Proyecto ETL con Azure Data Factory

Valentina Giraldo Gaviria
Diana Marcela Silva
Eliana Marcela Fajardo Bermúdez
Héctor Hernán Betancourt Lopez

Docente:
Ángela Villota Gómez

Universidad Icesi
Facultad Barberi de Ingeniería, Diseño y Ciencias Aplicadas
Santiago de Cali
2025

Introducción

Cada día, las empresas generan y dependen de enormes cantidades de datos para tomar decisiones. Pero para que toda esa información tenga verdadero valor, es clave saberla gestionarla, es por ello que, existen herramientas ETL (Extract, Transform, Load) que se vuelven esenciales porque permiten reunir datos de diferentes fuentes y formatos, transformarlos y convertirlos en información útil y accesible para la organización.

En el presente proyecto se trabaja con el *dataset* Ames Housing, que contiene una amplia variedad de características 81 variables relacionadas con propiedades residenciales ubicadas en Estados Unidos. Estas variables incluyen información sobre tamaño, calidad de construcción, ubicación, atributos arquitectónicos y precios históricos de venta, para un total de 2.930 registros.

A continuación, se describen cada una de las actividades realizadas para implementar *pipelines* mediante Azure Data Factory (ADF), con el objetivo de aplicar procesos ETL y consolidar un archivo final en formato CSV que facilite el análisis del sector inmobiliario.

Actividades realizadas

- **Base de datos relacional:**

Se trabajó con un modelo relacional compuesto por seis tablas SQL: una tabla de hechos denominada amesdbtemp (35 columnas y 2.930 filas) y cinco tablas de dimensiones —MSSubClass (2 columnas, 16 filas), MSZoning (3 columnas, 8 filas), TypeQuality (3 columnas, 5 filas), SaleProperties (5 columnas, 2.930 filas) y FloorDetail (5 columnas, 4.182 filas).

Para conectar estas tablas, se estableció una conexión con **Azure Database for PostgreSQL** y a partir del modelo relacional proporcionado, se inició la creación del *pipeline*, comenzando con una actividad de tipo “Copy Data” que ejecuta una consulta SQL para unir las tablas de hechos y dimensiones, consolidando la información básica en una tabla temporal llamada *fullamesdbtemp*.

Posteriormente, se integró un *dataflow* de transformaciones que tiene como entrada la tabla temporal mencionada junto con la tabla de dimensiones *FloorDetail*, ya que esta última no fue incluida en el JOIN durante la consulta SQL inicial. Esta tabla temporal sirve como base para un Data Flow que realiza las siguientes transformaciones:

- ✓ Creación de nuevas columnas calculadas, como mes y año de la venta.
- ✓ Extracción del mes y año a partir de la fecha de venta.
- ✓ Agregación de datos relacionados con propiedades de múltiples pisos.
- ✓ Cálculo de superficie habitable total sumando las áreas de primer y segundo piso más terminación de baja calidad.
- ✓ Conversión de booleanos en registros legibles
- ✓ Joins y eliminación de columnas redundantes.

El resultado es un archivo de salida en formato CSV, denominado outputrelacional, que sintetiza la información de las seis tablas y las transformaciones indicadas.

- **Archivo CSV:**

El archivo AmesProperty.csv cuenta con 19 columnas que indican las características de la ubicación del predio, la fecha de construcción y la fecha de remodelación en caso de que se haya realizado; tiene un total de 2.930 filas cada fila es un dato independiente, y cuenta con un código asociado único (PID).

En el archivo csv se requería modificar el contenido de cuatro columnas: Lot Shape, Condition 1, Year Remod/Add y Neighborhood acorde con un código específico para las columnas (Lot Shape, Condition 1 y Neighborhood) y para la cuarta columna diligenciar los datos nulos. Para lograrlo se realizó una conexión con Blob Storage en Azure, y se generó por medio de un Data Flow con las siguientes actividades:

- ✓ Modificación del nombre de columnas para retirar espacios.
- ✓ Generar un condicional, que, al identificar un valor vacío o nulo, replicara la información de la columna Year Built, de lo contrario dejar el dato existente en la columna Yaer Remod/Add.
- ✓ Se creo un archivo con dos columnas una para código del barrio, y otra con la descripción del nombre del barrio. Lo anterior para realizar un left outer join, para cambiar el contenido de la columna Neighborhood.
- ✓ Se utiliza la función case para reemplazar la información de las columnas Lot Shape y Condition 1 para que queden los datos con el código que corresponde a cada característica.

Al final se obtuvo un archivo csv, con 19 columnas, sin embargo, las columnas indicadas anteriormente quedaron codificadas para mejorar el análisis de la información.

- **MongoDB Atlas:**

En MongoDB se encuentran cuatro colecciones con información complementaria sobre características de los inmuebles:

- Garage: 8 campos y 2.773 documentos
- Pool: 3 campos y 13 documentos
- Bsmt (sótano): 3 campos y 106 documentos
- Misc (otras características): 3 campos y 106 documentos

Para integrarlas en el entorno, se creó una conexión a MongoDB por medio de una actividad tipo Copy Data dentro de un pipeline. Este proceso, se realiza para cada colección, conformando al final un flujo conectado. Esta actividad extrae cada colección por separado y la exporta en un dataset en formato CSV.

A continuación, se configuró un Dataflow encargado de consolidar la información. El primer paso consistió en combinar los cuatro archivos CSV mediante operaciones de Join, cuidando especialmente la unificación del campo PID en cada combinación. Lo anterior, para asegurar una integración precisa y consistente de los registros.

Una vez generado un único archivo consolidado, se aplicaron las siguientes transformaciones:

- ✓ Unificación de las columnas PID provenientes de los cuatro archivos para conservar solo una columna final.
- ✓ Conversión de valores booleanos a representaciones legibles.
- ✓ Eliminación de columnas redundantes.
- ✓ Eliminación de registros duplicados según el PID.
- ✓ Imputación de valores nulos:
    - En variables cualitativas se reemplazaron por **"NA"**.
    - En variables cuantitativas se imputaron con **0**.

El resultado final es un archivo CSV denominado output_mongo, que consolida y estandariza la información de las cuatro colecciones originales junto con todas las transformaciones descritas.

- **Dataflow final:**

Finalmente, se construyó un dataflow consolidado que integra las salidas en formato CSV provenientes de las tres fuentes de datos descritas anteriormente. Sobre este flujo unificado se aplicaron las siguientes transformaciones:

- ✓ Unificación del campo PID de los tres archivos, con el fin de conservar una única columna identificadora final.
- ✓ Eliminación de registros duplicados según el PID.
- ✓ Imputación de valores nulos:
    - En variables cualitativas se reemplazaron por **"NA"**.
    - En variables cuantitativas se imputaron con **0**.

Como resultado, se obtiene un archivo final en formato CSV, compuesto por 81 columnas y 2.930 registros, que reúne de manera integrada la información proveniente de la base relacional, las colecciones de MongoDB y el archivo CSV inicial.

Lecciones aprendidas

- La importancia en comprender el modelo relacional antes de construir el pipeline fue una de las actividades claves, puesto que, de esto dependía directamente las relaciones y la correcta selección de joins y llaves; adicional, combinar la consulta SQL con transformaciones en el dataflow permitió mantener el pipeline más claro y modular.

- Se requiere el diseño de procesos ETL flexibles que permitan consolidar datos provenientes de diversas estructuras y formatos, tales como bases de datos relacionales, archivos CSV y colecciones en MongoDB.

- Fue necesario unificar correctamente el campo PID, ya que se evidenció que una configuración inadecuada en los joins puede afectar la integridad de los datos. Por eso, es fundamental revisar con cuidado cómo se combina la columna PID cuando se usa como identificador.

- El uso de *Copy Data* simplificó la extracción desde diversas fuentes, mientras que el Dataflow facilitó la transformación de los datos. Ahora, el uso simultáneo permitió comprender y obtener un pipeline robusto.

- La eliminación de atributos redundantes, la corrección de nombres y tipos de datos son actividades primordiales dentro de un proceso ETL, estas hacen referencia a efectuar una adecuada estandarización y limpieza de los datos, para generar un conjunto datos coherente y enriquecido garantizando calidad en el resultado final.

Referencias bibliográficas

Microsoft Azure, “Azure Data Factory Tutorial,” YouTube, 10-may-2021. [En línea]. Disponible en: https://www.youtube.com/watch?v=EpDkxTHAhOs

Microsoft, “Azure Data Factory — Documentación,” Microsoft Learn. [En línea]. Disponible en: https://learn.microsoft.com/en-us/azure/data-factory/

Microsoft, “Linked services in Azure Data Factory and Azure Synapse Analytics,” Microsoft Learn. [En línea]. Disponible en: https://learn.microsoft.com/ru-ru/azure/data-factory/concepts-linked-services?tabs=data-factory

MongoDB Inc., “Replication — MongoDB Manual.” [En línea]. Disponible en: https://docs.mongodb.com/manual/replication/

Universidad Politécnica de Madrid, “MOOC BigData: Sistemas gestores de bases de datos orientados a documentos III,” YouTube, Universidad Politécnica de Madrid, 17-may-2017. [En línea]. Disponible en: https://www.youtube.com/watch?v=eYiebokW2hg