EA4. Documentación de la Arquitectura y Modelo de Datos

Infraestructura y Arquitectura para Big Data

Docente: Andrés Felipe Callejas

Estudiantes:

Mateo Valencia Minota

Carlos Andrés Cardona Quintero

Institución Universitaria Digital de Antioquia

Medellín

2025

1. Introducción

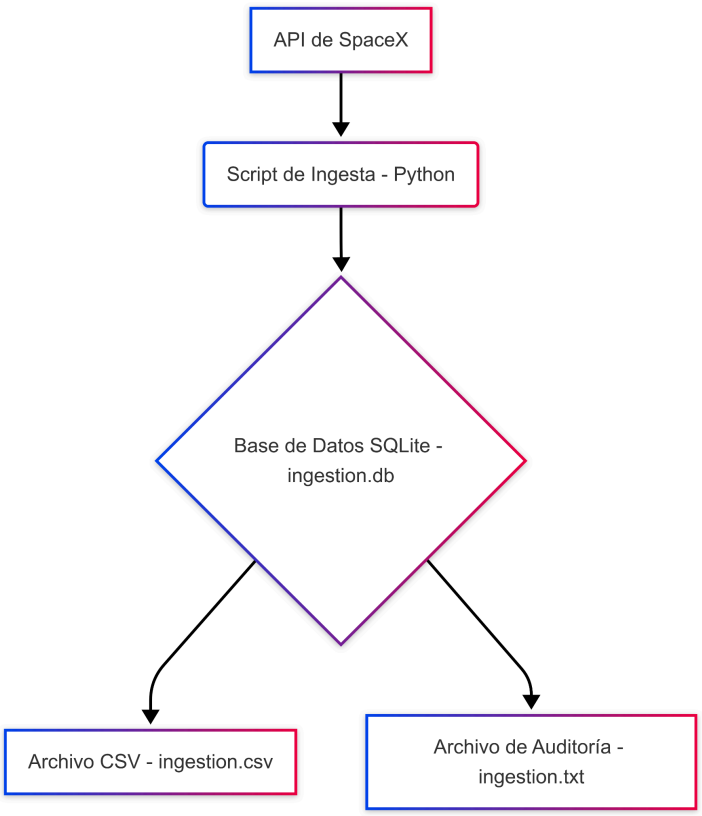
Este documento describe la arquitectura y el modelo de datos del proyecto integrador Big Data, que simula un entorno de nube para el procesamiento y análisis de datos de lanzamientos de cohetes de la API de SpaceX. El proyecto se divide en tres fases (entregas) principales: ingesta de datos, preprocesamiento y limpieza, y enriquecimiento de datos. El objetivo es construir un modelo de datos robusto y una arquitectura escalable que permita realizar análisis significativos sobre los datos de lanzamientos, así como de servir de herramienta de aprendizaje para desarrollar, con buenas prácticas y conocimientos de arquitectura, proyectos de Big Data en la industria.

2. Descripción General de la Arquitectura

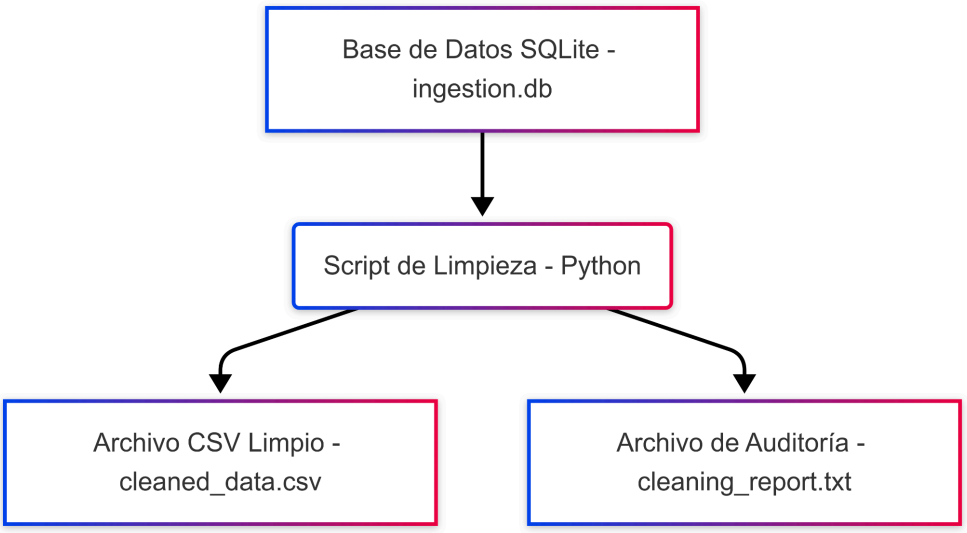
La arquitectura del proyecto se basa en un enfoque modular y automatizado. Cada fase del proyecto (ingesta, preprocesamiento y enriquecimiento) se implementa como un script de Python independiente. Los datos se almacenan en una base de datos SQLite para simular un entorno de almacenamiento en la nube. GitHub Actions se utiliza para automatizar la ejecución de los scripts y la generación de informes, asegurando la reproducibilidad y la eficiencia de nuestro proyecto.

3. Diagramas de Arquitectura

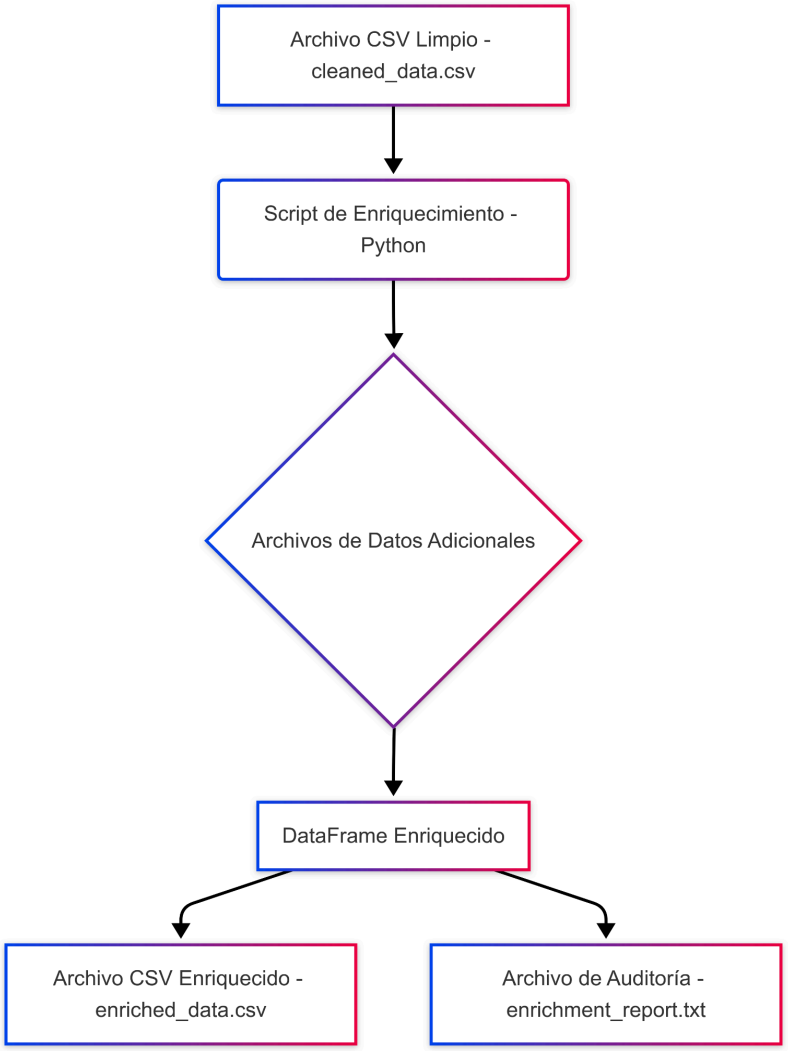
- **Diagrama de Flujo de Datos (Ingesta)**:



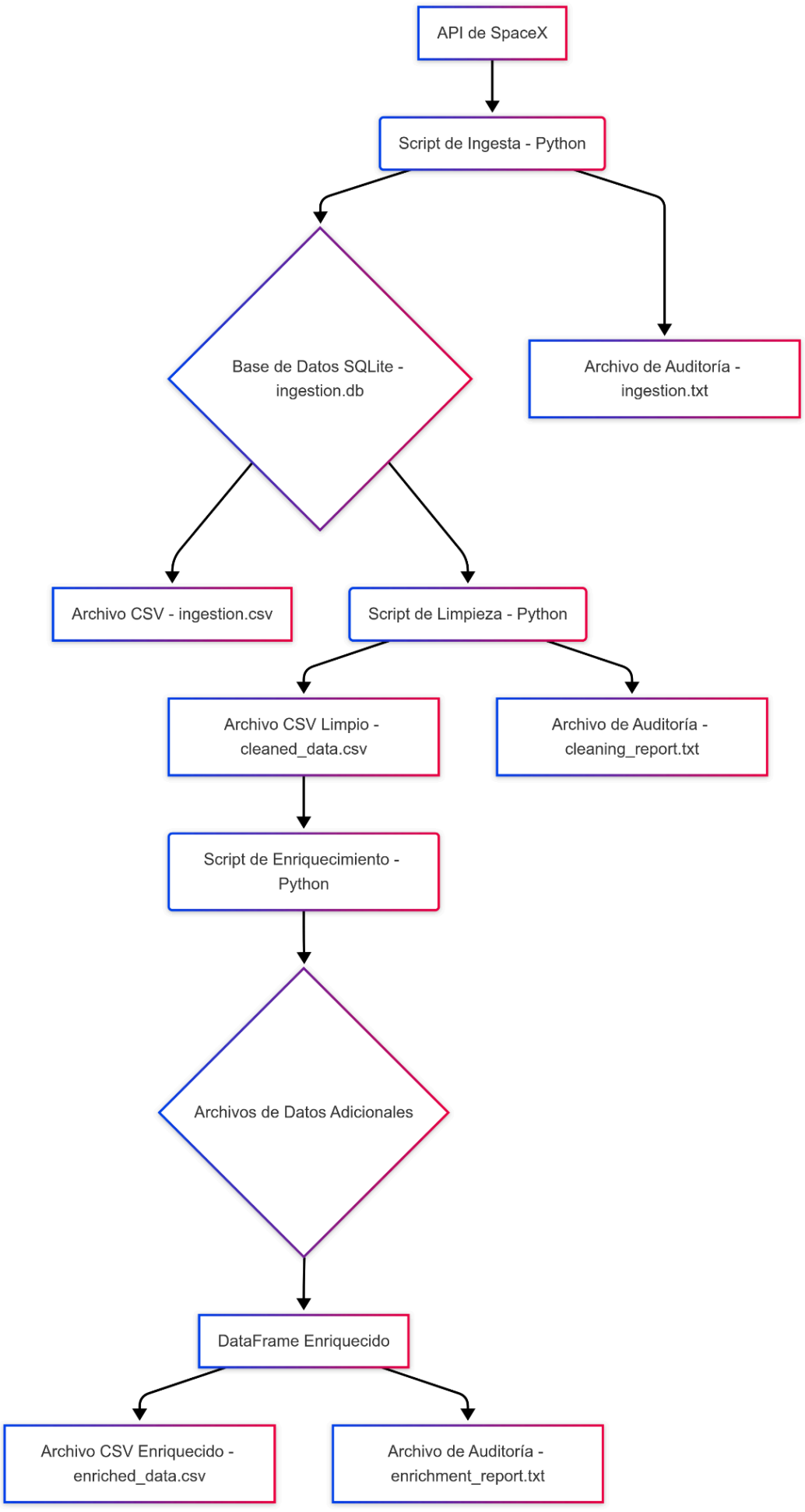
- **Diagrama de Flujo de Datos (Preprocesamiento y Limpieza)**:



- **Diagrama de Flujo de Datos (enriquecido)**:



- **Diagrama de arquitectura (unificado):**


API de SpaceX
Script de Ingesta - Python
Base de Datos SQLite - ingestion.db
Archivo de Auditoría - ingestion.txt
Archivo CSV - ingestion.csv
Script de Limpieza - Python
Archivo CSV Limpio - cleaned_data.csv
Archivo de Auditoría - cleaning_report.txt
Script de Enriquecimiento - Python
Archivos de Datos Adicionales
DataFrame Enriquecido
Archivo CSV Enriquecido - enriched_data.csv
Archivo de Auditoría - enrichment_report.txt

4. Modelo de Datos

LAUNCHES			
TEXT	id	PK	Identificador único del lanzamiento
INTEGER	flight_number		Número de vuelo
TEXT	name		Nombre del lanzamiento
TEXT	date_utc		Fecha y hora UTC
TEXT	date_local		Fecha y hora local
BOOLEAN	success		Éxito del lanzamiento
TEXT	details		Detalles adicionales
TEXT	rocket_id		ID del cohete utilizado
TEXT	launchpad_id		ID de la plataforma
TEXT	payloads		Cargas útiles (JSON)
TEXT	ships		Barcos involucrados (JSON)
TEXT	capsules		Cápsulas involucradas (JSON)
TEXT	failures		Fallos ocurridos (JSON)
TEXT	timestamp		Fecha y hora de inserción

- **Esquema de la Base de Datos**:

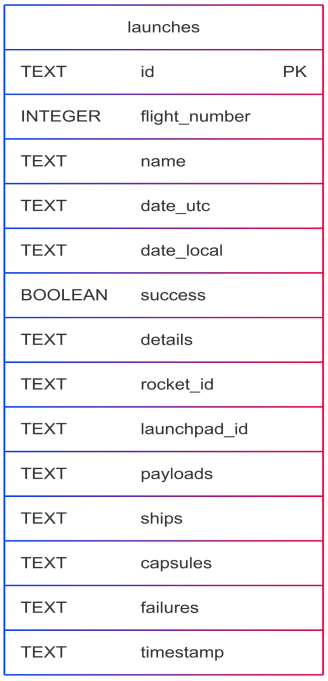
    - `id` (TEXT, PRIMARY KEY): Identificador único del lanzamiento.
    - `flight_number` (INTEGER): Número de vuelo del lanzamiento.
    - `name` (TEXT): Nombre del lanzamiento.
    - `date_utc` (TEXT): Fecha y hora UTC del lanzamiento.
    - `date_local` (TEXT): Fecha y hora local del lanzamiento.
    - `success` (BOOLEAN): Indica si el lanzamiento fue exitoso.
    - `details` (TEXT): Detalles adicionales sobre el lanzamiento.
    - `rocket_id` (TEXT): Identificador del cohete utilizado.
    - `launchpad_id` (TEXT): Identificador de la plataforma de lanzamiento.
    - `payloads` (TEXT): Datos de las cargas útiles (JSON).
    - `ships` (TEXT): Datos de los barcos involucrados (JSON).
    - `capsules` (TEXT): Datos de las cápsulas involucradas (JSON).
    - `failures` (TEXT): Datos de los fallos ocurridos (JSON).
    - `timestamp` (TEXT): Fecha y hora de inserción del registro.

- **Diagrama ER**:

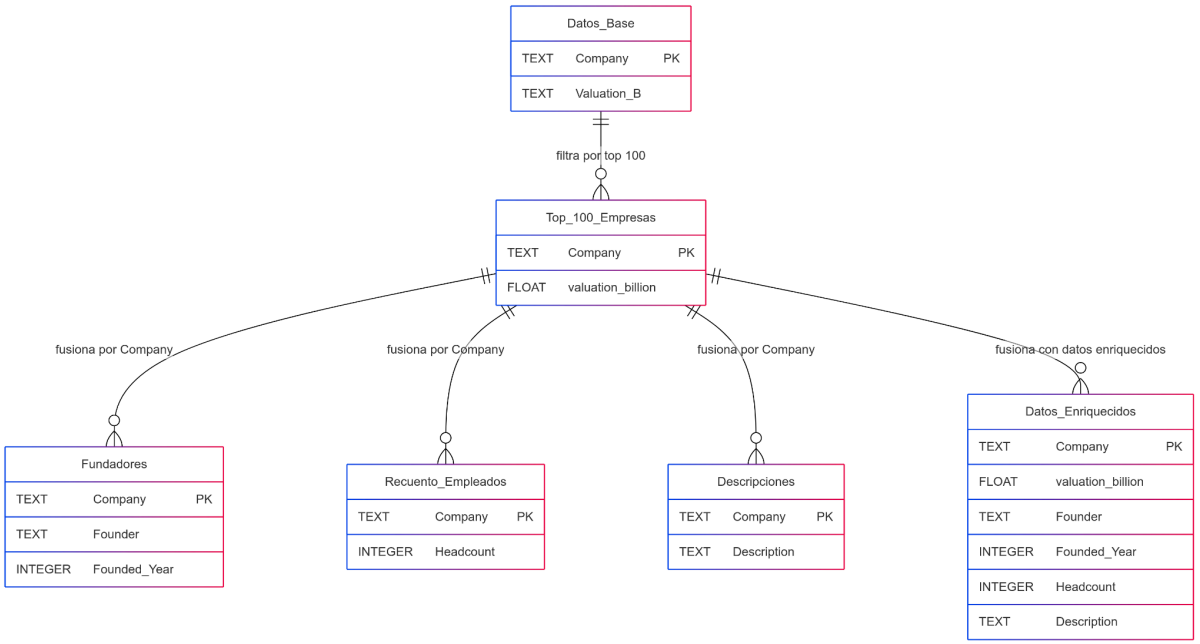
    En nuestro caso, como solo tenemos una tabla, el diagrama ER muestra la estructura interna de esa tabla. Si tuviéramos múltiples

tablas con relaciones entre ellas , el diagrama ER mostraría esas relaciones con líneas que conectan las tablas y anotaciones que indican la cardinalidad.

En el siguiente diagrama ER se visualiza la tabla *launches* y sus columnas, mostrando la clave primaria (PK) ID.



Sin embargo, como en el proceso de enriquecimiento decidimos usar una data diferente para simular el proceso, a continuación se muestra el diagrama ER de este proceso simulado:



- **Justificación**:

    - El modelo de datos se diseñó para almacenar la información relevante de los lanzamientos de SpaceX de manera estructurada y eficiente. Las columnas permiten realizar análisis detallados sobre los lanzamientos, como la tasa de éxito, los cohetes utilizados y las cargas útiles.

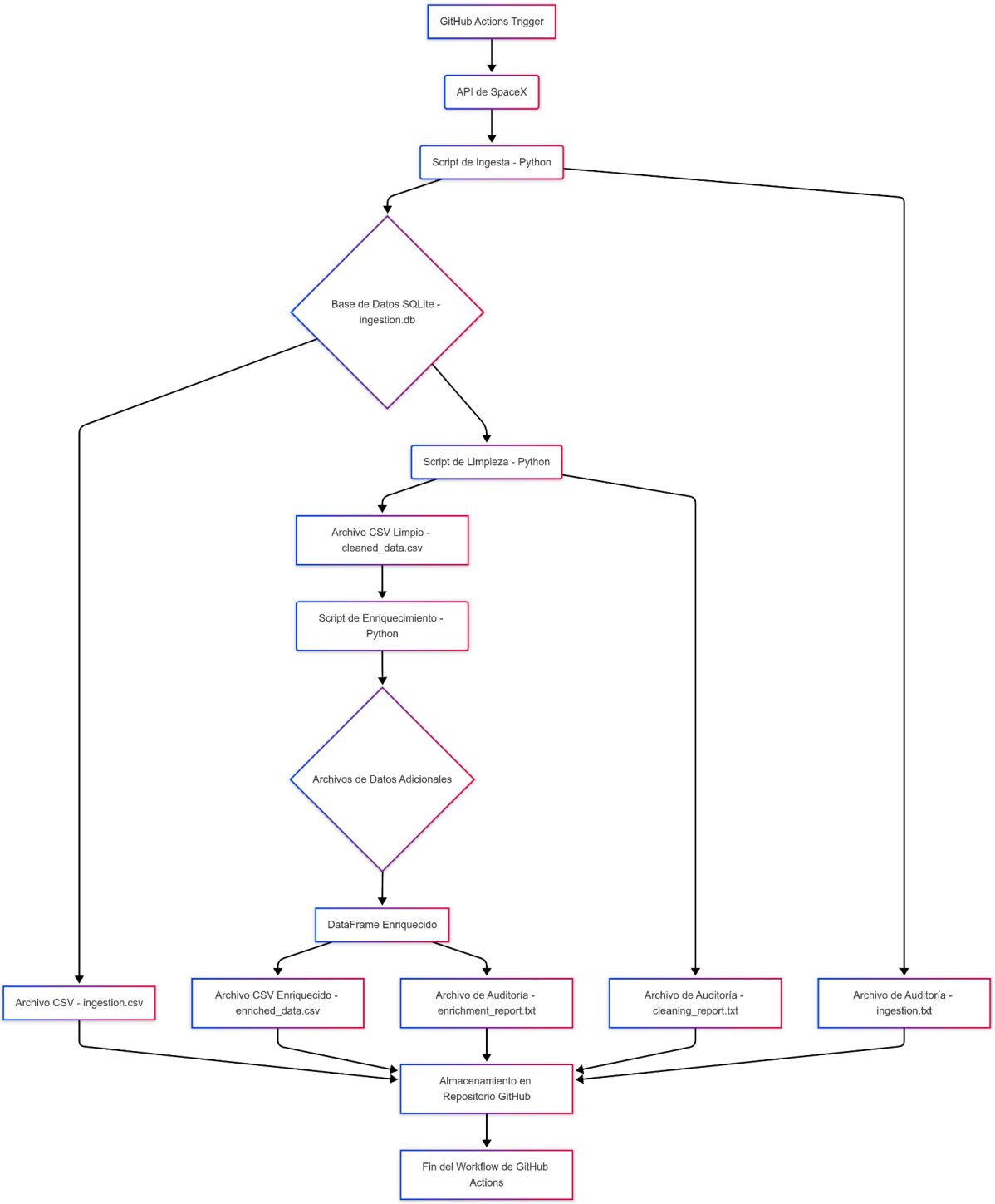
5. Justificación de Herramientas y Tecnologías

- **SQLite**: Se eligió por su simplicidad y facilidad de uso, simulando un entorno de almacenamiento en la nube sin la complejidad de configurar un sistema de base de datos distribuido.
- **Pandas**: Se utilizó para la manipulación y análisis de datos en Python, facilitando la limpieza, transformación y enriquecimiento de los datos.
- **GitHub Actions**: Se implementó para automatizar el flujo de trabajo del proyecto, asegurando la reproducibilidad y facilitando la integración continua.

- **Simulación del Entorno Cloud**: La combinación de SQLite y GitHub Actions simula un entorno de nube al proporcionar almacenamiento de datos y automatización de procesos de manera similar a los servicios en la nube.

6. Flujo de Datos y Automatización

- **Flujo de Datos**: Los datos se extraen de la API de SpaceX, se almacenan en la base de datos SQLite, se limpian y transforman con Pandas, se enriquecen con datos adicionales y se exportan a archivos CSV para su análisis.
- **Automatización**: GitHub Actions ejecuta automáticamente los scripts de Python para cada fase del proyecto, genera archivos de auditoría y almacena los resultados en el repositorio de GitHub.



7. Conclusiones y Recomendaciones

Beneficios: El desarrollo de este proyecto nos ayudó a desarrollar la capacidad de construir un flujo de trabajo de Big Data completo utilizando herramientas y tecnologías accesibles. La automatización con GitHub Actions mejora la eficiencia y la reproducibilidad del proyecto,

permitiendo un entorno de integración y despliegue continuo, simulando una de las maneras estándar de trabajar en la industria.

Limitaciones: La simulación del entorno de nube con SQLite tiene limitaciones en cuanto a escalabilidad y rendimiento en comparación con los servicios de nube reales.