Proyecto De Fin De Semestre

José Vargas, Joshua Morocho, Jordy Navarro, Cristian Tambaco, Carlos Quintana
Escuela Politécnica Nacional, Escuela De Formación De Tecnólogos
Desarrollo De Software, Análisis De Datos
04/02/2024

DEFINICION DE CASO DE ESTUDIO

En este proyecto aplicamos los diferentes métodos que hemos aprendido a lo largo del semestre para poder realizar lo que es limpieza de datos, migración de datos, anális de los mismos con Python o Power BI, de diferentes temáticas propuestas.

OBJETIVO GENERAL

Realizar el análisis de datos con diferentes datasets de las temáticas propuestas.

OBJETIVOS ESPECIFICOS

Utilizar bases de datos relacionales y no relacionales para realizar tanto el análisis de datos como para la migración de datos, además del análisis de sentimientos.

Evaluar los resultados obtenidos de cada temática elegida por el grupo para poder sacar conclusiones y así poder generar casos de estudio.

Descripción del equipo de trabajo y actividades realizadas por cada uno.

El equipo trabajo se conformó de la siguiente manera:
Cada integrante eligió una temática de las propuestas para realizar el análisis de datos, que incluye limpieza, migración entre bases de datos, análisis de sentimientos, es decir todo el flujo de datos desde cargar los datasets a la respectiva base de datos, convertir de json a csv o viceversa, para luego pasar los datos al repositorio final que es SQL server y después proceder a crear dashboards con los datos obtenidos, para poder generar visualizaciones que permitan generar casos de estudio. Luego, para los demás requerimientos para realizar el proyecto, Cristian y Joshua se encargaban de la presentación en canva, Jordy de hacer los videos de Youtube,

José Vargas de realizar el informe y en general de ayudarnos en todo el proceso del proyecto.

Cronograma de actividades incluído el diagrama de Gantt

Gantt Project			
Tarea	**Asignado a**	**Inicio**	**Fin**
Extracción, migración de datos Eventos deportivos	José Vargas	26/01/2025	01/02/2025
Extracción, migración de datos Hobbies	Cristian Tambaco	26/01/2025	01/02/2025
Extracción, migración de datos Accidentes	Jordy Navarro	26/01/2025	01/02/2025
Extracción, migración de datos Eventos mundiales (Mercado de Valores)	Joshua Morocho	26/01/2025	01/02/2025
Extracción, migración de datos Restaurantes	Carlos Quintana	26/01/2025	01/02/2025
Videos de youtube	Jordy Navarro	30/01/2025	02/02/2025
Presentación (Canva)	Cristian Tambaco	30/01/2025	03/02/2025
Informe	José Vargas	31/01/2025	03/02/2025

Tabla de Gantt Project. En esta tabla se muestran las actividades designadas a cada integrante del grupo, así como la fecha de inicio y fin de la mism .

RECURSO Y HERRAMIENTAS UTILIZADAS.

Programas de bases de datos, entre los cuales se encuentran MySQL, SQLite, MongoDB, CouchDB, SQL server, Python, Power BI, estas dos últimas como herramientas de visualización, youtube para subir los videos, filmora para editar el video, canva para realizar la presentación, IA para momentos en que se necesitaba realizar consultas.

ARQUITECTURA DE LA SOLUCIÓN

La infraestructura del proyecto se realizó de la siguiente manera:

Bases de datos relacionales: SQL Server, SQLite, MySQL.
Bases de datos NoSQL: CouchDB, MongoDB.
Lenguaje de programación: Python (para manipulación de datos y análisis de sentimientos).

Herramientas de visualización: Power BI, Python Formato de datos: JSON y CSV.

Diagrama de arquitectura

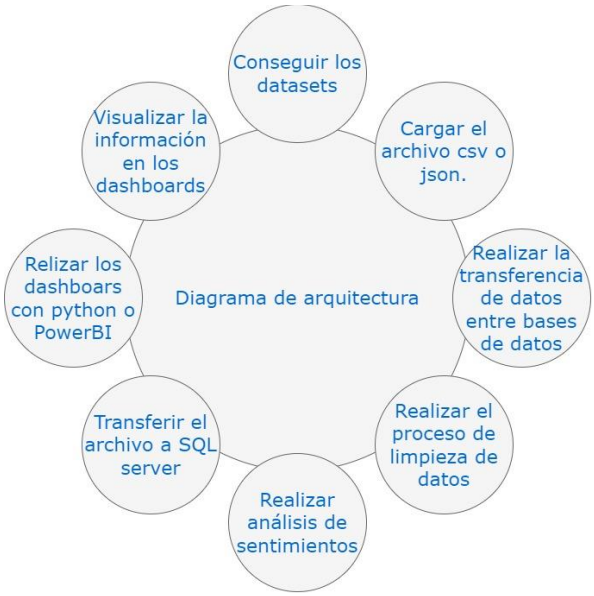
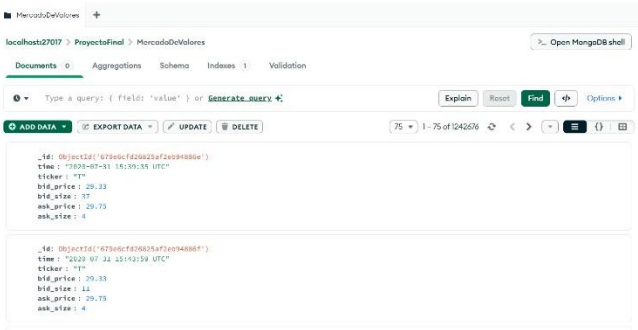


Fig.1. Se muestra el diagrama de arquitectura con el proceso realizado

EXTRACCIÓN DE DATOS

Para poder extraer información utilizamos como fuente de datos https://www.kaggle.com/ para obtener los datasets para realizar el informe. Los data sets usados son: datos.json, hobbies_datasets.json, student_preferences_dataset.json, accidentes.json, RestDB-MongoDB.json.

Carga de datos y migración entre bases de datos



Carga de archivo json en MongoDb

```
import json

# Cargar el archivo JSON
with open("JuegosOlimpicosDATASETS.json", "r", encoding="utf-8") as f:
    data = json.load(f)  # Asegúrate de que el JSON es un array

# Seleccionar solo los primeros 250,000 registros
data_filtrada = data[:250000]

# Guardar el nuevo JSON filtrado
with open("JuegosOlimpicosDATASETS_Recortado.json", "w", encoding="utf-8") as f:
    json.dump(data_filtrada, f, ensure_ascii=False, indent=4)

print("Archivo filtrado guardado como 'JuegosOlimpicosDATASETS_Recortado.json'")
```

```
Archivo filtrado guardado como 'JuegosOlimpicosDATASETS_Recortado.json'
```

Recorte de archivo json de gran cantidad de registros

```
import pymongo

import sqlite3
import pandas as pd

# Conectar a MongoDB
client = pymongo.MongoClient("mongodb://localhost:27017/")
db = client["mibase"]
collection = db["Juegos_Olimpicos"]

# Extraer los datos desde MongoDB
data = list(collection.find({}, {"_id": 0}))  # Para excluir id
df = pd.DataFrame(data)  # Convertir a DataFrame
```

Conexión con MongoDB

```
#conectar a sqlite

SQLITE_DB = "EventoDeportivo.db"
sqlite_conn = sqlite3.connect(SQLITE_DB)
cursor = sqlite_conn.cursor()

# Crear tabla en SQLite (ajustar según la estructura del JSON)
columns = ", ".join([f"{col} TEXT" for col in df.columns])
cursor.execute(f"CREATE TABLE IF NOT EXISTS JuegosOlimpicos ({columns})")
```

```
<sqlite3.Cursor at 0x2919b28b4c0>
```

```
# Insertar datos en SQLite
df.to_sql("JuegosOlimpicos", sqlite_conn, if_exists="replace", index=False)
```

Traspaso de datos a SQLite desde JupyterNoteboo

```
# Crear tabla en SQLite (ajustar según la estructura del JSON)
columns = ", ".join([f"{col} TEXT" for col in df.columns])
cursor.execute(f"CREATE TABLE IF NOT EXISTS JuegosOlimpicos ({columns})")
```

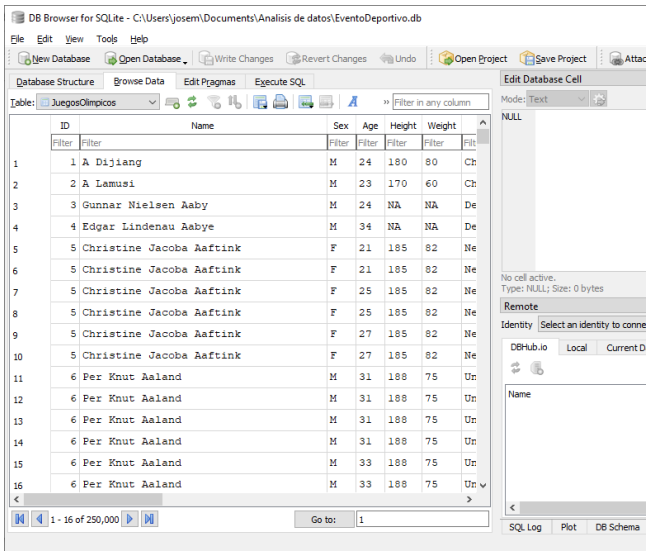
```
<sqlite3.Cursor at 0x2919b28b4c0>
```

```
# Insertar datos en SQLite
df.to_sql("JuegosOlimpicos", sqlite_conn, if_exists="replace", index=False)
```

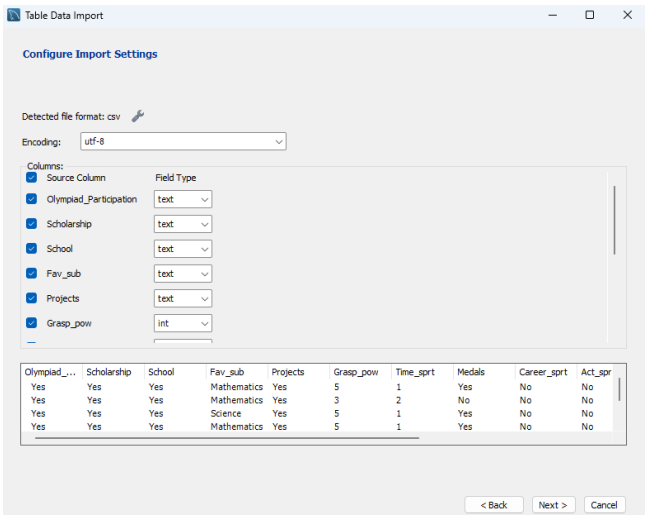
```
250000
```

```
#Convertir de json a csv
df.to_csv("JuegosOlimpicos.csv", index=False)
```

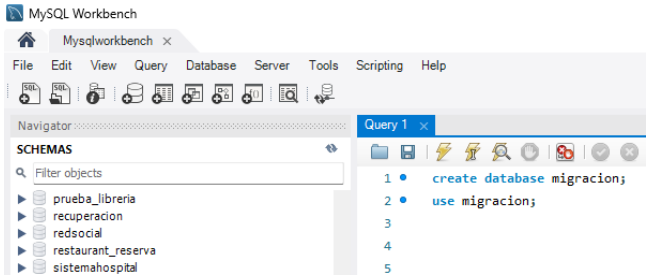
Ajuste de tablas, insertar datos en SQLite y conversión de json a csv.



Datos ingresados en SQLite correctamente.

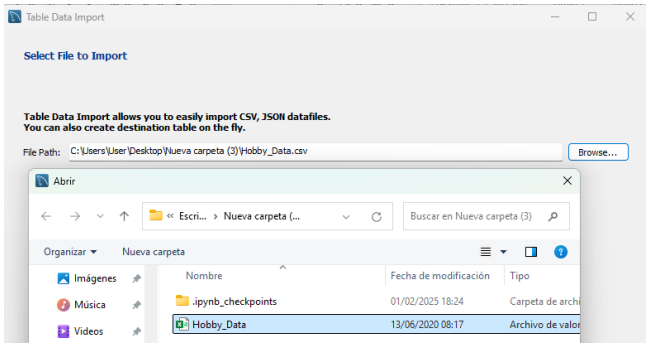


Configurar tabla

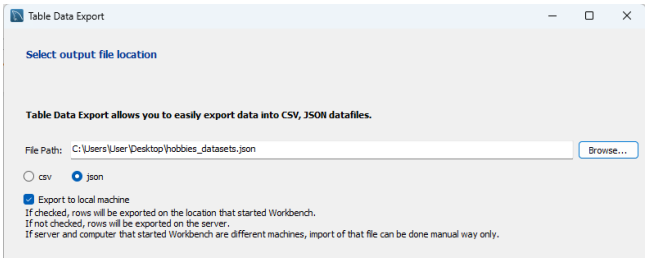


Creación de una base de datos en MySQL para realizar la importación

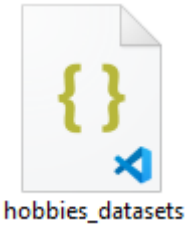
Visualización en MySQL Worckbench



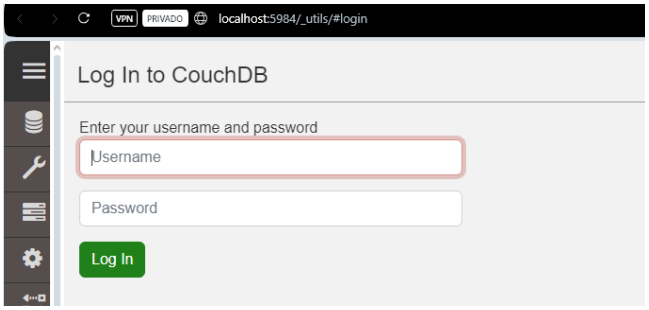
Importar el archivo csv



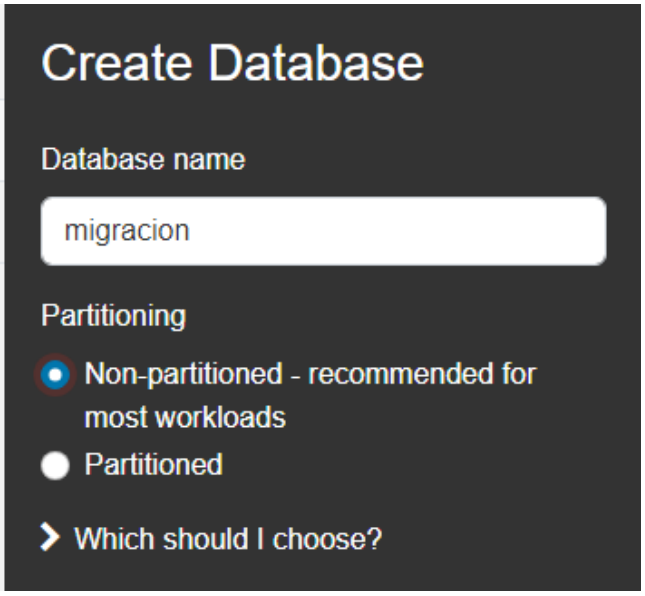
Exportar archivo en json



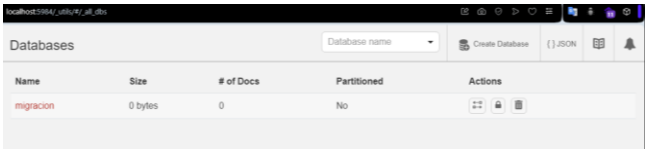
Archivo exportado a json



Para exportar a CouchDB hay que iniciar sesión



Se crea la base de datos



Base de datos creada

```python
import requests
import json
from requests.auth import HTTPBasicAuth

# URL de la base de datos en CouchDB
db_url = "http://localhost:5984/migracion/_bulk_docs"
```

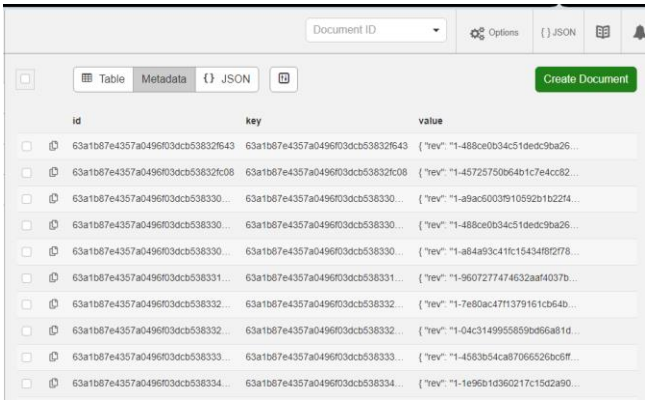
En jupyter notebook se importa las librerías yse establece la conexión

```python
# Credenciales de acceso a CouchDB
username = '[redacted]'
password = '[redacted]'
```

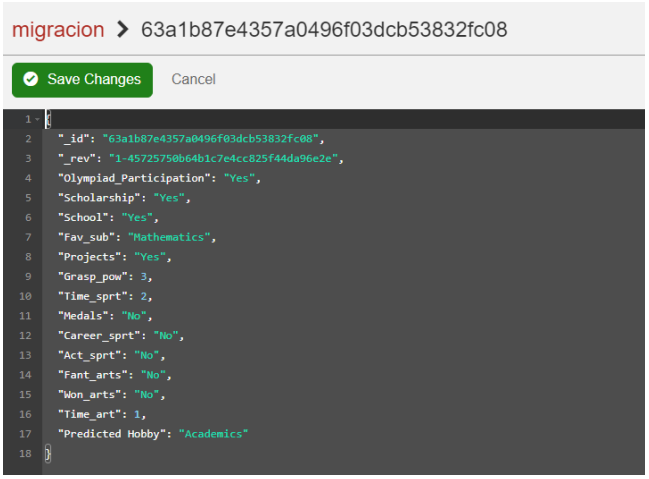
Usar las credenciales para acceder a CouchDB

```python
# Leer el archivo línea por línea y convertirlo a una lista de documentos
docs = []
with open('Hobby_Data.json', 'r') as f:
    for line in f:
        docs.append(json.loads(line.strip()))  # Cada línea es un documento JSON

# Empaquetar los documentos en un objeto JSON con la clave "docs"
data = {"docs": docs}

# Enviar la solicitud POST para insertar los documentos con autenticación básica
response = requests.post(db_url, json=data, auth=HTTPBasicAuth(username, password))

# Ver la respuesta
if response.status_code == 200:
    print("¡Documentos importados exitosamente!")
else:
    print(f"Error al importar documentos: {response.text}")
```

Leer el data set



Data set importado en CouchDB



Ejemplo de un Registro en CouchDB

ANÁLISIS DE INFORMACIÓN

Aquí mostramos los métodos usados para realizar limpieza de datos y análisis de sentimientos:

```
#limpieza de datos
# Eliminar filas duplicadas
df1 = df1.drop_duplicates()

# Verificar si se eliminaron duplicados
print(df1.shape)  # Verifica el número de filas y columnas después de eliminar
```

```
(248750, 16)
```

Limpieza de datos sencilla

```
# Mostrar las primeras filas para verificar la estructura
print(df1.head())
print(df1.columns)  # Verificar el nombre de las columnas
```

```
   ID                    Name Sex Age Height Weight            Team  NOC  \
0   1               A Dijiang   M  24    180     80           China  CHN
1   2                A Lamusi   M  23    170     60           China  CHN
2   3       Gunnar Nielsen Aaby   M  24     NA     NA         Denmark  DEN
3   4      Edgar Lindenau Aabye   M  34     NA     NA  Denmark/Sweden  DEN
4   5  Christine Jacoba Aaftink   F  21    185     82     Netherlands  NED

        Games  Year  Season       City          Sport  \
0  1992 Summer  1992  Summer  Barcelona     Basketball
1  2012 Summer  2012  Summer     London          Judo
2  1920 Summer  1920  Summer  Antwerpen       Football
3  1900 Summer  1900  Summer      Paris     Tug-Of-War
4  1988 Winter  1988  Winter    Calgary  Speed Skating

                          Event Medal  sentimiento
0       Basketball Men's Basketball    na         -1.0
1     Judo Men's Extra-Lightweight    na         -1.0
2           Football Men's Football    na         -1.0
3        Tug-Of-War Men's Tug-Of-War  gold          1.0
4  Speed Skating Women's 500 metres    na         -1.0
Index(['ID', 'Name', 'Sex', 'Age', 'Height', 'Weight', 'Team', 'NOC', 'Games',
       'Year', 'Season', 'City', 'Sport', 'Event', 'Medal', 'sentimiento'],
      dtype='object')
```

Verificamos la estructura de la tabla

```
# Función para asignar sentimiento según la medalla
def sentimiento_por_medalla(medalla):
    if isinstance(medalla, str):
        medalla = medalla.lower()  # Convertir a minúsculas para evitar problemas de formato
        if "gold" in medalla:
            return 1.0  # Oro
        elif "silver" in medalla:
            return 0.5  # Plata
        elif "bronze" in medalla:
            return 0.0  # Bronce
        else:
            return -1.0  # Sin medalla
    return -1.0  # Negativo si no es un valor válido

# Aplicar la función a la columna 'Medal' para crear la columna 'sentimiento'
df1["sentimiento"] = df1["Medal"].apply(sentimiento_por_medalla)

# Verificar los primeros registros después de aplicar la función
print(df1[['Medal', 'sentimiento']].head(50))
```

```
   Medal  sentimiento
0     na         -1.0
1     na         -1.0
2     na         -1.0
3   gold          1.0
4     na         -1.0
```

Al no contar con mayor información para sentimientos, se creó una función para medir sentimientos en base a la medalla obtenida, donde oro es muy positivo, plata es positivo, bronce es neutro y si no hay medalla es negativo.

```
# Verificar los valores únicos en la columna 'sentimiento'
print("Valores únicos en 'sentimiento':")
print(df1["sentimiento"].unique())
```

```
Valores únicos en 'sentimiento':
[-1.   1.   0.   0.5]
```

```
# Verificar el conteo de cada tipo de sentimiento
sentimientos_count = df1["sentimiento"].value_counts()
print("Conteo de Sentimientos:")
print(sentimientos_count)
```

```
Conteo de Sentimientos:
sentimiento
-1.0    212337
 1.0     12207
 0.0     12195
 0.5     12011
Name: count, dtype: int64
```

Verificación de las columnas para comprobar que todo esté correcto

```
import matplotlib.pyplot as plt

# Asegurarnos de que las categorías estén correctamente contadas
sentimientos_labels = ['Negativo (sin medalla)', 'Bronce (Neutro)', 'Plata (Positivo)', 'Oro (Muy Positivo)']
sentimientos_values = [
    sentimientos_count.get(-1.0, 0),  # Negativo (sin medalla)
    sentimientos_count.get(0.0, 0),   # Bronce
    sentimientos_count.get(0.5, 0),   # Plata
    sentimientos_count.get(1.0, 0)    # Oro
]

# 1. Crear gráfico de barras
plt.figure(figsize=(8,6))
plt.bar(sentimientos_labels, sentimientos_values, color=['red', 'orange', 'gray', 'gold'])

# Añadir título y etiquetas
plt.title('Distribución de Sentimientos según Medalla')
plt.xlabel('Tipo de Sentimiento')
plt.ylabel('Cantidad de Registros')

# Mostrar gráfico de barras
plt.show()

# 2. Crear gráfico de pie
plt.figure(figsize=(8,6))
plt.pie(sentimientos_values, labels=sentimientos_labels, autopct='%1.1f%%', startangle=90, colors=['red', 'orange', 'gray', 'gold'])

# Añadir título
plt.title('Distribución de Sentimientos según Medalla')

# Mostrar gráfico de pie
plt.show()
```

Análisis de sentimientos realizado con matplotlib

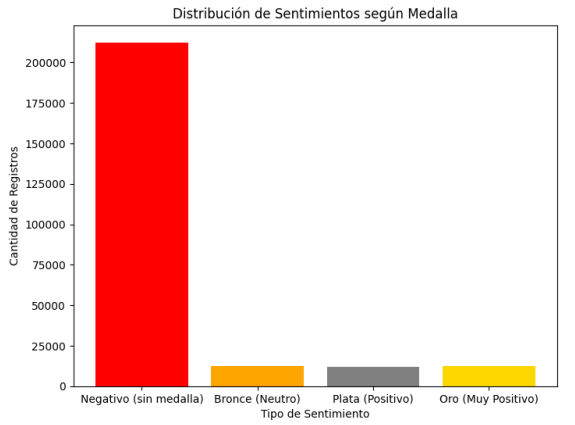


Gráfico de distribución de sentimientos por medalla

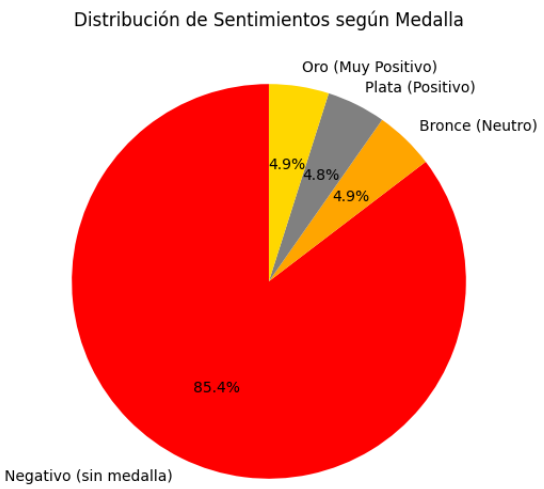
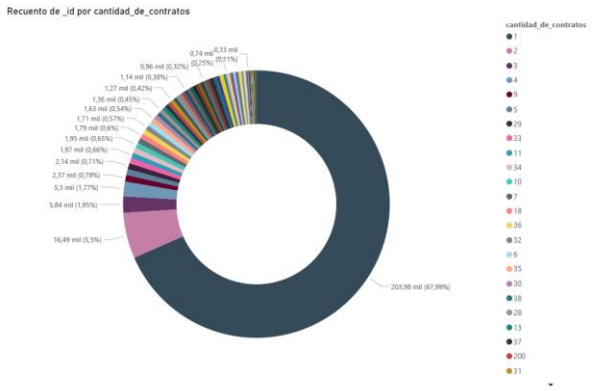


Gráfico de pastel sobre distribución de sentimientos por medalla.

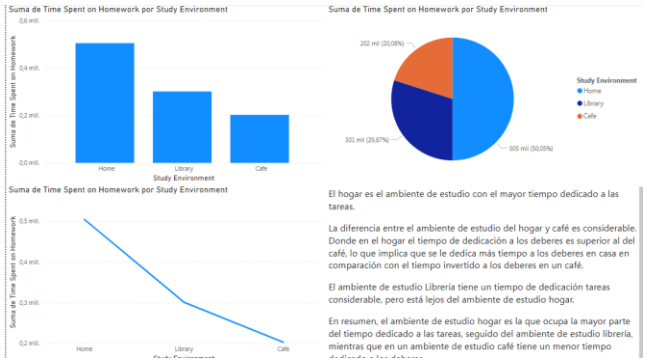
VISUALIZACIÓN DE INFORMACIÓN

Aquí se muestran algunos de los dashboards que se obtuvieron

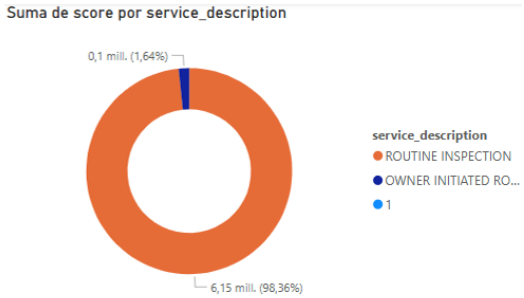


Mercado de valores

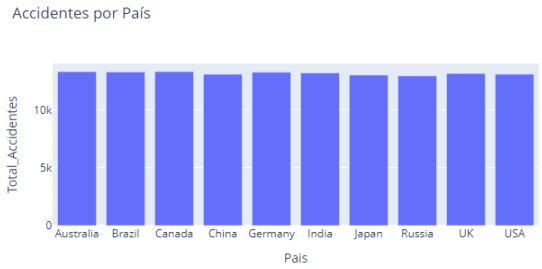
Suma de tiempo dedicado al aprendizaje



Tiempo dedicado a deberes



Inspecciones en rutina en los restaurantes



Accidentes por pais

RESULTADOS OBTENIDOS

Aquí analizamos parte de los resultados obtenidos.

En la parte de restaurantes basándonos en la información obtenida a través del dashboard, podemos realizar un análisis detallado sobre las inspecciones de rutina en los restaurantes. Los datos reflejan que la gran mayoría de estas inspecciones no son iniciadas voluntariamente por los propietarios, sino que deben ser llevadas a cabo por organismos de control.

Específicamente, solo el 1.64% de los restaurantes registrados en la base de datos han tomado la iniciativa de realizar una inspección de manera voluntaria y/o anticipada,

demostrando así un compromiso con la calidad del servicio y el cumplimiento de las normativas sanitarias. En contraste, el 98.36% de los establecimientos no han realizado esta acción por cuenta propia, lo que ha requerido la intervención de las entidades reguladoras para garantizar el cumplimiento de los estándares de higiene y seguridad.

En la parte de actividades y hobbies podemos ver que el tiempo dedicado al aprendizaje adicional en Lecturas representa una proporción significativa del total. Lo que sugiere que el tipo de interacción de lecturas es más significativa en términos de tiempo dedicado al aprendizaje fuera de la clase o como actividad principal.

También que la diferencia entre el ambiente de estudio del hogar y café es considerable. Donde en el hogar el tiempo de dedicación a los deberes es superior al del café, lo que implica que se le dedica más tiempo a los deberes en casa en comparación con el tiempo invertido a los deberes en un café. Además que el estatus socioeconómico bajo tiene un tiempo de aprendizaje cercano al estatus socioeconómico medio, pero sigue siendo algo inferior, mostrando que se le dedica solo una pequeña cantidad menos de tiempo comparado con el estatus socioeconómico medio.

CONCLUSIONES Y RECOMENDACIONES

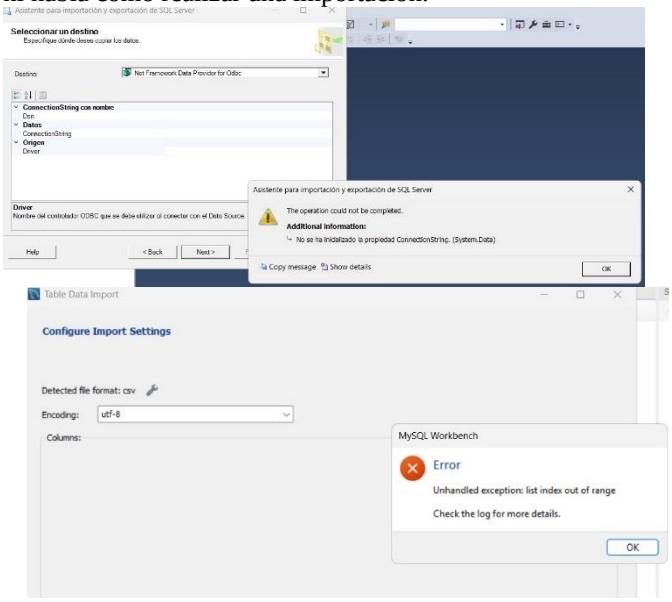
De lo que pudimos ver en los dashboards, en la parte de restaurantes pone en evidencia la necesidad de promover una mayor conciencia y responsabilidad en el sector gastronómico, incentivando a los propietarios a realizar inspecciones preventivas que contribuyan a mejorar la calidad y seguridad del servicio ofrecido a los clientes.

Por otra parte en los hobbies y actividades vemos que el tipo de interacción de lecturas es la actividad más preferida en cuanto al tiempo dedicado al aprendizaje adicional, seguida por la interacción por discusiones, mientras que en el trabajo grupal tiene el menor tiempo en aprendizaje adicional, además que el ambiente de estudio hogar es la que ocupa la mayor parte del tiempo dedicado a las tareas, seguido del ambiente de estudio librería, mientras que en un ambiente de estudio café tiene un menor tiempo dedicado a los deberes.

Como recomendación podemos decir que se debe realizar más análisis para ayudar a encontrar más problemáticas que se puedan dar y a de la misma forma ayudar a encontrar una solución para cada caso

DESAFÍOS Y PROBLEMAS ENCONTRADOS

Durante todo el desarrollo del proyecto, hubo algunos inconvenientes para poder realizar la migración de datos, ya que en ciertos casos no se quería conectar a la base de datos ni había como realizar una importación.



En el caso de análisis de sentimientos tocó crear una función para que en la parte de juegos olímpicos se pueda tomar la medalla obtenida como sentimientos para poder realizar lo solicitado.