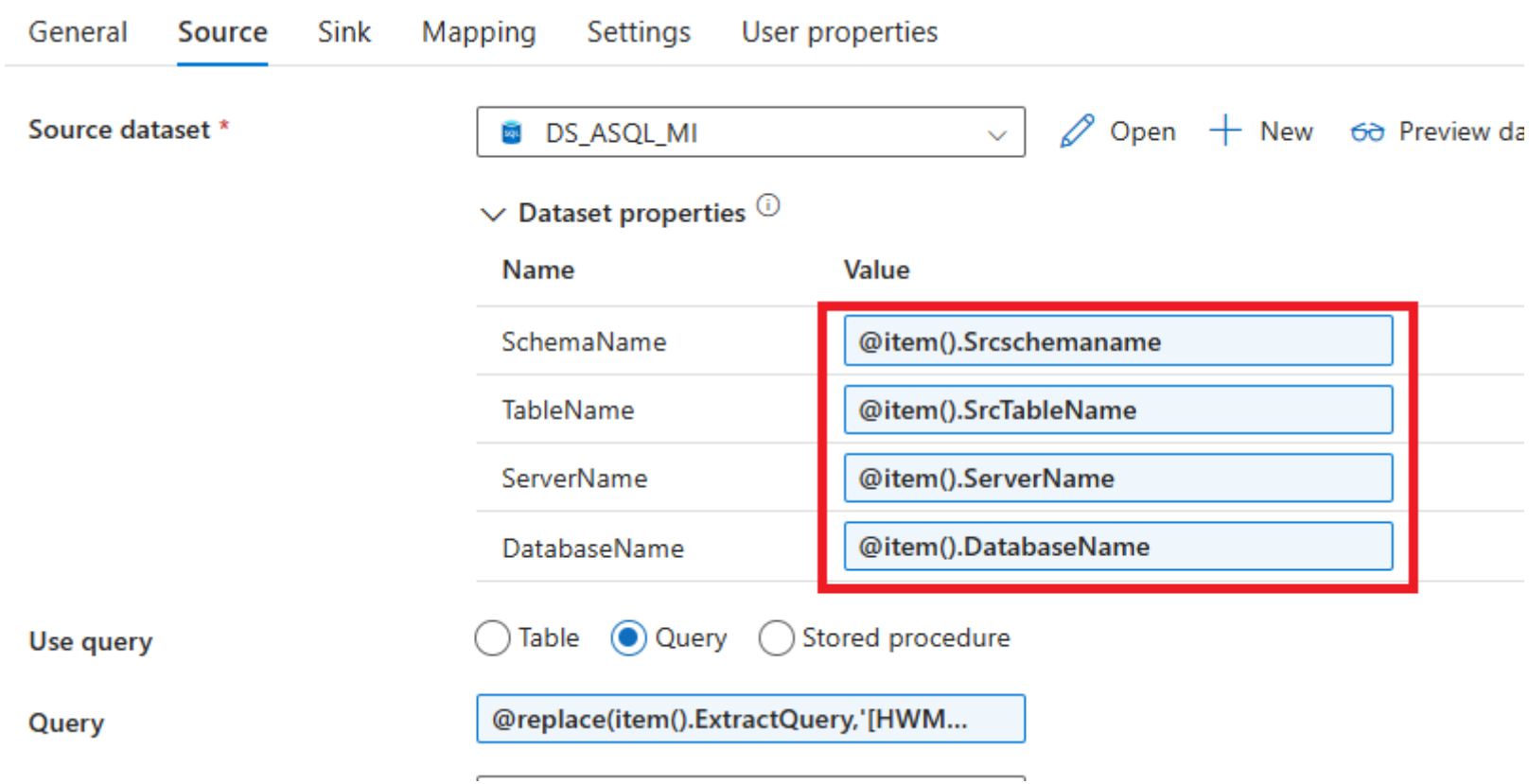
Parameterize Link Services



Parameterize Dataset



Parameterize Pipeline



Benefits of Configuration Tables

Centralizes configurations for easy management and updates.

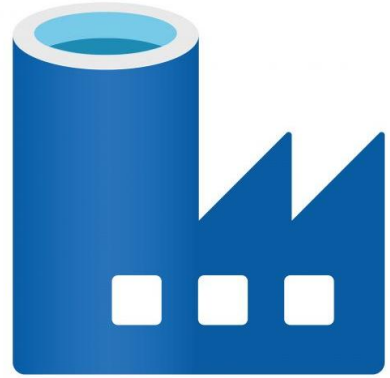
Facilitates configuration consistency across various pipelines.

Provides flexibility by allowing dynamic adjustments.

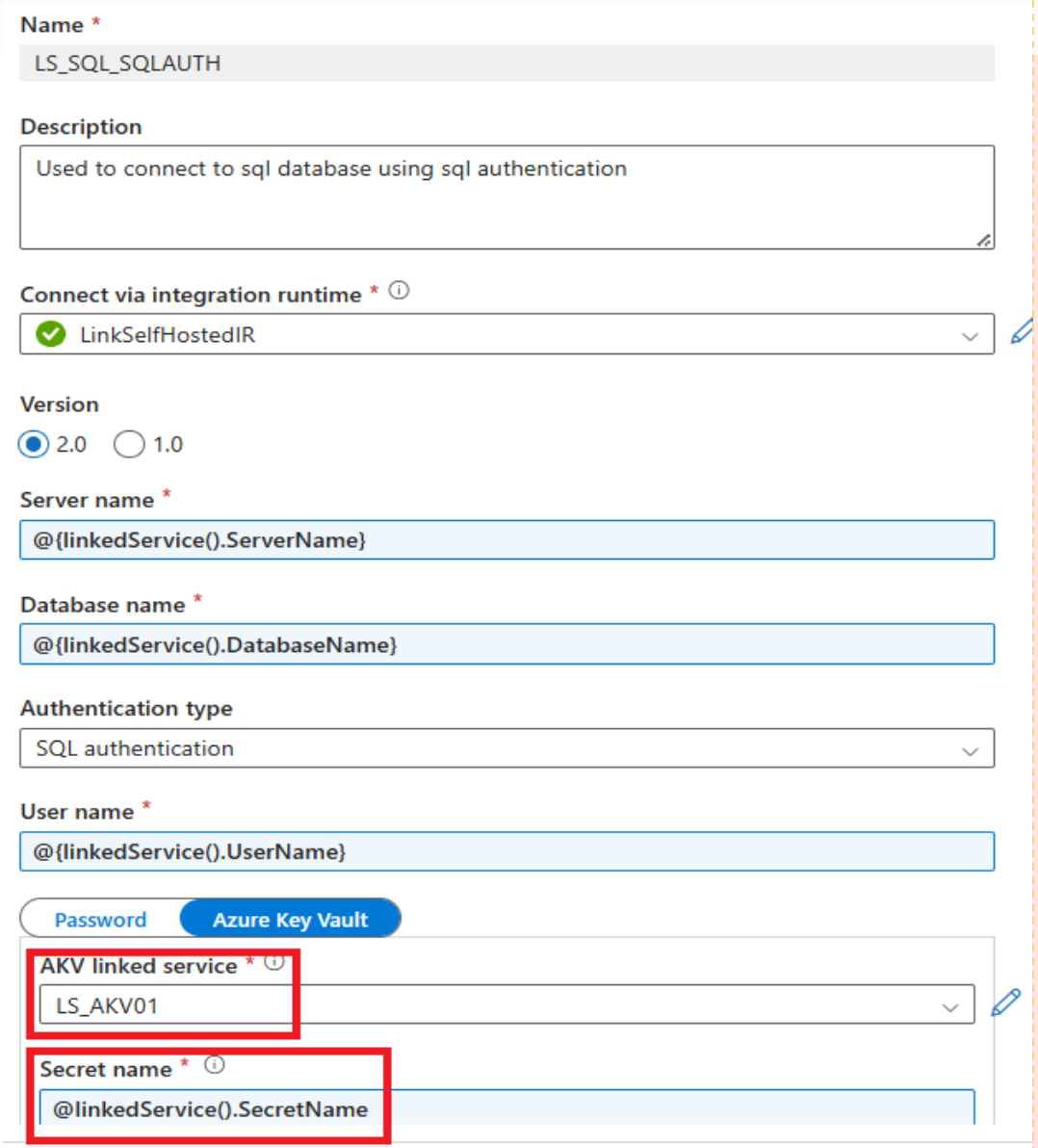
Promotes reusability of configurations for multiple tasks.

Leveraging AKV with ADF

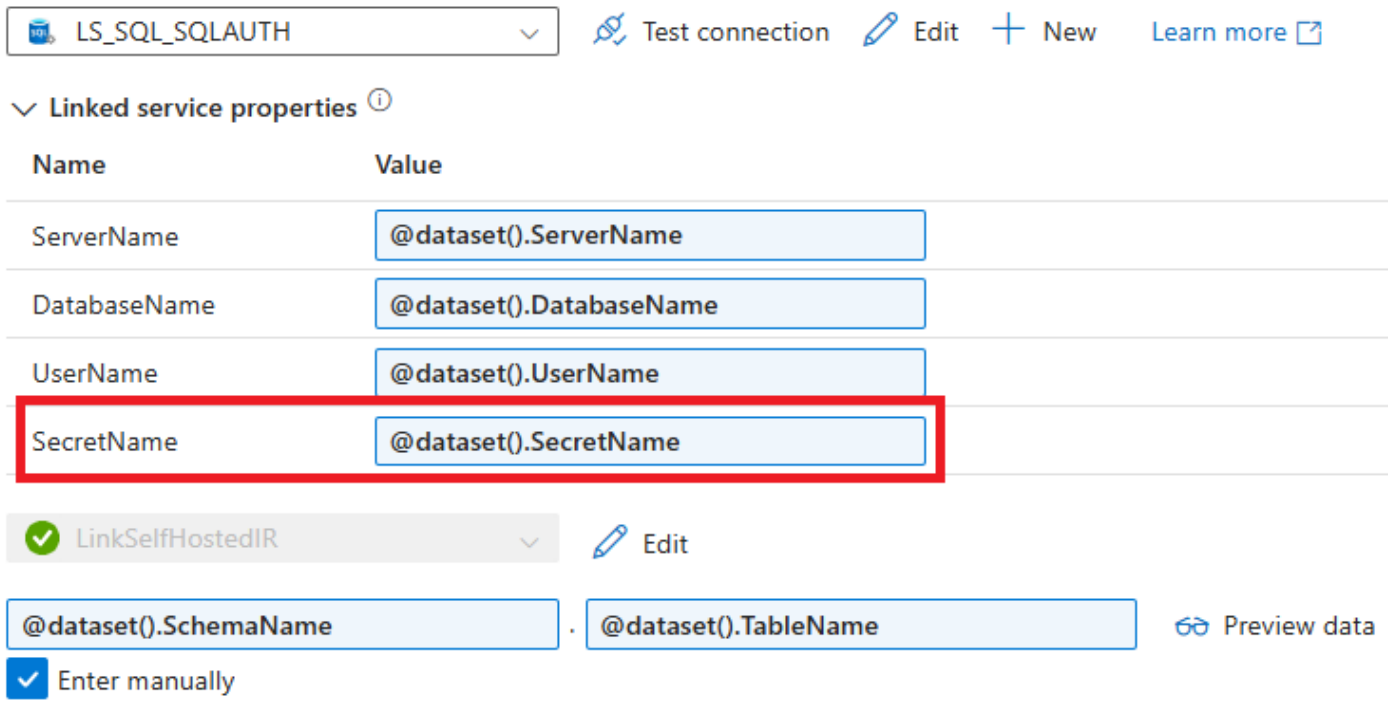
- **Store** credentials, connection strings, and secrets safely.
- **Simplify** access control through Azure Key Vault integration.
- **Automatically retrieve** secrets during pipeline execution.



Using AKV in Linked Services



Using AKV in Dataset



Benefits of CI/CD in ADF

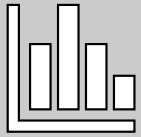
- Streamlines deployment processes for data pipelines.
- Enhances collaboration among development teams.
- Facilitates rapid changes.

Demos on Best Practices

Additional Recommendations

Bulk Insert Table Lock

Set to Yes for improved performance



User Properties for Monitoring

Enhanced tracking and management

Description in LS_ASQL_MI

More user-friendly interface

Conclusion

Adopt	Adopt Standard Naming Conventions
Reuse	Reuse Linked Services, Datasets and Pipelines
Parameterize	Parameterize As Much As Possible
Use and Standardize	Use and Standardize the Configuration Tables
Integrate	Integrate with Azure Key Vault
Integrate	Integrate with CI/CD

https://github.com/edjaja-olr/sqlsatdemo