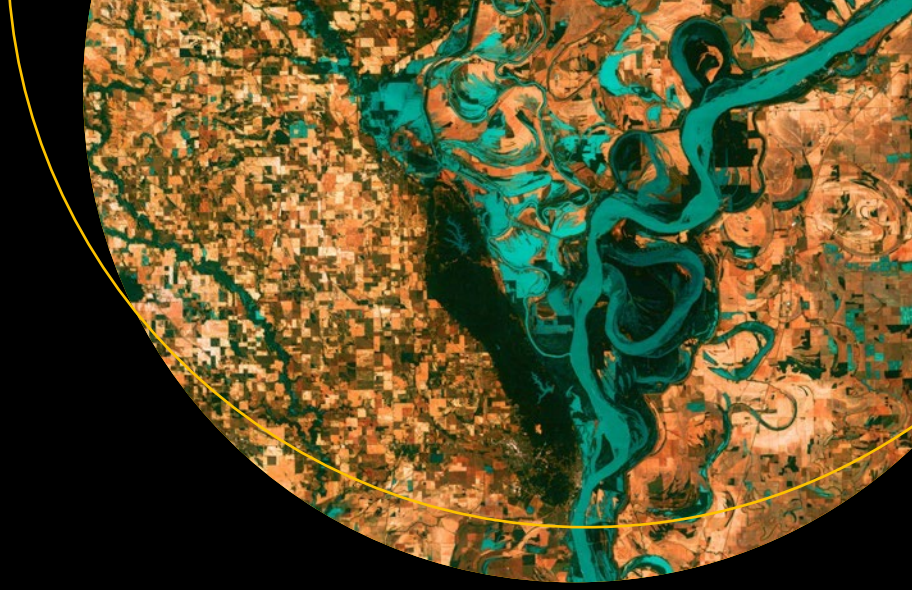

Mapping Data Flows in Azure Data Factory

Building Scalable ETL Projects in the Microsoft Cloud

Mark Kromer

Apress®

Mapping Data Flows in Azure Data Factory

Building Scalable ETL Projects in the Microsoft Cloud

Mark Kromer

Apress®

Mapping Data Flows in Azure Data Factory: Building Scalable ETL Projects in the Microsoft Cloud

Mark Kromer
SNOHOMISH, WA, USA

ISBN-13 (pbk): 978-1-4842-8611-1 ISBN-13 (electronic): 978-1-4842-8612-8
https://doi.org/10.1007/978-1-4842-8612-8

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Jonathan Gennick
Development Editor: Laura Berendson
Coordinating Editor: Jill Balzano

Cover designed by eStudioCalamar

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media New York, 1 New York Plaza, Suite 4600, New York, NY 10004-1562, USA. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub (https://github.com/Apress). For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

This book is dedicated to my loving wife Stacy and
our boys Ethan and Jude. Thank you for putting up with
my late hours working on data analytics and writing this book!

Table of Contents

About the Author

Mark Kromer has been in the data analytics product space for over 20 years and is currently a Principal Program Manager for Microsoft's Azure data integration products. Mark often writes and speaks on big data analytics and data analytics and was an engineering architect and product manager for Oracle, Pentaho, AT&T, and Databricks prior to Microsoft Azure.

About the Technical Reviewer

Andy Leonard is a husband, dad, and grandfather; creator of – and Data Philosopher at – DILM Suite for Data Integration Lifecycle Management (dilmsuite.com); a blogger (andyleonard.blog); founder and Chief Data Engineer at Enterprise Data & Analytics (entdna.com); an SSIS and Azure Data Factory trainer, consultant, and developer; a SQL Server database and data warehouse developer; and an author, mentor, engineer, and farmer.

Introduction

The ETL (extract, transform, load) process has been a cornerstone of data warehouses, data marts, and business intelligence for decades. ETL is how data engineers have traditionally refined raw data into business analytics that guide the business to make better decisions. These projects have allowed engineers to build up libraries of common ETL processes and practices from traditional on-premises data warehouses over the years, very commonly with data coming from Oracle, Microsoft, IBM, or Sybase databases or business ERP/CRM applications like Salesforce, SAP, Dynamics, etc. However, over the past decade, our industry has seen these analytical workloads migrate to the cloud at a very rapid pace.

To keep up with these changes, we've had to adjust ETL techniques to account for more varied and larger data. The big data revolution and cloud migrations have forced us to rethink many of our proven ETL patterns to meet modern data transformation challenges and demands. Today, the vast majority of data that we process exists primarily in the cloud. And that data may not always be governed and curated by rigid business processes in the way that our previous ETL processes could rely on.

The common scenarios of processing well-known hardened schemas from SAP and CSV exports will now have a new look and challenge. The data sources will likely vary in shape, size, and scope from day to day. We need to account for schema drift, data drift, and other possible obstructions to refining data in a way that turns the data into refined business analytics.

Cloud-First ETL with Mapping Data Flows

Welcome to *Mapping Data Flows in Azure Data Factory*! In this book, I'm going to introduce you to Microsoft Azure Data Factory and the Mapping Data Flows feature in ADF as the key ETL toolset to tackle these modern data analytics challenges. As you make your way through the book, you'll learn key concepts, and through the use of examples, you'll begin to build your first cloud-based ETL projects that can help you to

unlock the potential of scaled-out big data ETL processing in the cloud. I'll demonstrate how to tackle the particularly difficult and challenging aspects of big data analytics and how to prepare data for business decision makers in the cloud.

To get the most value from this book, you should have a firm understanding of building data warehouses and business intelligence projects. It is not necessary to have many hours of experience building cloud-first big data analytics projects already. However, having some experience in cloud computing will provide valuable context that will help you as you work through some of these new approaches.

The examples and scenarios used in this book will be patterns and practices that are based on ETL common scenarios, so having data engineering experience and background will also be very helpful. I'll help guide you along as you migrate from traditional on-premises data engineering to the world of Azure Data Factory.

Overview of Azure Data Factory

To become familiar with the data engineering process in Microsoft Azure, we'll need to begin with an overview of Azure Data Factory (ADF), which is the Azure service for building data pipelines. The first chapter will focus on conceptual discussions of how to build a process to transform massive of amounts of data with many quality issues in the cloud. Essentially, we need to redefine ETL for cloud-based big data, where data volumes and veracity can change daily, and we'll compare and contrast the Azure mechanism for the modern data engineer with traditional ETL. That's where we'll begin the process of building ETL pipelines that will serve as the basis for your big data analytics projects. I'm going to present a series of common use cases that will demonstrate how to apply the concepts discussed in the earlier chapters to practical ETL projects. From there, the focus will turn to a deep dive on Mapping Data Flows and how to build ETL frameworks in ADF by using the visual design-time interface to build code-free data flows. Mapping Data Flows is primarily a code-free visual design experience, so we'll walk through techniques and best practices for managing the software development life cycle of a data flow in ADF. Data Factory provides many different means to process and transform data that include coding and calling external compute processes. However, in this book, the focus will be on building ETL pipelines in a code-free style in Mapping Data Flows.

As you work your way through the early chapters in this book, you should begin to develop an understanding of how to apply data engineering principles in ADF and Mapping Data Flows. That's where we'll begin to implement mechanisms to

help organize your work and design-time environment, preparing for eventual operationalization at runtime. We'll set up a Git repo for our work, as you should in real-life scenarios. We'll design interactive data transformation graphs using serverless compute that can scale out as needed. You won't need to manage physical servers and clusters with ADF, but I will explain how things work behind the scenes to provide this serverless compute power for your pipelines. Behind the scenes, ADF will leverage the Azure platform-as-a-service workflow engine Logic Apps for pipeline execution and scheduling. The transformation engine for Mapping Data Flows is Apache Spark. But you won't have to learn anything about those underlying dependent services. The Azure Integration Runtimes will provide that compute for you dynamically in a serverless manner.

Operationalizing Data Pipelines

As you begin designing data flows for cloud-first big data workloads, we will test and debug in nonproduction environments and then promote that work to production environments. Execution of those jobs will be performed via ADF data pipelines based on schedules. These chapters will focus on operationalizing our work in a way that will become the eventual automated ETL framework for your business analytics. A complete end-to-end solution must also require monitoring and management of these processes on an ongoing basis. The final chapters will provide mechanisms in ADF that can be leveraged to monitor runs over time and to examine the performance of your pipelines. Because the nature of big data in the cloud is that the data will be messy and ever-changing, it is important to establish alerts and handling for schema and data drift. I'll explain how to add fail-safe mechanisms, monitoring, and traps for these common problems so that your data pipelines can execute continuously. The frameworks needed for design, debug, schedule, monitor, and manage are all contained inside of ADF, and we'll spend time digging into each one of those areas.

Goal for the Book

My goal is that by the end of this book, you'll be able to apply the concepts and the patterns presented here to build ETL pipelines for your next big data analytics project in the cloud. By mapping these new, updated approaches to processing data for analytics

(a.k.a. big data analytics) to the world of traditional ETL processing that you are already familiar with, you will be able to use Azure Data Factory and Mapping Data Flows to provide your business with analytics that will result in making better business decisions. Many of the patterns and practices in this book can be applied directly to your projects where you are beginning to build cloud-first data projects in Azure. You can use these techniques to begin building a new set of reusable common ETL patterns. As you work your way through the progression of this book's chapters, you'll build upon the lessons learned in each chapter with the goal of having all of the necessary lessons learned to begin building your own big data analytics ETL solution natively in the cloud using Azure Data Factory with Mapping Data Flows. So welcome, and I hope you find this book helpful as you begin building powerful ETL solutions in the cloud!

PART I

Getting Started with Azure Data Factory and Mapping Data Flows

CHAPTER 1

ETL for the Cloud Data Engineer

In the modern business data ecosystem, "digital transformation" is one of the most prominently used terms to describe the transformation of traditional technology practices to cloud and big data approaches. The term has become a ubiquitous term in IT and has come to represent the embrace of cloud and big data technologies in the data engineering world.

The data part of this digital business transformation puts data engineers at the center of the data processing value chain. What data engineers are challenged with is how to find a way to effectively extract, transform, and load massive amounts of new data points that are often unwieldy in nature. That means that we have to update our ETL processes to meet these new cloud-first big data approaches. Digital transformation is crucial for the success of businesses to compete and grow in today's cloud-first IT strategies, so let's dig into how to adjust and build comparable solutions in Azure using ADF and Mapping Data Flows.

General ETL Process

Figure 1-1 is an example of a general ETL process from traditional on-premises projects where your sources are highly governed source data like data that originates from SAP, database tables, and file extracts that abide by well-known contracts.

M. Kromer, *Mapping Data Flows in Azure Data Factory*, https://doi.org/10.1007/978-1-4842-8612-8_1

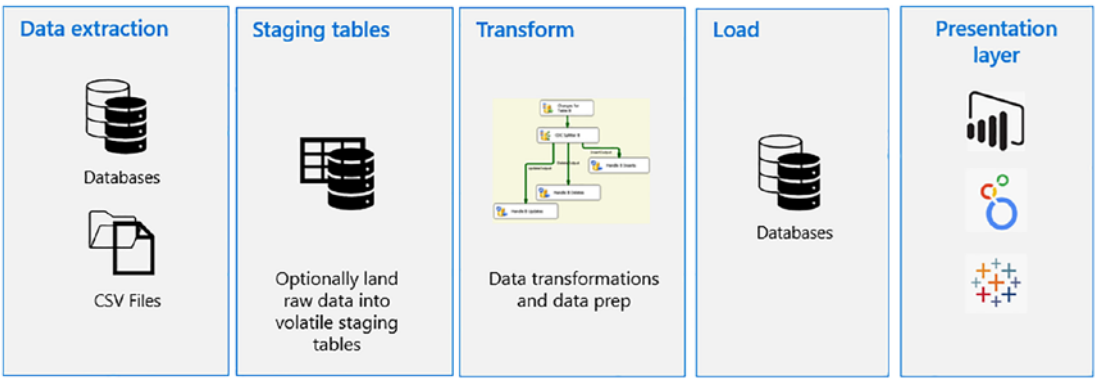

Figure 1-1. *Traditional ETL general process*

As a data engineer working on cloud-first projects in Microsoft Azure, you'll employ a process similar to the diagram in Figure 1-2, which only differs slightly from the concepts shown in Figure 1-1. But the details in each step bring about a significant amount of change that will be the topic of the ADF-specific chapters to come. At the end of the day, the objective of preparing data for business decision makers, who will use business intelligence tools, SQL queries, Excel, data science tools, and other decision-oriented tooling, is no different than you see in traditional on-premises scenarios with highly curated data sources and targets.

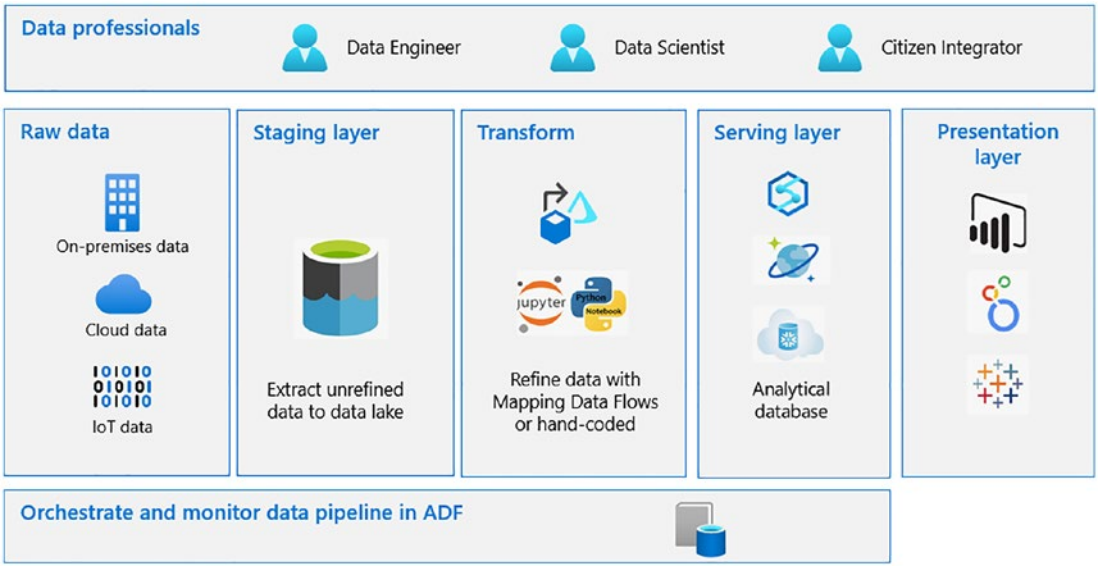

Figure 1-2. *A general example of the ETL process in Azure*

The consumers of the analytics in both of these instances are analysts who are building reports where actual business value is derived for business decision makers. For the data to be useful, the data engineers, data scientists, and citizen data integrators must contribute in a governed way to refining raw data into business-friendly models for exploration and reporting.

Differences in Cloud-Based ETL

We'll need to have a common understanding of what we are achieving in this book, so let's dive into this process in detail and identify some of the differences in cloud-based ETL in Azure from similar traditional on-premises ETL projects:

1. Raw data
    a. Much of the data extraction in big data cloud ETL will be of unknown quality and can change shape and size dramatically between job executions. In ADF, we'll make use of the Copy Activity and Data Flow Activity connectors, linked services, and datasets. In traditional data warehouse scenarios, you may have found that all of your business data resides on-premises and inside the network confines of your business. Additionally, often that data has been curated and already refined through a data quality process. Do not make such assumptions about data that you'll land in the data lake. The details of the different ADF components will come in the next set of chapters.
2. Staging layer
    a. This is where we will land an initial snapshot, lightly transformed, version of the source data in a landing zone in the data lake. For most of the demo scenarios in the book, we'll land the data in Azure Data Lake Store Gen2 (ADLS Gen2 or simply ADLS). If you've previously designed data warehouses with an ODS model or used database tables as staging tables, you can equate the staging layer in the data lake as an analogy. Because the data volumes are expected to be very large here, we will implement incremental data loading patterns in ADF rather than attempt to extract the entire set of data every time.