

Unidad 4

Actividad:

Documentación del proceso de extracción, carga y enriquecimiento de datos en una infraestructura de Big Data simulada.

Julio César Cárdenas Veloth

Módulo – Infraestructura y arquitectura para Big Data

Grupo PREICA2501B010112

Profesor

Andrés Felipe Callejas

Institución Universitaria Digital de Antioquia

Ingeniería de Software y Datos

Medellín

2025

1 Introducción y descripción global de la arquitectura

Este proyecto tiene como objetivo simular una arquitectura de procesamiento de datos en un entorno de Big Data en la nube, utilizando tecnologías accesibles como Python, SQLite y GitHub. La arquitectura está diseñada para ejecutar de forma automática y controlada un flujo de trabajo completo de datos, desde su adquisición hasta su transformación y enriquecimiento, todo integrado dentro de un entorno local sincronizado con un repositorio remoto.

1.1 Componentes de la Arquitectura

1.1.1 Origen de Datos - API Pública COVID-19

Se consumen datos desde la API <https://coronavirus.m.pipedream.net/>, que ofrece información actualizada sobre contagios, muertes, recuperaciones y tasas de incidencia del COVID-19 a nivel global.

1.1.2 Almacenamiento Local - SQLite

Los datos son almacenados localmente en una base de datos relacional SQLite, simulando el uso de una base de datos transaccional en un entorno de nube.

Se crean distintas tablas para almacenar datos crudos (covid_data) y enriquecidos (dat_trans).

1.1.3 Procesamiento - Scripts en Python

Se desarrollaron varios scripts en Python para realizar:

- Ingesta de datos desde la API.
- Carga en SQLite para persistencia local.
- Limpieza de datos, manejo de nulos, estandarización de formatos y tipos.

- Enriquecimiento de información, añadiendo:
 - Códigos ISO por país.
 - Clave compuesta code_key.
 - Información socioeconómica poblacional y de ingreso per cápita mediante merge con archivos CSV externos.

1.1.4 Automatización - GitHub Actions

Todo el proceso fue automatizado mediante GitHub Actions, que actúa como orquestador simulado en la nube:

Cada vez que se realiza un push al repositorio, los scripts se ejecutan automáticamente.

Se generan nuevos archivos, bases de datos actualizadas y reportes de auditoría que son sincronizados automáticamente con el repositorio.

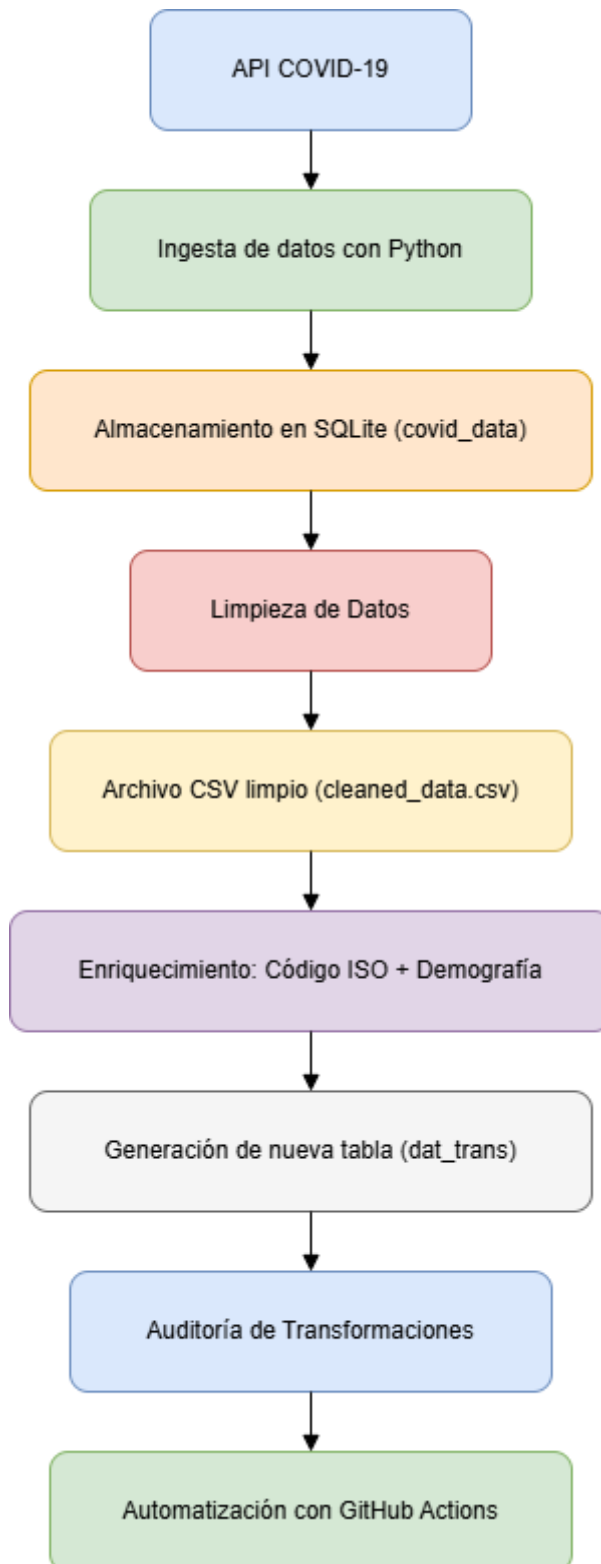
Esta funcionalidad simula el comportamiento de pipelines automatizados como los que se encontrarían en plataformas de nube (por ejemplo, AWS Lambda + S3 + Glue, o GCP Cloud Functions + BigQuery).

1.1.5 Trazabilidad y Auditoría

- Se generan archivos .txt con reportes de auditoría:
- Comparación entre datos obtenidos y almacenados.
- Registro de columnas agregadas y merge realizados.
- Resumen de registros afectados.

1.2 Diagrama de flujo del proceso

En el siguiente diagrama se muestra de manera gráfica, cada uno de los pasos realizado para simular una infraestructura para Big Data.



1.3 Modelo de datos

1.3.1 Datos de entrada

Para realizar la simulación de una infraestructura de Big Data en la nube, se utilizaron como insumo los siguientes datos.

- Dataset de Entrada (API COVID-19)

Estos datos provienen de la API <https://coronavirus.m.pipedream.net/>, que posee datos de contagio por COVID para todo el mundo.

A continuación se describen cada uno de los campo.

Campo	Descripción
FIPS	Código de identificación geográfica de EE. UU.
Admin2	Subdivisión administrativa secundaria (condado o municipio).
Province_State	Provincia o estado donde se reporta el caso.
Country_Region	País de origen del registro.
Last_Update	Fecha y hora de la última actualización.
Lat	Latitud del lugar reportado.
Long_	Longitud del lugar reportado.
Confirmed	Total de casos confirmados.
Deaths	Total de muertes reportadas.
Recovered	Total de personas recuperadas.
Active	Casos activos en el momento de la medición.
Combined_Key	Llave combinada que une varias ubicaciones (país, estado, municipio).
Incident_Rate	Tasa de incidencia por cada 100,000 personas.
Case_Fatality_Ratio	Porcentaje de letalidad (muertes confirmadas sobre total de casos).

- Tabla de Enriquecimiento (pob_ipc.csv)

Esta tabla se creo utilizando IA a partir de la tabla de código ISO-3166, que contiene un código único de 3 letras para cada país del mundo y se completó con datos poblacionales y PIB.

En la siguiente tabla se describen los campos

Campo	Descripción
Population	Población estimada del país o subdivisión.
Income_pc	Ingreso per cápita estimado (USD).

1.3.2 Datos de salida

Como tablas de salida y almacenadas en sqlite se tienen las siguientes con su respectiva descripción de campos.

- Tabla covid_data

Contiene los mismos campos que el dataset de entrada más las siguientes columnas agregadas code_iso y code_key.

Descripción de los campos.

Campo	Descripción
FIPS	Código Federal de Procesamiento de Información Estándar. Identifica condados en EE. UU.
Admin2	Subdivisión administrativa de segundo nivel, como condados o municipios.
Province_State	Estado, provincia o territorio donde se registra el dato.
Country_Region	País o región donde ocurre el evento de salud.
Last_Update	Fecha y hora de la última actualización del registro.
Lat	Latitud geográfica del lugar donde se registran los casos.
Long_	Longitud geográfica del lugar donde se registran los casos.
Confirmed	Número total acumulado de casos confirmados por COVID-19.
Deaths	Número total acumulado de muertes por COVID-19.

Recovered	Número total de personas que se han recuperado del COVID-19.
Active	Casos activos al momento del reporte (generalmente: Confirmed - Deaths - Recovered).
Combined_Key	Llave compuesta que concatena valores de ubicación (e.g., Admin2, Province_State, Country_Region).
Incident_Rate	Tasa de incidencia por cada 100,000 habitantes.
Case_Fatality_Ratio	Proporción de muertes entre los casos confirmados (%).
code_iso	Código ISO-3166-1 alfa-3 del país, utilizado para estandarización internacional (e.g., "COL" para Colombia).
code_key	Llave única para cada registro, construida concatenando code_iso, Province_State y Admin2.

- Tabla dat_trans

Es la tabla enriquecida, producto del merge entre covid_data y datos poblacionales/IPC.


Descripción de los campos.

Campo	Descripción
FIPS	Código Federal de Procesamiento de Información Estándar. Identifica condados en EE. UU.
Admin2	Subdivisión administrativa de segundo nivel, como condados o municipios.
Province_State	Estado, provincia o territorio donde se registra el dato.
Country_Region	País o región donde ocurre el evento de salud.
Last_Update	Fecha y hora de la última actualización del registro.
Lat	Latitud geográfica del lugar donde se registran los casos.
Long_	Longitud geográfica del lugar donde se registran los casos.
Confirmed	Número total acumulado de casos confirmados por COVID-19.

Deaths	Número total acumulado de muertes por COVID-19.
Recovered	Número total de personas que se han recuperado del COVID-19.
Active	Casos activos al momento del reporte (generalmente: Confirmed - Deaths - Recovered).
Combined_Key	Llave compuesta que concatena valores de ubicación (e.g., Admin2, Province_State, Country_Region).
Incident_Rate	Tasa de incidencia por cada 100,000 habitantes.
Case_Fatality_Ratio	Proporción de muertes entre los casos confirmados (%).
code_iso	Código ISO-3166-1 alfa-3 del país, utilizado para estandarización internacional (e.g., "COL" para Colombia).
code_key	Llave única construida concatenando code_iso, Province_State y Admin2, separadas por guiones.
Population	Total de habitantes del país o región, utilizado en el proceso de enriquecimiento.
Income_pc	Ingreso per cápita estimado del país o región (en USD), usado para análisis socioeconómicos.

1.3.3 Modelo relacional

Debido a que se trabajo con tablas solo para almacenar el resultado de la carga y las transformaciones para enriquecer los datos, la base de datos en sqlite no posee relaciones, en la siguiente imagen se observan las tablas que están almacenadas en sqlite en la base de datos ingestión.db .

 covid_data
A-Z FIPS
A-Z Admin2
A-Z Province_State
A-Z Country_Region
A-Z Last_Update
A-Z Lat
A-Z Long_
A-Z Confirmed
A-Z Deaths
A-Z Recovered
A-Z Active
A-Z Combined_Key
A-Z Incident_Rate
A-Z Case_Fatality_Ratio

 dat_trans
123 FIPS
A-Z Admin2
A-Z Province_State
A-Z Country_Region
A-Z Last_Update
123 Lat
123 Long_
123 Confirmed
123 Deaths
123 Recovered
123 Active
A-Z Combined_Key
123 Incident_Rate
123 Case_Fatality_Ratio
A-Z code_iso_x
A-Z code_key
A-Z code_iso_y
123 population_est
123 income_per_capita_est
A-Z income_group

1.4 Explicación detallada de cada componente y justificación de las herramientas utilizadas.

A continuación, se describe cada uno de los componentes empleados en el proceso de simulación de una infraestructura para Big Data.

1.4.1 Componentes Principales del Proyecto

- API Pública (<https://coronavirus.m.pipedream.net/>)

Función: Fuente de datos de contagio, muertes y recuperación de COVID-19.

Justificación: Proporciona datos en tiempo real estructurados en JSON, listos para ser consumidos por scripts automatizados.

- Script de Ingesta (ingestion.py)

Función: Conecta a la API, descarga los datos, selecciona las columnas clave y los almacena en un archivo tipo texto y SQLite.

Herramienta utilizada: Python + Pandas + SQLite

Justificación:

Python: Lenguaje versátil, ideal para scripts ETL.

Pandas: Facilita la manipulación tabular.

SQLite: Base de datos liviana, sin necesidad de servidor, ideal para entornos simulados.

- Base de Datos SQLite

Tablas:

covid_data: Datos originales descargados desde la API.

dat_trans: Tabla enriquecida con códigos ISO y datos poblacionales.

Justificación: Al ser un archivo .db, permite persistencia sin configuración compleja. Es fácilmente portable y se integra bien con Pandas.

- Script de Limpieza (cleaning.py)

Función:

Elimina duplicados.

Trata valores nulos.

Convierte tipos de datos.

Genera un archivo limpio cleaned_data.csv.

Justificación: Establece la base para un análisis confiable, eliminando inconsistencias del dataset.

- Script de Enriquecimiento (add_code_iso.py)

Función; Adiconar:

Columna code_iso usando codigo_iso_pais.csv.

Columna code_key (clave compuesta de región).

Realiza merge con datos poblacionales (pob_ipc.csv).

Guarda la tabla final enriquecida como dat_trans en la base SQLite.

Justificación: La fusión de datos contextuales (demográficos y económicos) con datos epidemiológicos permite un análisis más completo y útil para toma de decisiones.

- Automatización con GitHub Actions

Función: Ejecuta los scripts automáticamente al hacer push al repositorio.

Justificación: Simula un entorno de procesamiento en la nube, donde cada cambio activa automáticamente los procesos ETL. Esto replica buenas prácticas de CI/CD y despliegue continuo de pipelines de datos.

- Auditoría

Archivos:

audit_trans.txt: Registra columnas nuevas agregadas y resumen de fusiones (merge).

cleaning_report.txt: Resume limpieza.

Justificación: Garantiza trazabilidad, validación e integridad de los datos en cada fase.

- Repositorio GitHub

Función:

Control de versiones y simulación de la infraestructura remota.

Justificación: Permite mantener historial, colaboración y despliegue automático mediante acciones configuradas.

2 Conclusiones y recomendaciones

Con el desarrollo de la actividad se logró integrar de manera exitosa las fases de ingesta, limpieza y enriquecimiento de datos sobre el COVID-19, utilizando Python, SQLite y GitHub Actions para simular una infraestructura de Big Data en la nube. La automatización del flujo permitió obtener datos actualizados desde una API, procesarlos y enriquecerlos con información demográfica y económica, generando una base robusta para análisis avanzados. Se debe tener en cuenta, para futuras implementaciones, migrar a bases de datos más escalables, utilizar contenedores como Docker o similar, incorporar visualizaciones interactivas y ampliar el conjunto de datos con variables adicionales que permitan análisis más completos y predictivos.

3 Bibliografía

Pipedream. (n.d.). Coronavirus COVID-19 API (rawData endpoint). Recuperado de <https://coronavirus.m.pipedream.net/>