

Earthquake Data Analysis using Python & MongoDB

Barbara Maier

Table of contents

1	My Project Title	2
1.1	Earthquake Data Analysis using Python & MongoDB	2
2	Summary	2
3	Requirements & Configuration	2
4	ELT Process	3
4.1	DB Setup	3
4.2	Extract	3
4.3	Load	4
4.4	Transform	5
5	Datastructure	7
5.0.1	Document Structure Diagram	7
5.0.2	Class Diagram	8
6	Data Analysis	9
6.0.1	Earthquakes Worldwide in the Past 10 Days	9
6.0.2	Earthquake Magnitude Scale	9
6.0.3	Recent Major Earthquakes	10
6.0.4	Global Earthquakes in the Past 30 Days	11
6.0.5	Daily Earthquakes	12
6.0.6	Spatial Distribution of the 70 Earthquakes Recorded on October 10, 2025	13
6.0.7	How many of the earthquakes on October 10, 2025, occurred around the Philippines?	14
6.0.8	Where was the main cluster of earthquakes on October 10, 2025?	15
6.0.9	Tsunami-Flagged Earthquakes in the Past 30 Days	16
7	Conclusions	17
8	Learnings	17

1 My Project Title

1.1 Earthquake Data Analysis using Python & MongoDB

2 Summary

This project analyzes recent earthquake data retrieved from the USGS API using Python and MongoDB. The workflow includes data extraction, transformation, and storage in a NoSQL database for efficient querying. Aggregations and visualizations are performed to explore earthquake frequency, magnitude distribution, and global locations. The analysis demonstrates how MongoDB's geospatial features and Python's data tools can support real-world seismic data insights.

3 Requirements & Configuration

```
# Check installed packages
!pip list 2>nul | findstr "pymongo pandas requests"
```

geopandas	1.1.1
pandas	2.3.3
pymongo	4.15.1
requests	2.32.5

```
# import packages
import pandas as pd
import requests
import datetime as dt
from pymongo import MongoClient
import cartopy.crs as ccrs
import cartopy.feature as cfeature
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
from pprint import pprint
```

```
# pandas configuration
pd.set_option('display.precision', 2)
pd.set_option('display.max_rows', 30)
```

```
# API and Database details
API_URL = "https://earthquake.usgs.gov/fdsnws/event/1/query"
CNX_STR = "mongodb://localhost:27017"
DB_NAME = "earthquakes_db"
COLL_NAME = "earthquakes"
START_UTC = datetime.utcnow() - timedelta(days=30)
```

4 ELT Process

4.1 DB Setup

```
# connection to MongoDB
client = MongoClient(CNX_STR)
db = client[DB_NAME]
quakes_col = db[COLL_NAME]

print("Connected with MongoDB (lokal)")
```

Connected with MongoDB (lokal)

```
# remove all existing documents
quakes_col.drop()
quakes_col.count_documents({})
```

0

4.2 Extract

The dataset is extracted from the USGS Earthquake API. Only the last 30 days are requested to keep the dataset manageable.

Each record contains details such as magnitude, location, and time of the earthquake. The parameter `minmagnitude = 3.0` filters out very small, irrelevant events.

```
# get url
url = API_URL

# earthquakes from the last 30 days
start = (dt.datetime.utcnow() - dt.timedelta(days=30)).strftime("%Y-%m-%d")
end = dt.datetime.utcnow().strftime("%Y-%m-%d")

# define API request parameters
params = {
    "format": "geojson",
    "starttime": start,
    "endtime": end,
    "minmagnitude": 3,
}

# send request to the API, load the JSON response, and inspect the result
response = requests.get(url, params=params)
data = response.json()
quakes = data["features"]
```

```

# print info for check
print(f"Number of records: {len(quakes)}")

# format long key list with line breaks (wrap text)
from textwrap import fill
if quakes:
    keys = ", ".join(quakes[0]["properties"].keys())
    print(" All keys:")
    print(fill(keys, width=80))
else:
    print("No earthquake data found.")

```

Number of records: 1069

All keys:
mag, place, time, updated, tz, url, detail, felt, cdi, mmi, alert, status,
tsunami, sig, net, code, ids, sources, types, nst, dmin, rms, gap, magType,
type, title

The output shows that the data was successfully loaded from the USGS Earthquake API.

- **Number of records:** is the total number of earthquakes found in the last 30 days with magnitude ≥ 3.0 .
- **All keys:** main data fields for each earthquake, such as mag (magnitude), place (location) or time (UTC time).

4.3 Load

```

# convert API earthquake features into MongoDB document structure
docs = []
for f in quakes:
    p = f["properties"]
    g = f.get("geometry", {}) or {}
    coords = g.get("coordinates") or [None, None, None]
    docs.append({
        "_id": f.get("id"),
        "time": dt.datetime.utcfromtimestamp(p["time"]/1000) if p.get("time")
        ↪ else None,
        "mag": p.get("mag"),
        "magType": p.get("magType"),
        "place": p.get("place"),
        "tsunami": p.get("tsunami", 0),
        "geometry": {"type": "Point", "coordinates": coords},
        "properties": p
    })

# Check for missing values (no filtering)
null_counts = {
    "mag": sum(d["mag"] is None for d in docs),
    "time": sum(d["time"] is None for d in docs),

```

```

        "coordinates": sum(
            not d["geometry"]["coordinates"]
            or None in d["geometry"]["coordinates"][:2]
            for d in docs
        )
    }
print("Missing values:", null_counts)

# reset collection and insert
quakes_col.drop()
quakes_col = db[COLL_NAME]
res = quakes_col.insert_many(docs, ordered=False)
len(res.inserted_ids)

```

Missing values: {'mag': 0, 'time': 0, 'coordinates': 0}

1069

The Output shows, that there are no missing values in mag, time and coordinates.

4.4 Transform

In this step, I prepare the earthquake data for analysis. I add a simplified date field (without time) and create a geospatial index for location-based queries. This makes later queries and aggregations faster and easier to handle.

The added field `date` was created in order to simplify time-based analysis. The original `time` field contains timestamps in milliseconds, but for grouping and trend analysis, it is easier to work with daily values. Therefore, the `date` field stores only the day (without hours and minutes).

```

# Create indexes for faster queries
quakes_col.create_index([("geometry", "2dsphere")])
quakes_col.create_index([("time", 1)])

# Add a field "date" (without time)
quakes_col.update_many({}, [
    {"$set": {
        "date": {"$dateTrunc": {"date": "$time", "unit": "day"}}
    }}
])

```

```

UpdateResult({'n': 1069, 'nModified': 1069, 'ok': 1.0, 'updatedExisting':
    ↪ True}, acknowledged=True)

```

```

r = quakes_col.aggregate([
    {"$project": {"_id": 0}},
    {"$limit": 1}
])

```

```

df_preview = pd.DataFrame(list(r))

dfp = df_preview.copy()
if 'geometry' in dfp.columns:
    dfp['geometry'] = dfp['geometry'].astype(str).str.slice(0, 50) + '...'
from pprint import pprint
for _, row in df_preview.iterrows():
    print(f"\n Document:")
    pprint(row.to_dict(), width=80, compact=True)

```

```

 Document:
{'date': Timestamp('2025-10-25 00:00:00'),
 'geometry': {'coordinates': [-82.9981, 8.8628, 10], 'type': 'Point'},
 'mag': 4.6,
 'magType': 'mb',
 'place': '5 km NNW of San Vito, Costa Rica',
 'properties': {'alert': None,
                 'cdi': 4.6,
                 'code': '6000rjhp',
                 'detail': 'https://earthquake.usgs.gov/fdsnws/event/1/query?ev
                 ↪ entid=us6000rjhp&format=geojson',
                 'dmin': 1.323,
                 'felt': 14,
                 'gap': 211,
                 'ids': ',us6000rjhp,',
                 'mag': 4.6,
                 'magType': 'mb',
                 'mmi': None,
                 'net': 'us',
                 'nst': 38,
                 'place': '5 km NNW of San Vito, Costa Rica',
                 'rms': 0.73,
                 'sig': 332,
                 'sources': ',us,',
                 'status': 'reviewed',
                 'time': 1761435913077,
                 'title': 'M 4.6 - 5 km NNW of San Vito, Costa Rica',
                 'tsunami': 0,
                 'type': 'earthquake',
                 'types': ',dyfi,origin,phase-data,',
                 'tz': None,
                 'updated': 1761440127445,
                 'url': 'https://earthquake.usgs.gov/earthquakes/eventpage/us60
                 ↪ 00rjhp'},
 'time': Timestamp('2025-10-25 23:45:13.077000'),
 'tsunami': 0}

```

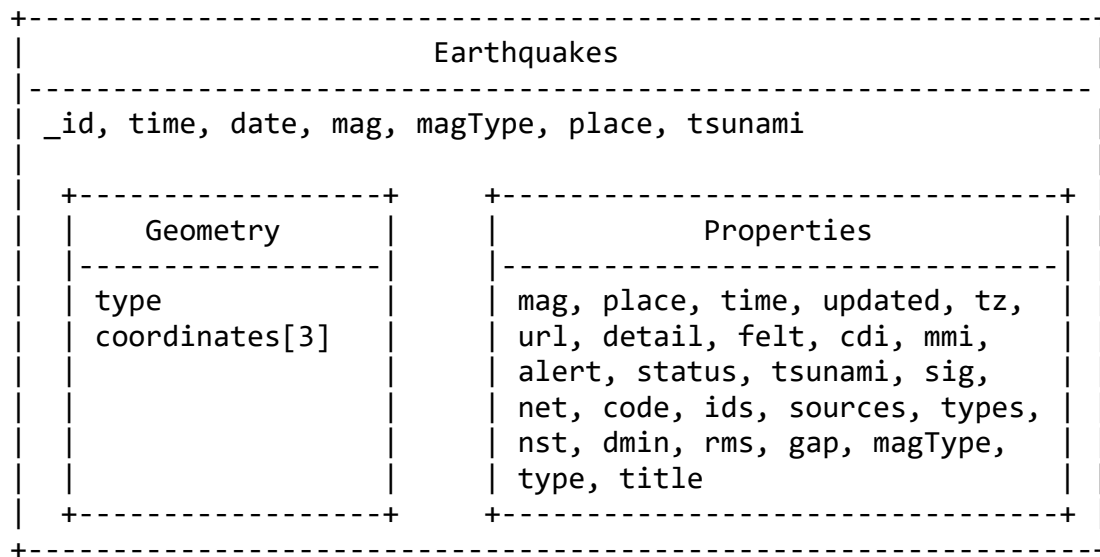
This step checks the stored fields in the database and displays a sample document to verify the data

structure.

5 Datastructure

The dataset is stored in a MongoDB database (earthquakes_db) containing a single collection named earthquakes. Each document represents one earthquake event and includes both simple fields and two embedded objects: geometry and properties. The structure is **denormalized**, meaning that all related data is stored within a single document, including the embedded fields.

5.0.1 Document Structure Diagram



5.0.2 Class Diagram

In order to illustrate the 1:1 relationship between these components more clearly, a Class Diagram is also presented below.



Figure 1: Diagram

Each earthquake document contains exactly one geometry and one properties object, represented here

as 1-to-1 relationships. The open circle in the diagram indicates that these links are technically optional, but in practice they exist for every record in the dataset.

6 Data Analysis

6.0.1 Earthquakes Worldwide in the Past 10 Days

```
# group earthquake by day and calculate Magnitude
r = quakes_col.aggregate([
    {"$group": {"_id": "$date", "count": {"$sum": 1}, "avgMag": {"$avg":
    ↪ "$mag"}}},
    {"$sort": {"_id": 1}}
])

# transform into a dataframe for better visualisation
df_daily = pd.DataFrame(list(r)).rename(columns={"_id": "date"})
df_daily.head(10)
```

	date	count	avgMag
0	2025-09-26	54	4.30
1	2025-09-27	52	4.29
2	2025-09-28	52	4.18
3	2025-09-29	30	4.31
4	2025-09-30	45	4.58
5	2025-10-01	35	4.38
6	2025-10-02	43	4.45
7	2025-10-03	55	4.55
8	2025-10-04	42	4.35
9	2025-10-05	58	4.42

The dataset shows daily earthquake activity over the past 10 days worldwide. The number of earthquakes per day varies between roughly 30 and 60. The average magnitude is typically around 4.2 to 4.5, which indicates mostly moderate events.

Since all earthquakes below a magnitude of 3.0 were removed, the actual total number of earthquakes would be higher and average magnitude of earthquakes during this period would be lower. However, events below 3.0 were excluded because such weak quakes are generally not perceptible to humans.

6.0.2 Earthquake Magnitude Scale

To better understand the relationship between earthquake magnitude and its potential impact, the following diagram presents the magnitude scale and corresponding typical effects.

Magnitude (Mw)	Description	Typical Effects
< 2.0	Micro	Not felt; recorded only by seismographs
2.0 – 2.9	Very Minor	Rarely felt by people
3.0 – 3.9	Minor	Often felt, but rarely causes damage
4.0 – 4.9	Light	Noticeable shaking, minor damage possible
5.0 – 5.9	Moderate	Can cause damage to poorly built structures
6.0 – 6.9	Strong	May cause serious damage in populated areas
7.0 – 7.9	Major	Serious damage over large areas
≥ 8.0	Great / Catastrophic	Massive destruction, severe structural failure

Figure 2: Diagram

Source: Adapted from GeoVera Insurance (2023), “Richter Scale – Earthquake Magnitude and Typical Effects”.

6.0.3 Recent Major Earthquakes

To understand recent seismic activity, let’s look at where the strongest earthquakes (magnitude 6.0 and above) have occurred worldwide during the past 30 days.

```
# List only earthquakes with a magnitude above 6.0
pipeline = [
  {"$match": {"mag": {"$gte": 6.0}}},
  {"$project": {
    "_id": 0,
    "place": 1,
    "mag": 1,
    "date": 1
  }},
  {"$sort": {"mag": -1}}, # sort decreasing
  {"$limit": 10}
]

# Run aggregation
df = pd.DataFrame(list(quakes_col.aggregate(pipeline)))

# Prevent table from being split in Quarto PDF
from IPython.display import HTML, display
display(HTML("<div style='page-break-inside: avoid'>" +
  ↪ df.to_html(index=False) + "</div>"))
```

mag	place	date
7.6	Drake Passage	2025-10-10
7.4	20 km E of Santiago, Philippines	2025-10-10
6.9	12 km E of Bateria, Philippines	2025-09-30
6.7	23 km ESE of Santiago, Philippines	2025-10-10
6.6	26 km WSW of Lae, Papua New Guinea	2025-10-07
6.5	194 km WNW of Abepura, Indonesia	2025-10-16
6.3	Drake Passage	2025-10-16
6.3	134 km SE of Lorengau, Papua New Guinea	2025-10-10
6.1	181 km SE of Vilyuchinsk, Russia	2025-10-03
6.1	0 km WSW of Union, Philippines	2025-10-16

The strongest event (mag 7.6) occurred in the Drake Passage — a deep oceanic region between South America and Antarctica. Other major earthquakes were located around the Philippines, Papua New Guinea, and Indonesia, regions that form part of the Pacific Ring of Fire.

6.0.4 Global Earthquakes in the Past 30 Days

To visualize recent global seismic activity, the following map displays all earthquakes recorded within the last 30 days, highlighting regions with frequent and intense events.

```
# Get coordinates and magnitude fields for mapping
pipe = [{"project":{"_id":0,"mag":1,
                    "lon":{"$arrayElemAt":["$geometry.coordinates",0]},
                    "lat":{"$arrayElemAt":["$geometry.coordinates",1]}}}]

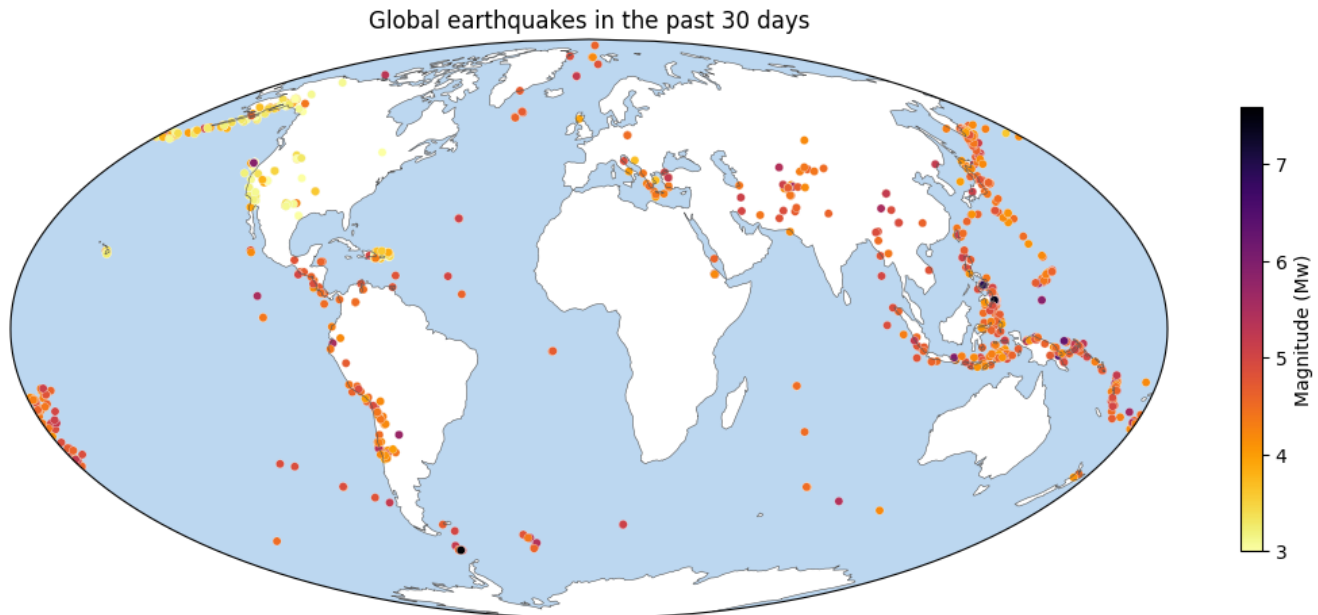
# Load and clean data
df = pd.DataFrame(quakes_col.aggregate(pipe))
df["mag"] = pd.to_numeric(df["mag"], errors="coerce")
df = df.dropna(subset=["mag", "lon", "lat"]).query("-90 <= lat <= 90 and -180 <=
↳ lon <= 180")
fig = plt.figure(figsize=(11,5))
ax = plt.axes(projection=ccrs.Mollweide())
ax.set_global()

# Add ocean color and coastlines
ax.add_feature(cfeature.OCEAN, facecolor="#bcd7f0")
ax.coastlines(linewidth=0.4, color="#555")

# Plot earthquakes (size, color by magnitude)
sc = ax.scatter(df["lon"], df["lat"],
                s=20, c=df["mag"],
                cmap="inferno_r",          # darker = stronger
                transform=ccrs.PlateCarree(),
                edgecolors="white", linewidths=0.2)

# Add colorbar, title, and show
```

```
plt.colorbar(sc, ax=ax, shrink=0.7).set_label("Magnitude (Mw)")
plt.title("Global earthquakes in the past 30 days")
plt.tight_layout()
plt.show()
```



The map shows all recorded earthquakes from the past 30 days. Each dot represents one event, where darker colors mean stronger magnitudes.

Most earthquakes occur along tectonic plate boundaries, especially around the Pacific Ocean (“Ring of Fire”). Earthquakes are particularly dense in regions such as Japan, Indonesia, and the west coast of the Americas, where several major plates meet.

The relative scarcity of events in areas like Africa and northern Europe reflects their position within stable continental interiors. This visualization emphasizes how global seismic activity is closely tied to the movement and interaction of Earth’s tectonic plates.

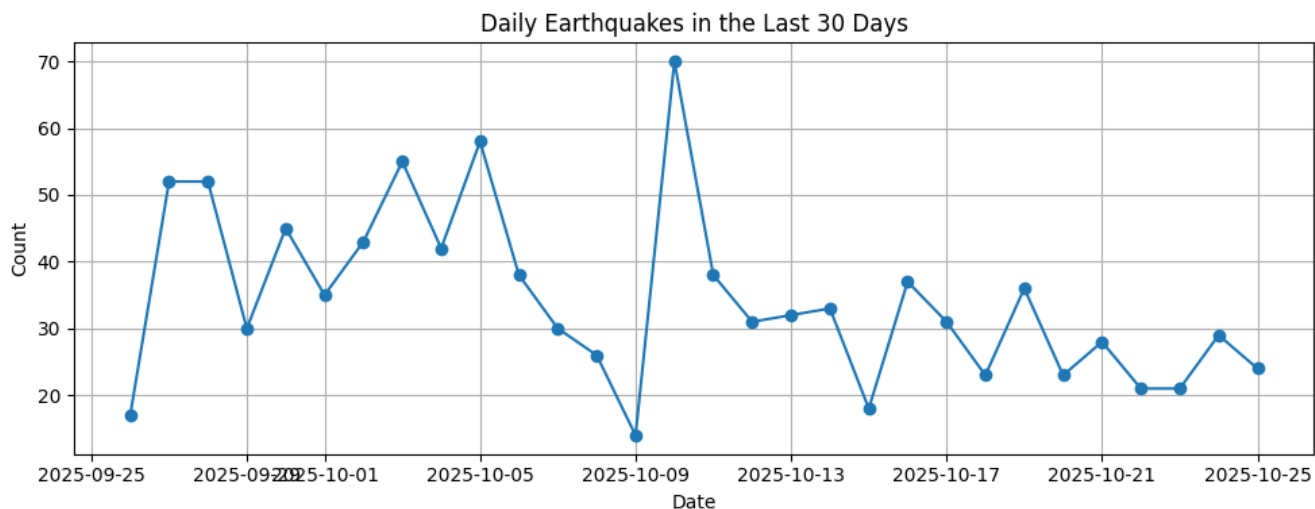
6.0.5 Daily Earthquakes

The following chart illustrates the daily number of recorded earthquakes during the past 30 days, revealing short-term fluctuations in global seismic activity.

```
# define aggregation pipeline
pipe = [
    {"$match": {"time": {"$gte": START.UTC}}},
    {"$group": {"_id": "$date", "count": {"$sum": 1}}},
    {"$sort": {"_id": 1}}
]
# Load and prepare data
df_daily =
    ↪ pd.DataFrame(quakes_col.aggregate(pipe)).rename(columns={"_id": "date"})
```

```
df_daily["date"] = pd.to_datetime(df_daily["date"])

# plot daily earthquake counts
plt.figure(figsize=(10,4))
plt.plot(df_daily["date"], df_daily["count"], marker="o")
plt.title("Daily Earthquakes in the Last 30 Days")
plt.xlabel("Date"); plt.ylabel("Count"); plt.grid(True); plt.tight_layout();
↪ plt.show()
```



The graph shows noticeable fluctuations in daily earthquake activity, with a peak of around 70 events on October 10, 2025. Overall, a slight decreasing trend can be observed toward the end of the observation period.

6.0.6 Spatial Distribution of the 70 Earthquakes Recorded on October 10, 2025

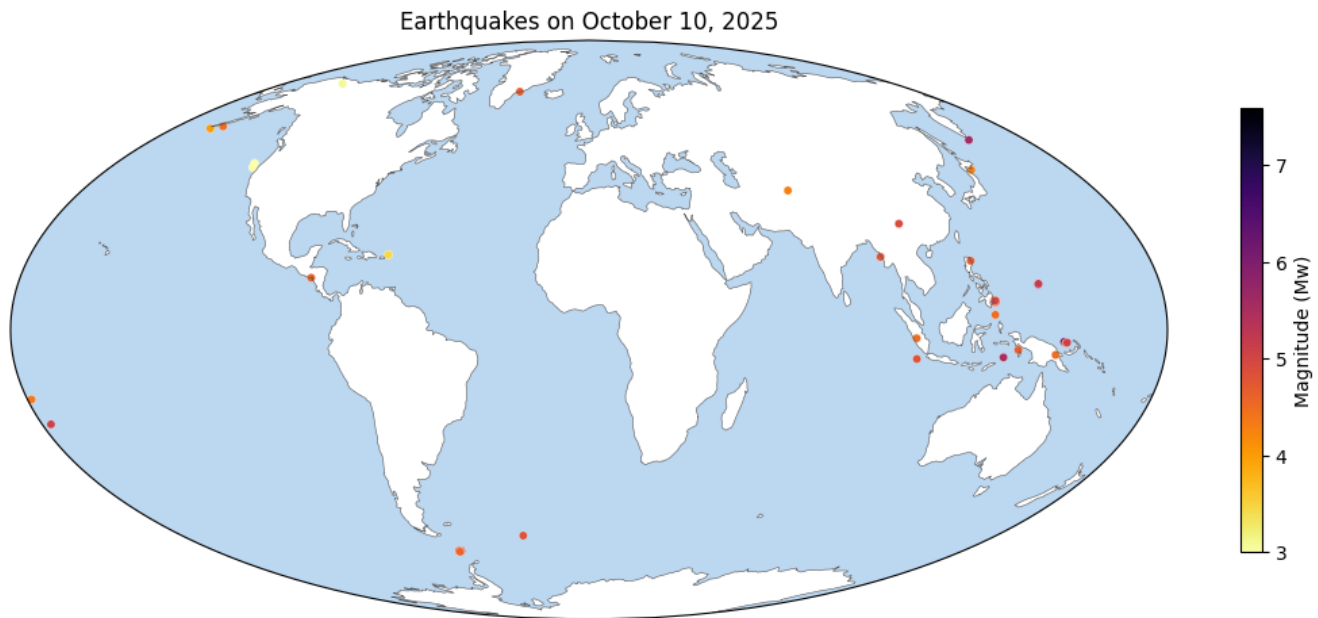
```
# define date range for one day
start_date = datetime(2025,10,10); end_date = start_date + timedelta(days=1)

# build the pipeline
pipe = [
    {"$match": {"time": {"$gte": start_date, "$lt": end_date}}},
    {"$project": {"_id":0, "mag":1,
                  "lon":{"$arrayElemAt":["$geometry.coordinates",0]},
                  "lat":{"$arrayElemAt":["$geometry.coordinates",1]}}}
]
# Load an clean data
df = pd.DataFrame(quakes_col.aggregate(pipe))
df["mag"] = pd.to_numeric(df["mag"], errors="coerce")
df = df.dropna(subset=["mag","lon","lat"]).query("-90 <= lat <= 90 and -180 <=
↪ lon <= 180")
```

```
# create global map
fig = plt.figure(figsize=(11,5))
ax = plt.axes(projection=ccrs.Mollweide())
ax.set_global()
ax.add_feature(cfeature.OCEAN, facecolor="#bcd7f0")
ax.coastlines(linewidth=0.4, color="#555")

# plot earthquakes
sc = ax.scatter(df["lon"], df["lat"],
               s=20, c=df["mag"], cmap="inferno_r",
               transform=ccrs.PlateCarree(),
               edgecolors="white", linewidths=0.2)

plt.colorbar(sc, ax=ax, shrink=0.7).set_label("Magnitude (Mw)")
plt.title("Earthquakes on October 10, 2025")
plt.tight_layout()
plt.show()
```



The spatial pattern indicates that the majority of earthquakes on October 10, 2025, occurred near the Philippines and Indonesia. This region is part of the Pacific “Ring of Fire,” where the interaction of several tectonic plates causes frequent and sometimes powerful seismic events.

6.0.7 How many of the earthquakes on October 10, 2025, occurred around the Philippines?

Based on the previous global map, most earthquakes on October 10, 2025, appeared to cluster around the Philippines. To verify this observation quantitatively, the following query filters all events within a 1000 km radius of the Philippine region using both spatial and time indexes in MongoDB.

```

# Philippines
center = {"type": "Point", "coordinates": [122.0, 12.0]} # [lon, lat]

# Date range for October 10, 2025
start_date = datetime(2025, 10, 10)
end_date = start_date + timedelta(days=1)

# Query for quakes on that date around the Philippines
query = {
    "time": {"$gte": start_date, "$lt": end_date}, # uses time index
    "geometry": { # uses 2dsphere index
        "$near": {
            "$geometry": center,
            "$maxDistance": 1000000 # 1000 km radius
        }
    }
}

# print the results
results = list(quakes_col.find(query, {"place": 1, "mag": 1, "_id": 0})) # <-
    ↪ limit(70) entfernen!
print(f"\nEarthquakes around Philippines on {start_date.date()}:
    ↪ {len(results)} found")

```

Earthquakes around Philippines on 2025-10-10: 35 found

The result shows that 35 earthquakes occurred near the Philippines on that day, confirming that the majority of seismic activity was concentrated in this area.

6.0.8 Where was the main cluster of earthquakes on October 10, 2025?

To identify whether the earthquakes on October 10, 2025, were concentrated in a specific region on the Philippines, the following aggregation groups all events by rounded latitude and longitude. This approach highlights spatial clusters of seismic activity (not a single epicenter).

```

# define date range for analysis
s,e = datetime(2025,10,10), datetime(2025,10,11)

# aggregate earthquakes by rounded coordinates
pipe = [
    {"$match":{"time":{"$gte":s,"$lt":e}}},
    {"$project":{"
        "cellLat":{"$round":[{"$arrayElemAt":["$geometry.coordinates",1]},1]},
        "cellLon":{"$round":[{"$arrayElemAt":["$geometry.coordinates",0]},1]}
    }},
    {"$group":{"_id":{"lat":"$cellLat","lon":"$cellLon"}, "count":{"$sum":1}}},
    {"$sort":{"count":-1}}, {"$limit":10}
]

```

```
# display the most active regions (potential seismic clusters)
list(quakes_col.aggregate(pipe))
```

```
[{'_id': {'lat': 7.1, 'lon': 126.8}, 'count': 4},
 {'_id': {'lat': 7.2, 'lon': 127.0}, 'count': 3},
 {'_id': {'lat': 7.1, 'lon': 126.9}, 'count': 2},
 {'_id': {'lat': 7.3, 'lon': 126.8}, 'count': 2},
 {'_id': {'lat': 7.2, 'lon': 127.3}, 'count': 2},
 {'_id': {'lat': 7.2, 'lon': 126.8}, 'count': 2},
 {'_id': {'lat': 7.0, 'lon': 126.8}, 'count': 2},
 {'_id': {'lat': 7.4, 'lon': 127.1}, 'count': 2},
 {'_id': {'lat': -60.1, 'lon': -62.5}, 'count': 2},
 {'_id': {'lat': 53.9, 'lon': -164.0}, 'count': 1}]
```

The aggregation shows a clear spatial concentration of earthquakes on October 10, 2025. Most events occurred around 7° N, 127° E, indicating an active cluster near the Philippine Trench, east of Mindanao. Several nearby cells (7.1–7.4° N, 126.8–127.3° E) also recorded multiple quakes, supporting the presence of a localized seismic hotspot. Only a few isolated events appeared elsewhere.

6.0.9 Tsunami-Flagged Earthquakes in the Past 30 Days

To assess the tsunami risk among recent seismic events, the query counts how many earthquakes were flagged as tsunami-related during the past 30 days, and specifically on October 10, 2025 — the day with the highest earthquake activity.

```
# define time windows
s = datetime(2025, 10, 10)
e = s + timedelta(days=1)

# aggregation: count tsunami-flagged events
pipe = [
    {"$match": {"tsunami": {"$gt": 0}}},          # include only tsunami-flagged
    ↪ events
    {"$facet": {                                  # parallel sub-queries
        "last30": [
            {"$match": {"time": {"$gte": START.UTC}}},
            {"$count": "n"}
        ],
        "oct10": [
            {"$match": {"time": {"$gte": s, "$lt": e}}},
            {"$count": "n"}
        ]
    }}
]

# execute and extract results
res = list(quakes_col.aggregate(pipe))[0]
n30  = res["last30"][0]["n"] if res["last30"] else 0
n1010 = res["oct10"][0]["n"] if res["oct10"] else 0
```



```
# print summary
print(f"Tsunami-flagged in last 30 days: {n30} | on {s.date()}: {n1010}")
```

Tsunami-flagged in last 30 days: 10 | on 2025-10-10: 3

The result shows that 10 tsunami-flagged earthquakes occurred globally within the last 30 days, 3 of which happened on October 10, 2025.

This suggests that several of the strong earthquakes on that day may have been powerful enough to trigger tsunami warnings, indicating elevated seismic–oceanic interaction in the region.

7 Conclusions

This project demonstrated the complete workflow of building a data pipeline with Python and MongoDB — from data extraction to visualization and insight generation. Using the USGS Earthquake API, I collected global earthquake data from the past 30 days and stored it in a NoSQL database for further analysis. During the ELT process, I prepared and indexed the data to support efficient time-based and geospatial queries, enabling high-performance exploration of temporal and spatial patterns.

The analytical phase included several MongoDB aggregations that revealed key patterns in recent seismic activity. Grouping earthquakes by day showed strong fluctuations in the number of events, with a noticeable peak around October 10, 2025. When filtering by magnitude, the data highlighted that most earthquakes fall within a moderate range (Mw 4–5), while only a few reach magnitudes above 6.0. Mapping the global distribution made clear that most seismic activity is concentrated along tectonic plate boundaries, especially around the Pacific Ring of Fire — including regions such as the Philippines, Indonesia, and Chile.

A focused analysis on October 10 revealed that over 70 earthquakes occurred on that single day, about half of them near the Philippines, suggesting a regional cluster or local swarm activity. Further investigation of tsunami-flagged events showed that 10 earthquakes in the last 30 days were identified as potentially tsunami-inducing, including three events on October 10. This supports the interpretation of elevated oceanic–seismic coupling in that region.

Overall, this project demonstrates how combining MongoDB’s aggregation framework with Python’s analytical ecosystem can effectively uncover both global and temporal patterns in real-world datasets. It also highlights the importance of data transformation, indexing, and visualization in turning raw event data into meaningful geophysical insights, capable of supporting both scientific interpretation and practical hazard awareness.

8 Learnings

While working on this project, I learned how to build a complete data pipeline — from collecting live data via the USGS API to cleaning, transforming, storing, and analyzing it in MongoDB. I became much more confident in using MongoDB’s aggregation framework and understood how powerful it is for handling nested data and filtering with complex conditions.

At first, the aggregation code seemed very abstract, but after trying many queries and debugging them with the help of ChatGPT, I finally understood how each stage works and how they can be combined to

filter, group, and count data efficiently. I spent a lot of time experimenting, fixing errors, and understanding how all the parts of the workflow fit together — from data collection to the final report.

One of the biggest challenges was getting the results to display correctly in Quarto. I struggled with formatting issues for hours — especially with long code outputs and wide tables — but through this process, I learned how to control layout, wrapping, and font settings much better. I now understand how Quarto handles code cells and Markdown much more clearly.

I also learned how to visualize data effectively with Matplotlib and Cartopy, and how different map projections and color scales can change the interpretation of results.

Overall, this project taught me not just technical skills, but also patience, precision, and persistence — and how valuable it is to truly understand the cause of an error instead of just trying to fix it quickly.