Proyecto

Segunda entrega

Reporte de proceso de EDA a el DATASET de

“Lung Carcer Prediction”

Realizado por:

Juliana Toro (2225658)

Luis Angel Garcia(2230177)

Antonio Cardenas Jurado(2230433)

DOCENTE: JAVIER ALEJANDRO VERGARA ZORRILLA

PROGRAMA DE INGENIERÍA DE DATOS E INTELIGENCIA ARTIFICIAL

FACULTAD DE INGENIERÍA

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE

SANTIAGO DE CALI

2025

Acerca del dataset

El dataset "Lung_cancer_prediction.csv" contiene un total de 221,000 registros y 24 columnas con información sobre factores de riesgo, condiciones ambientales y variables epidemiológicas relacionadas con el cáncer de pulmón en 25 distintos países. Incluye datos demográficos como el país de origen, tamaño poblacional, edad y género, así como factores de riesgo individuales, como tabaquismo (fumador, años fumando, cigarrillos por día), exposición pasiva al humo y antecedentes familiares. También se presentan variables médicas y ambientales, como el diagnóstico de cáncer de pulmón, etapa del cáncer (con una alta cantidad de valores nulos), tipo de adenocarcinoma, exposición a contaminación del aire y contaminación dentro del hogar, además del acceso a servicios de salud, detección temprana y tipo de tratamiento (otra columna con muchos valores nulos). A nivel epidemiológico, se incluyen tasas de prevalencia y mortalidad del cáncer de pulmón, así como el número anual de muertes por esta enfermedad en cada país. Finalmente, la variable "Developed_or_Developing" permite diferenciar entre países desarrollados y en desarrollo, lo que facilita estudios comparativos sobre el impacto de factores ambientales y acceso a la salud en la incidencia del cáncer de pulmón.

Análisis del dataset

Conexión con MySQL y creacion de base de datos

Para esta parte se crearon variables como conn y cursor para poder permitir la ejecución de comandos desde Jupyter y poder realizar la conexión, además de realizar las diferentes funciones básicas de una conexión con postgres desde el mismo WSL como creación de usuario y base de datos, conexión con base de datos, creación de tablas de los datos limpios y del modelo dimencional.

```python
import psycopg2
import pandas as pd

def crear_usuario_y_bd(admin_user="postgres", admin_pass="admin"):
    conn = psycopg2.connect(
        dbname="postgres",
        user=admin_user,
        password=admin_pass,
        host="localhost"
    )
    conn.autocommit = True
    cur = conn.cursor()

    cur.execute("SELECT 1 FROM pg_roles WHERE rolname='etl_user'")
    if not cur.fetchone():
        cur.execute("CREATE USER etl_user WITH PASSWORD 'etl_pass'")
        print("Usuario 'etl_user' creado.")
    else:
        print("El usuario 'etl_user' ya existe.")

    cur.execute("SELECT 1 FROM pg_database WHERE datname='lung_cancer_database'")
    if not cur.fetchone():
        cur.execute("CREATE DATABASE lung_cancer_database OWNER etl_user")
        print("Base de datos 'lung_cancer_database' creada.")
    else:
        print("La base de datos 'lung_cancer_database' ya existe.")

    cur.close()
    conn.close()

def conectar_etl():
    return psycopg2.connect(
        dbname="lung_cancer_database",
        user="etl_user",
        password="etl_pass",
        host="localhost"
    )
```

(Foto de la configuración de “postgres_setup.py”)

Después se prosiguió a crear la base de datos en MySQL desde comandos en jupyter y a conectarse a esa base de datos.

```
    cur.execute("CREATE DATABASE lung_cancer_database OWNER etl_user")
    print("Base de datos 'lung_cancer_database' creada.")
lse:
```

(Mensaje de confirmación)

<u>Análisis del dataset</u>

Se realizo la cargar del dataset desde MySQL en un dataframe.

```
cursor.execute("SELECT * FROM lung_cancer_dirty")

columns = [col[0] for col in cursor.description]
datas= cursor.fetchall()
df = pd.DataFrame(datas, columns=columns)
```
[6] ✓ 1.1s

(lectura y confirmación de los datos en un nuevo dataframe)

Se realizo análisis de los 5 primeros registros para tener un panorama mas claro acerca de los datos

```
print("\nPrimeras filas del dataset:")
print(df.head())
```
[7] ✓ 0.0s

(código de la impresión)

```
Primeras filas del dataset:
   id       Country  Population_Size  Age  Gender Smoker  Years_of_Smoking  \
0   1         China             1400   80    Male    Yes                30
1   2          Iran               84   53    Male     No                 0
2   3        Mexico              128   47    Male    Yes                12
3   4     Indonesia              273   39  Female     No                 0
4   5  South Africa               59   44  Female     No                 0

   Cigarettes_per_Day Passive_Smoker Family_History  ...  \
0                  29            No             No  ...
1                   0           Yes             No  ...
2                   6           Yes             No  ...
3                   0            No            Yes  ...
4                   0           Yes             No  ...

  Air_Pollution_Exposure Occupational_Exposure  Indoor_Pollution  \
0                    Low                  Yes               No
1                    Low                  Yes               No
2                 Medium                   No               No
3                    Low                   No               No
4                 Medium                  Yes               No

  Healthcare_Access Early_Detection Treatment_Type Developed_or_Developing  \
0              Poor              No      TEMP_NULL                Developing
...
3                    40000                        0.75            0.0
4                    15000                        2.44            0.0
```

(Primeros 5 registros)

A traves de los 5 primeros datos se dio a conocer que si habian datos nulos a lo cual se investigo el porque de la existencia de estos y se determino que únicamente era dos columnas, “Treatment_Type” y “Cancer_Stage”.

Se prosiguió a conocer los nulos y se confirmo la hipótesis de los datos nulos.

```
nulos=(df == "TEMP_NULL").sum()
print(f'Los nulos de cada columna son:', nulos )
```

(Conteo de los nulos por columnas)

```
Los nulos de cada columna son: id
Country                              0
Population_Size                      0
Age                                  0
Gender                               0
Smoker                               0
Years_of_Smoking                     0
Cigarettes_per_Day                   0
Passive_Smoker                       0
Family_History                       0
Lung_Cancer_Diagnosis                0
Cancer_Stage                    211671
Survival_Years                       0
Adenocarcinoma_Type                  0
Air_Pollution_Exposure               0
Occupational_Exposure                0
Indoor_Pollution                     0
Healthcare_Access                    0
Early_Detection                      0
Treatment_Type                  213968
Developed_or_Developing              0
Annual_Lung_Cancer_Deaths            0
Lung_Cancer_Prevalence_Rate          0
Mortality_Rate                       0
dtype: int64
```

(resultado de los nulos por columnas)

Gracias a esto se determino que los datos nulos eran debido a registros de personas que no tenían tratamiento de cáncer de pulmón debido a que no poseían esta enfermedad y a su vez no se encontraba en alguna etapa del cáncer.

Después se realizo el conteo de valores duplicados.

```
print("\nCantidad de filas duplicadas:", df.duplicated().sum())
```

(lectura y conteo de valores duplicados)

```
Cantidad de filas duplicadas: 0
```

(resultados de filas duplicadas)

Se realizo análisis de los valores únicos.

```
print("\nValores únicos en la columna 'Gender':", df['Gender'].unique())
print("Valores únicos en la columna 'Smoker':", df['Smoker'].unique())
print("Valores únicos en la columna 'Passive_Smoker':", df['Passive_Smoker'].unique())
print("Valores únicos en la columna 'Family_History':", df['Family_History'].unique())
print("Valores únicos en la columna 'Lung_Cancer_Diagnosis':", df['Lung_Cancer_Diagnosis'].unique())
print("Valores únicos en la columna 'Cancer_Stage':", df['Cancer_Stage'].unique())
print("Valores únicos en la columna 'Adenocarcinoma_Type':", df['Adenocarcinoma_Type'].unique())
print("Valores únicos en la columna 'Air_pollution_exposure':", df['Air_Pollution_Exposure'].unique())
print("Valores únicos en la columna 'Occupational_Exposure':", df['Occupational_Exposure'].unique())
print("Valores únicos en la columna 'Indoor_pollution':", df['Indoor_Pollution'].unique())
print("Valores únicos en la columna 'Healtcare_Access':", df['Healthcare_Access'].unique())
print("Valores únicos en la columna 'Early_Detection':", df['Early_Detection'].unique())
print("Valores únicos en la columna 'Developed_or_Developing':", df['Developed_or_Developing'].unique())
```

(Valores evaluados)

```
Valores únicos en la columna 'Gender': ['Male' 'Female']
Valores únicos en la columna 'Smoker': ['Yes' 'No']
Valores únicos en la columna 'Passive_Smoker': ['No' 'Yes']
Valores únicos en la columna 'Family_History': ['No' 'Yes']
Valores únicos en la columna 'Lung_Cancer_Diagnosis': ['No' 'Yes']
Valores únicos en la columna 'Cancer_Stage': ['TEMP_NULL' 'Stage 1' 'Stage 2' 'Stage 3' 'Stage 4']
Valores únicos en la columna 'Adenocarcinoma_Type': ['Yes' 'No']
Valores únicos en la columna 'Air_pollution_exposure': ['Low' 'Medium' 'High']
Valores únicos en la columna 'Occupational_Exposure': ['Yes' 'No']
Valores únicos en la columna 'Indoor_pollution': ['No' 'Yes']
Valores únicos en la columna 'Healtcare_Access': ['Poor' 'Good']
Valores únicos en la columna 'Early_Detection': ['No' 'Yes']
Valores únicos en la columna 'Developed_or_Developing': ['Developing' 'Developed']
```

(Resultado de los valores)

A excepcion de los valores “TEMP_NULL” no se encontraron valores extraños.

Durante el análisis se encontraron valores de cigarrillos por día un poco altos se decide conocer los limites superiores e inferiores de esa columna para conocer si hay valores atípicos a través del IQR.

```
min_cigarettes = df["Cigarettes_per_Day"].min()
max_cigarettes = df["Cigarettes_per_Day"].max()

print(f"\nLos minimos cigarrillos al dia encontrados fueron: ",min_cigarettes)
print(f"Los maximos cigarrillos al dia encontrados fueron: ",max_cigarettes)

smokers = (df["Cigarettes_per_Day"] > 0).sum()
print(f"\nNúmero de personas que fuman son fumadoras: {smokers}")

df_smokers = df[df["Cigarettes_per_Day"]>0]

q1 = df_smokers["Cigarettes_per_Day"].quantile(0.25)
q3 = df_smokers["Cigarettes_per_Day"].quantile(0.75)
iqr=q3-q1

limit_inf = max(1, q1- 1.5 * iqr)
limit_sup = q3 + 1.5 * iqr

atipic_value= df_smokers[(df_smokers["Cigarettes_per_Day"] < limit_inf) | (df_smokers["Cigarettes_per_Day"]>limit_sup)]

print(f"\nQ1: {q1}, Q3: {q3}, IQR: {iqr}")
print(f"Límite inferior ajustado: {limit_inf}")
print(f"Límite superior: {limit_sup}")
print(f"Número de valores atípicos: {atipic_value.shape[0]}")
```

(análisis de los rangos)

```
Los minimos cigarrillos al dia encontrados fueron:  0
Los maximos cigarrillos al dia encontrados fueron:  30

Número de personas que fuman son fumadoras: 88341

Q1: 11.0, Q3: 24.0, IQR: 13.0
Límite inferior ajustado: 1
Límite superior: 43.5
Número de valores atípicos: 0


Los valores Maximos de cigarrillos por dia pueden son un poco altos, se decidio conocer el limte superior
```

(resultado del analisis)

No se encontraron valores que se salieran de los limites y se ajusto el limite inferior a 1 ya que un fumador se considera fumador cuando fuma uno o más al día.

Se verifico la edad y no se encontraron valores máximos y mínimos extraños.

```
age_max = df["Age"].max()
age_min = df["Age"].min()
print(f"\nLa edad maxima es: ", age_max)
print(f"La edad minima es: ", age_min)
```

(análisis de la edad)

```
La edad maxima es:  85
La edad minima es:  20
```

(Resultado)

Limpieza y conversiones necesarias

Inicialmente se determinaron las columnas necesarias para realizar la visualización de los datos y los analisis que se deseaban realizar

```
query = "SELECT * FROM lung_cancer_dirty"

df = pd.read_sql(query, conn)

df_selected = df[["id","Age","Country","Lung_Cancer_Prevalence_Rate","Smoker","Years_of_Smoking","Cigarettes_per_Day","Passive_Smoker","Lung_Cancer_Diagnosis","Healthcare_Access","Early_Detection","Survival_
print(df_selected.info())
```

(Foto de la selección)

```
Data columns (total 21 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   id                          220632 non-null  int64
 1   Age                         220632 non-null  int64
 2   Country                     220632 non-null  object
 3   Lung_Cancer_Prevalence_Rate 220632 non-null  float64
 4   Smoker                      220632 non-null  object
 5   Years_of_Smoking            220632 non-null  int64
 6   Cigarettes_per_Day          220632 non-null  int64
 7   Passive_Smoker              220632 non-null  object
 8   Lung_Cancer_Diagnosis       220632 non-null  object
 9   Healthcare_Access           220632 non-null  object
 10  Early_Detection             220632 non-null  object
 11  Survival_Years              220632 non-null  int64
 12  Developed_or_Developing     220632 non-null  object
 13  Mortality_Rate              220632 non-null  float64
 14  Annual_Lung_Cancer_Deaths   220632 non-null  int64
 15  Air_Pollution_Exposure      220632 non-null  object
 16  Occupational_Exposure       220632 non-null  object
 17  Indoor_Pollution            220632 non-null  object
 18  Family_History              220632 non-null  object
 19  Treatment_Type              220632 non-null  object
 20  Cancer_Stage                220632 non-null  object
dtypes: float64(2), int64(6), object(13)
```

(Tabla resultante)

Ya habiendo seleccionado se decidio cambiar los nulos por “none” con el fin de rellenar esos espacios

```python
nulos=(df_selected == "TEMP_NULL").sum()
print(f'Antes los nulos de cada columna son:', nulos )

df_selected.replace("TEMP_NULL", None, inplace=True)

nulos=(df_selected == "TEMP_NULL").sum()
print(f'Despues los nulos de cada columna son:', nulos )
```

(análisis y remplazo de los nulos)

```
Antes los nulos de cada columna son: id
Age                                    0
Country                                0
Lung_Cancer_Prevalence_Rate            0
Smoker                                 0
Years_of_Smoking                       0
Cigarettes_per_Day                     0
Passive_Smoker                         0
Lung_Cancer_Diagnosis                  0
Healthcare_Access                      0
Early_Detection                        0
Survival_Years                         0
Developed_or_Developing                0
Mortality_Rate                         0
Annual_Lung_Cancer_Deaths              0
Air_Pollution_Exposure                 0
Occupational_Exposure                  0
Indoor_Pollution                       0
Family_History                         0
Treatment_Type                    213968
Cancer_Stage                      211671
```

(Antes de la limpieza)

```
Despues los nulos de cada columna son: id
Age                           0
Country                       0
Lung_Cancer_Prevalence_Rate   0
Smoker                        0
Years_of_Smoking              0
Cigarettes_per_Day            0
Passive_Smoker                0
Lung_Cancer_Diagnosis         0
Healthcare_Access             0
Early_Detection               0
Survival_Years                0
Developed_or_Developing       0
Mortality_Rate                0
Annual_Lung_Cancer_Deaths     0
Air_Pollution_Exposure        0
Occupational_Exposure         0
Indoor_Pollution              0
Family_History                0
Treatment_Type                0
Cancer_Stage                  0
dtype: int64
```

(después de la limpieza)

Se convirtio a booleanos los de valores “TRUE” y “FALSE”.

```
df_selected["Early_Detection"] = df_selected["Early_Detection"].map({"Yes": True, "No": False})
df_selected["Smoker"] = df_selected["Smoker"].map({"Yes": True, "No": False})
df_selected["Passive_Smoker"] = df_selected["Passive_Smoker"].map({"Yes": True, "No": False})
df_selected["Lung_Cancer_Diagnosis"] = df_selected["Lung_Cancer_Diagnosis"].map({"Yes": True, "No": False})
df_selected["Occupational_Exposure"] = df_selected["Occupational_Exposure"].map({"Yes": True, "No": False})
df_selected["Indoor_Pollution"] = df_selected["Indoor_Pollution"].map({"Yes": True, "No": False})
df_selected["Family_History"] = df_selected["Family_History"].map({"Yes": True, "No": False})
print(df_selected.info)
```

(Conversión de los booleanos)

Carga de datos crudos

Por ultimo habiendo realizado todos los pasos de conversión, limpieza y selección se realizo la carga de esos nuevos datos en una tabla nueva dentro de esa misma base de datos.

```
cursor.execute("""
CREATE TABLE IF NOT EXISTS lung_cancer_cleaned (
                id INT AUTO_INCREMENT NOT NULL PRIMARY KEY,
                Age INT,
                Country VARCHAR(20),
                Lung_Cancer_Prevalence_Rate FLOAT,
                Smoker BOOLEAN,
                Years_of_Smoking INT,
                Cigarettes_per_Day INT,
                Passive_Smoker BOOLEAN,
                Lung_Cancer_Diagnosis BOOLEAN,
                Healthcare_Access VARCHAR(20),
                Early_Detection BOOLEAN,
                Survival_Years INT,
                Developed_or_Developing VARCHAR(20),
                Mortality_Rate FLOAT,
                Annual_Lung_Cancer_Deaths INT,
                Air_Pollution_Exposure VARCHAR(20),
                Occupational_Exposure BOOLEAN,
                Indoor_Pollution BOOLEAN,
                Family_History BOOLEAN,
                Treatment_Type VARCHAR(20),
                Cancer_Stage VARCHAR(20));
""")
```

(Nueva tabla de datos limpios)

```
cursor.execute("SELECT COUNT(*) FROM lung_cancer_cleaned")
resultado = cursor.fetchone()
if resultado[0] == 0:
    for _, row in df_selected.iterrows():
        cursor.execute("""
                INSERT INTO lung_cancer_cleaned (Age,Country,Lung_Cancer_Prevalence_Rate,Smoker,Years_of_Smoking,Cigarettes_per_Day,Passive_Smoker,Lung_Cancer_Diagnosis,Healthcare_Access,Early_Detectio
                VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)
            """, (row['Age'],row['Country'],row['Lung_Cancer_Prevalence_Rate'],row['Smoker'],row['Years_of_Smoking'],row['Cigarettes_per_Day'],row['Passive_Smoker'],row['Lung_Cancer_Diagnosis'],row['H
    conn.commit()
    print("Dato cargados correcctamente")
else:
    conn.commit()
    print("No es necesario volver a cargar los datos")
```

(Inserción de los datos)

```
Dato cargados correcctamente
```

(resultado)

Automatización del proceso ETL con Airflow

Para la automatización del flujo de datos del proyecto, se utilizó Apache Airflow. Se diseñó un DAG llamado 'etl_lung_cancer' con la finalidad de orquestar el proceso completo de ETL (Extracción, Transformación y Carga) de los datos utilizados en el análisis.

El DAG contiene las siguientes tareas:
- extract_task: extrae los datos desde una base de datos PostgreSQL.

```python
from airflow.decorators import task
import pandas as pd
import psycopg2

DB_PARAMS = {
    "host": "localhost",
    "database": "lung_cancer_database",
    "user": "postgres",
    "password": "admin"
}

@task()
def extract_from_postgres():
    """Extrae datos desde la tabla lung_cancer_post en PostgreSQL y los retorna como dict"""
    conn = psycopg2.connect(**DB_PARAMS)
    df = pd.read_sql("SELECT * FROM lung_cancer_post", conn)
    conn.close()
    return df.to_dict()
```

(Extract_task)

- transform: realiza las transformaciones necesarias, incluyendo limpieza y estandarización de variables.

```python
from airflow.decorators import task
import pandas as pd
import logging

@task()
def transform_data(data_dict):
    """
    Recibe un diccionario de datos (output de extract), lo convierte en DataFrame,
    limpia los datos y retorna como JSON (serializable para Airflow).
    """
    df = pd.DataFrame.from_dict(data_dict)
    df.fillna("any", inplace=True)
    logging.info(f"Columnas después de limpieza: {df.columns.tolist()}")
    return df.to_json()
```

(transform)

- load_dimensions: carga las dimensiones necesarias para el modelo analítico.

```python
from airflow.decorators import task
import pandas as pd
import psycopg2
import logging

DB_PARAMS = {
    "host": "localhost",
    "database": "lung_cancer_database",
    "user": "postgres",
    "password": "admin"
}

@task()
def load_dimensions(json_data):
    df = pd.read_json(json_data)
    logging.info(f"Columnas recibidas en load_dimensions: {df.columns.tolist()}")
    logging.info("Primeras filas:\n%s", df.head())

    conn = psycopg2.connect(**DB_PARAMS)
    cur = conn.cursor()


    df_pais = df.groupby("country").agg({
        "developed_or_developing": "first",
        "annual_lung_cancer_deaths": "mean"
    }).reset_index()

    for _, row in df_pais.iterrows():
        cur.execute("""
            INSERT INTO dim_pais (country, developed_or_developing, annual_deaths)
            VALUES (%s, %s, %s)
        """, (row["country"], row["developed_or_developing"], row["annual_lung_cancer_deaths"]))


    df_salud = df[["healthcare_access", "early_detection", "treatment_type", "cancer_stage"]].drop_duplicates()

    for _, row in df_salud.iterrows():
        cur.execute("""
            INSERT INTO dim_salud (healthcare_access, early_detection, treatment_type, cancer_stage)
            VALUES (%s, %s, %s, %s)
        """, (row["healthcare_access"], row["early_detection"], row["treatment_type"], row["cancer_stage"]))
```

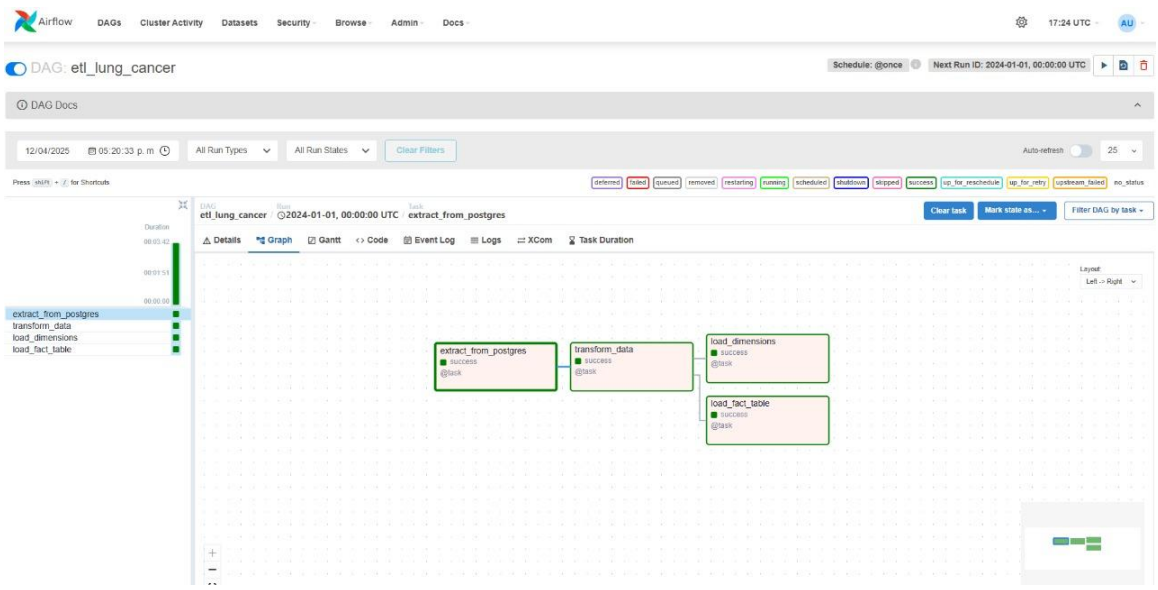
(fragmento de load_dimension)

- load_fact: carga los datos procesados en la tabla de hechos para su posterior análisis.

```python
from airflow.decorators import task
import pandas as pd
import psycopg2
import logging

DB_PARAMS = {
    "host": "localhost",
    "database": "lung_cancer_database",
    "user": "postgres",
    "password": "admin"
}

@task()
def load_fact_table(json_data):
    df = pd.read_json(json_data)
    logging.info(f"Columnas recibidas en load_fact_table: {df.columns.tolist()}")

    conn = psycopg2.connect(**DB_PARAMS)
    cur = conn.cursor()

    for _, row in df.iterrows():
        cur.execute("""
            INSERT INTO hechos_persona (
                edad, tasa_prevalencia, tasa_mortalidad, survival_years,
                id_pais, id_salud, id_exposicion, id_fumador,
                id_diagnostico, id_stage, id_tratamiento, id_historia
            ) VALUES (
                %s, %s, %s, %s,
                (SELECT id_pais FROM dim_pais WHERE country = %s LIMIT 1),
                (SELECT id_salud FROM dim_salud WHERE healthcare_access = %s AND early_detection = %s AND treatment_type = %s AND cancer_stage = %s LIMIT 1),
                (SELECT id_exposicion FROM dim_exposicion WHERE air_pollution_exposure = %s AND occupational_exposure = %s AND indoor_pollution = %s AND family_history = %s LIMIT 1),
                (SELECT id_fumador FROM dim_fumador WHERE fumador = %s AND fumador_pasivo = %s AND anios_fumando = %s AND cigarrillos_por_dia = %s LIMIT 1),
                (SELECT id_diagnostico FROM dim_diagnostico WHERE descripcion = %s LIMIT 1),
                (SELECT id_stage FROM dim_cancer_stage WHERE descripcion = %s LIMIT 1),
                (SELECT id_tratamiento FROM dim_tratamiento WHERE descripcion = %s LIMIT 1),
                (SELECT id_historia FROM dim_historia_familiar WHERE descripcion = %s LIMIT 1)
            )
        """, (
            row["age"], row["lung_cancer_prevalence_rate"], row["mortality_rate"], row["survival_years"],
            row["country"],
            row["healthcare_access"], row["early_detection"], row["treatment_type"], row["cancer_stage"],
            row["air_pollution_exposure"], row["occupational_exposure"], row["indoor_pollution"], row["family_history"],
            row["smoker"], row["passive_smoker"], row["years_of_smoking"], row["cigarettes_per_day"],
            row["lung_cancer_diagnosis"],
            row["cancer_stage"],
            row["treatment_type"],
            row["family_history"]
        ))
```

(fragmento de load_fact

A continuación, se presentan las ejecuciones exitosas y el orden de las tareas asignadas para el DAG:



Modelo Dimensional del Proyecto

A continuación, se presentan las tablas que conforman el modelo dimensional diseñado para el análisis del cáncer de pulmón.

Dim_pais

Esta dimensión contiene información sobre el país de origen del paciente, lo cual permite analizar la prevalencia del cáncer de pulmón en diferentes contextos nacionales. También incluye la clasificación económica de cada país, permitiendo la comparación entre países desarrollados y en desarrollo. Además, se registra el número de muertes anuales por cáncer de pulmón (campo anual_deaths), lo cual es útil para estudios comparativos entre países.

Dim_Pais	
Campo	**Tipo de dato**
id_pais	(PK)
country	VARCHAR
developed_or_developing	VARCHAR
annual_deaths	FLOAT

Dim_salud

Representa las condiciones del sistema de salud del paciente y su acceso al mismo. Esta dimensión incluye detalles sobre si el paciente tuvo acceso a la salud (campo Healthcare_Access), si recibió detección temprana de cáncer de pulmón (campo Early_Detection), qué tipo de tratamiento se le administró (campo Treatment_Type), y en qué etapa se encontraba el cáncer en el momento del diagnóstico (campo Cancer_Stage). Estos factores son esenciales para evaluar los resultados de salud y la respuesta al tratamiento.

Dim_Salud	
Campo	**Tipo de dato**
id_salud	(PK)
healthcare_access	VARCHAR
early_detection	(VARCHAR o BOOLEAN, según versión)
treatment_type	VARCHAR
cancer_stage	VARCHAR

Dim_exposicion

Agrupa los factores de exposición que pueden haber influido en el diagnóstico del cáncer de pulmón. Estos incluyen la exposición a la contaminación del aire (campo Air_Pollution_Exposure), exposición ocupacional (campo Occupational_Exposure), contaminación en interiores (campo Indoor_Pollution), y antecedentes familiares de cáncer (campo Family_History). Estos factores son importantes para evaluar los riesgos y ayudar en la identificación de patrones relacionados con la enfermedad.

Dim_Exposicion	
Campo	**Tipo de dato**
id_exposicion	(PK)
air_pollution_exposure	VARCHAR
occupational_exposure	BOOLEAN
indoor_pollution	BOOLEAN
family_history	BOOLEAN

Dim_fumador

Contiene información sobre los hábitos de tabaquismo del paciente. El tabaquismo es un factor de riesgo conocido para el cáncer de pulmón, y esta dimensión captura si el paciente es un fumador activo o pasivo (campo fumador_pasivo), la cantidad de años que ha fumado (campo anios_fumado), y la intensidad del hábito (cigarrillos por día, campo cigarrillos_por_dia).

Dim_Fumador	
Campo	**Tipo de dato**
id_fumador	(PK)
fumador	BOOLEAN
fumador_pasivo	BOOLEAN
anios_fumado	INTEGER
cigarrillos_por_dia	INTEGER

Dim_diagnostico

Esta dimensión indica si el paciente fue diagnosticado con cáncer de pulmón (campo descricion). Es esencial para determinar la prevalencia y para vincular los diagnósticos con otros factores, como la exposición y el acceso al sistema de salud.

Dim_diagnostico	
Campo	**Tipo de dato**
id_diagnostico	(PK)
descripcion	BOOLEAN

Dim_tratamiento

Contiene información sobre el tipo de adenocarcinoma diagnosticado en los pacientes que fueron diagnosticados con cáncer de pulmón (campo descripcion). Esta información es clave para determinar el tratamiento adecuado según el tipo de cáncer.

Dim_tratamiento	
Campo	**Tipo de dato**
id_tratamiento	(PK)
descripcion	VARCHAR

Dim_historia_familiar

Registra si el paciente tiene antecedentes familiares de cáncer de pulmón (campo descripcion), lo cual es un factor de riesgo importante. Este factor es relevante para la evaluación del riesgo y las decisiones de prevención y tratamiento.

Dim_historia_familiar	
Campo	**Tipo de dato**
id_historia	(PK)
descripcion	BOOLEAN

Hechos_Persona

La tabla central de hechos que almacena medidas clave sobre el paciente, como la edad (campo edad), las tasas de prevalencia y mortalidad del cáncer de pulmón (campos tasa_prevalence y tasa_mortalidad), y los años de supervivencia (campo survival_years). Conecta todas las dimensiones a través de claves foráneas (FK). Es la base para realizar análisis agregados y comparativos de los datos.

Hechos_Persona	
Columna	**Tipo de Dato**
id	(PK)
edad	INTEGER
tasa_prevalencia	FLOAT
tasa_mortalidad	FLOAT
survival_years	FLOAT
id_pais	(FK → dim_pais)
id_salud	(FK → dim_salud)
id_exposicion	(FK → dim_exposicion)
id_fumador	(FK → dim_fumador)
id_diagnostico	(FK → dim_diagnostico)
id_stage	(FK → dim_cancer_stage)
id_tratamiento	(FK → dim_tratamiento)
id_historia	(FK → dim_historia_familiar)

Consideraciones sobre la API

Uno de los requisitos del proyecto era incluir el consumo de una API externa que complementara el análisis del cáncer de pulmón. Se realizó una búsqueda exhaustiva de APIs relacionadas con datos de salud pública, enfermedades respiratorias y estadísticas oncológicas.

Sin embargo, no se logró encontrar una API que cumpliera con los criterios mínimos necesarios para su implementación.

Queremos dejar claro que sí se buscó integrar una API, pero la no implementación se debió a la ausencia de una fuente de datos adecuada y funcional para el caso de uso específico del proyecto.