**REPORTE DE PROYECTO FINAL INFRATI**

El proyecto tuvo como objetivo diseñar e implementar un flujo completo de integración y transformación de datos utilizando Azure Data Factory (ADF). Se trabajó con tres tipos de fuentes heterogéneas: una base relacional en PostgreSQL (Neon), una base documental en MongoDB Atlas y un conjunto de archivos CSV alojados en Azure Blob Storage. A partir de estas fuentes se buscó construir un dataset final limpio y unificado, para un posible análisis posterior del Ames Housing Dataset.

El proceso comenzó con la etapa de extracción. En el caso de Azure Blob Storage, se creó un linked service que permitió almacenar tanto el archivo principal (AmesProperty.csv) como el archivo de salida final output.csv. Para la información proveniente de Neon PostgreSQL, se configuró un linked service específico para este motor, lo que facilitó la creación de datasets asociados a las tablas relevantes: amesdbtemp, SalesProperty, FloorDetail, MsZoning, MsSubClass, etc. Por su parte, la extracción desde MongoDB Atlas requirió un manejo distinto: se configuró un linked service tipo MongoDB Atlas para acceder a las colecciones bsmt, garage, misc y pool. Debido a que estas colecciones estaban en formato JSON, fue necesario pasarlas por un pipeline adicional que permitió convertirlas a archivos tabulares (CSV). Solo después de este proceso pudieron integrarse al Data Flow principal, por lo que se generaron también archivos de salida específicos en Blob Storage para cada colección.

Una se cargaron las fuentes de datos, se procedió con el diseño del Data Flow. La primera etapa consistió en cargar todos los orígenes necesarios: la tabla principal de propiedades, las tablas dimensionales provenientes de SQL y los CSV resultantes de MongoDB. A cada uno de estos orígenes se le aplicó una transformación Select para estandarizar los nombres de columnas y asegurar la compatibilidad de tipos antes de realizar cualquier unión. La integración de datos continuó con una cadena de joins, el primer paso fue realizar un inner join entre el CSV principal y la tabla amesdbtemp utilizando la columna PID como llave primaria, después sse aplicaron distintos left joins para incorporar las demás tablas, también usando PID, para mantener la estructura de la base principal que es la obtenida en el pipeline de Postgres. Las únicas uniones que usaron claves distintas fueron las correspondientes a MsZoning y MsSubClass, que se enlazaron mediante sus columnas code, posteriormente relacionadas con amesdbtemp dentro de la estructura dimensional.

Para garantizar la calidad del dataset, se implementó una limpieza de valores nulos mediante columnas derivadas. Las columnas de tipo string reemplazaron los valores nulos por “NA”, mientras que las columnas numéricas (integer, long o short) los reemplazaron por 0.Una vez realizada la limpieza, se procedió a crear unas transformaciones dentro de nuestro dataset GrLivArea, calculada a partir de superficies habitables; MoSold y YrSold, obtenidas de la fecha de venta; y YearRemod/Add, ajustada considerando el año original de construcción cuando el dato venía nulo. Además, se realizaron agregaciones sobre la tabla FloorDetail para calcular métricas como TotalFullBath, TotalHalfBath y Bedroom.

Una vez integrados, limpiados y transformados los datos, el dataset final se exportó mediante un sink hacia un archivo CSV destino previamente configurado en Blob Storage. Este archivo representa la versión consolidada y lista para análisis del conjunto Ames Housing.Para ejecutar todo el procedimiento, el pipeline del proyecto se estructuró con una única actividad principal de tipo Data Flow. Su función fue orquestar y disparar la secuencia completa del proceso ETL, garantizando que cada transformación se aplicara en el orden adecuado hasta generar el archivo final.

El proyecto dejó varias lecciones valiosas. La más evidente fue la importancia de estructurar correctamente un flujo de datos en ADF: pequeños detalles como nombres inconsistentes, tipos de datos incorrectos o columnas innecesarias generaron errores en los joins que afectaron la calidad del resultado. Igualmente, trabajar la limpieza de valores nulos y las columnas derivadas demostró no ser una tarea trivial, pues exige entender a profundidad el contexto del dato para tomar decisiones adecuadas. Aunque varias de estas transformaciones presentaron dificultades iniciales por desconocimiento de la herramienta, se pudo conocer esta y valorar su utilidad para trabajos posteriores.