```
!nvidia-smi
```

```
Thu Sep 12 21:10:09 2024
+---------------------------------------------------------------------------------------+
| NVIDIA-SMI 535.104.05             Driver Version: 535.104.05   CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M. |
|=========================================+======================+======================|
|   0  Tesla T4                       Off | 00000000:00:04.0 Off |                    0 |
| N/A   35C    P8              9W /  70W |      0MiB / 15360MiB |      0%      Default |
|                                         |                      |                  N/A |
+-----------------------------------------+----------------------+----------------------+

+---------------------------------------------------------------------------------------+
| Processes:                                                                            |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory |
|        ID   ID                                                             Usage      |
|=======================================================================================|
|  No running processes found                                                           |
+---------------------------------------------------------------------------------------+
```

```
import numpy as np
import pandas as pd
import folium
import seaborn as sns
from folium import Choropleth, Circle, Marker
from folium.plugins import HeatMap, MarkerCluster
from folium.plugins import FastMarkerCluster
import math
import html
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings("ignore")

from IPython.display import Image


from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Mineral ores round the world.csv',low_memory=False)
pd.set_option('display.max_columns', len(df.columns))
df.head()
```

	site_name	latitude	longitude	region	country	state	county	com_type	commod1	commod2	commod3	oper_type	dep_type	prod_si
0	Lookout Prospect	55.05612	-132.14344	NaN	United States	Alaska	NaN	M	Copper	Gold, Silver	NaN	Unknown	NaN	
1	Lucky Find Prospect	55.52751	-132.68514	NaN	United States	Alaska	NaN	M	Copper	Gold	NaN	Unknown	NaN	
2	Mccullough Prospect	55.97751	-132.99906	NaN	United States	Alaska	NaN	M	Copper	NaN	Zinc, Gold	Unknown	NaN	
3	Lucky Jim Claim	55.52195	-132.68653	NaN	United States	Alaska	NaN	M	Gold	NaN	Copper, Lead	Unknown	NaN	
4	Matilda Occurrence	55.14556	-132.05233	NaN	United States	Alaska	NaN	M	Gold	NaN	NaN	Unknown	NaN	

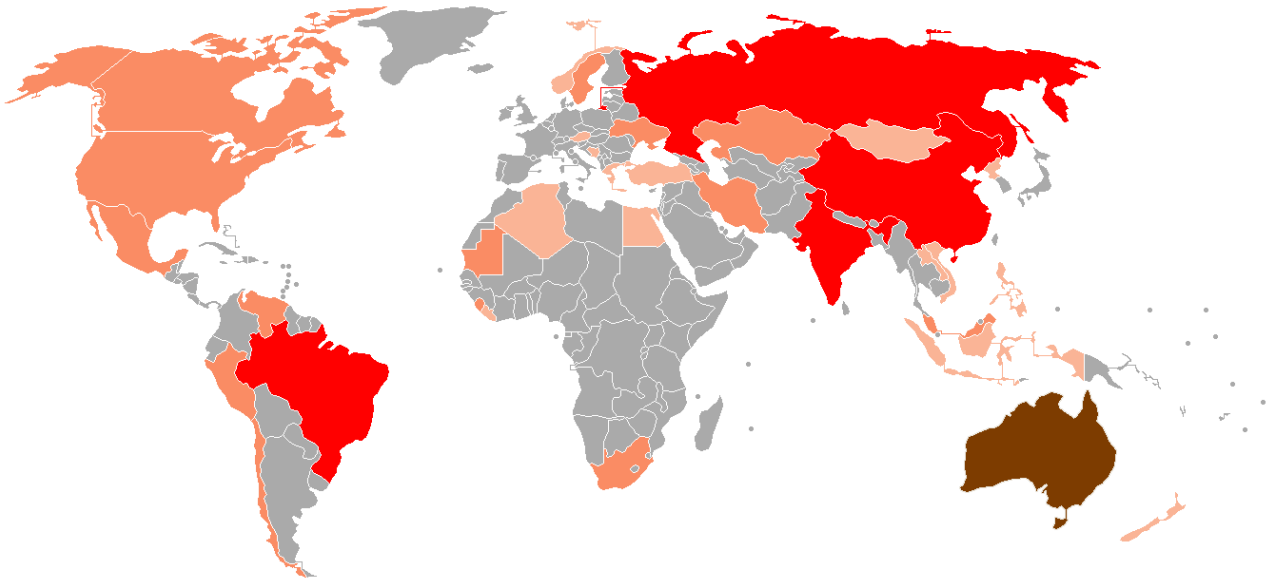
MINERALS MINES

Start coding or generate with AI.

```
percent_missing = df.isnull().sum() * 100 / len(df)
missing_value_df = pd.DataFrame({'column_name': df.columns,
                                 'percent_missing': percent_missing})
missing_value_df.sort_values('percent_missing', inplace=True)
round(missing_value_df,2)
```

	column_name	percent_missing
dev_stat	dev_stat	0.00
oper_type	oper_type	0.00
site_name	site_name	0.00
country	country	0.00
longitude	longitude	0.01
latitude	latitude	0.01
com_type	com_type	1.25
state	state	4.10
commod1	commod1	6.90
county	county	16.99
prod_size	prod_size	64.43
ore	ore	76.84
hrock_type	hrock_type	77.18
names	names	78.55
commod3	commod3	79.27
dep_type	dep_type	87.70
gangue	gangue	89.16
region	region	89.77
arock_type	arock_type	91.58
work_type	work_type	91.60
ore_ctrl	ore_ctrl	92.15
commod2	commod2	92.90

Explore Countries' Mineral Ore



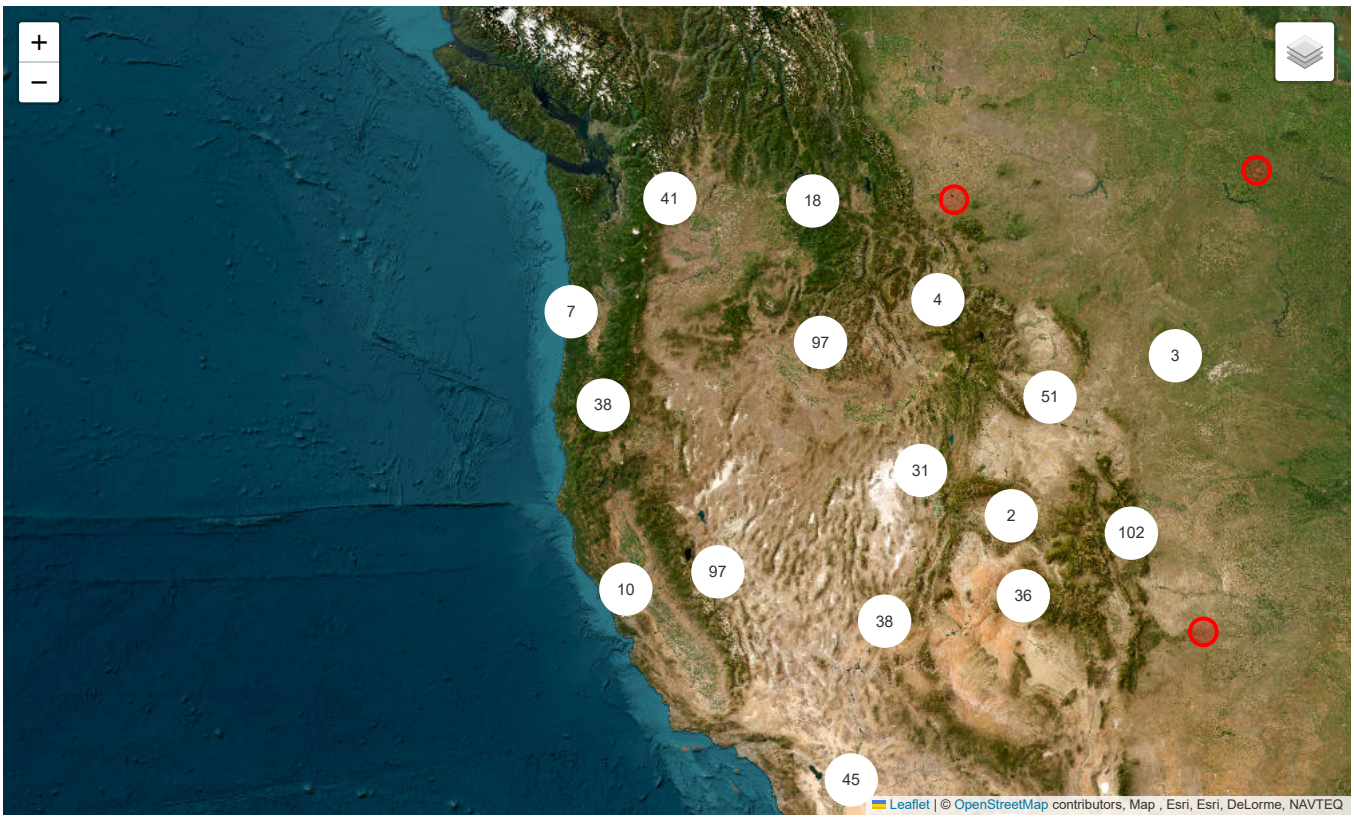
```
country_group = df.groupby(['country','commod1','latitude','longitude','site_name','state'])['country'].count().reset_index(name='count
country_group.sort_values(by='count',ascending=False,inplace=True,ignore_index=True)
country_group = country_group[country_group["count"] > 1]
country_group
```

	country	commod1	latitude	longitude	site_name	state	count
0	United States	Chromium	42.36105	-123.55343	Name Unknown	Oregon	18
1	United States	Copper, Gold, Silver	33.79896	-113.72267	Unknown	Arizona	13
2	United States	Lead, Silver	32.93088	-114.16658	Unknown	Arizona	8
3	United States	Copper, Gold, Silver	34.12925	-114.12749	Unknown	Arizona	8
4	United States	Gold, Silver	32.93088	-114.16658	Unknown	Arizona	7
...	...	...	...	...	...	...	...
1037	United States	Uranium	38.64189	-108.97238	Unnamed Mine	Colorado	2
1038	United States	Sand and Gravel, Construction	34.97791	-82.54346	Hendrix Pit	South Carolina	2
1039	United States	Gold	33.86681	-86.33887	Hatchet Creek Placer	Alabama	2
1040	United States	Gold	44.02408	-115.80876	Mohawk Mine	Idaho	2
1041	United States	Gold	44.02098	-115.83816	Bruser Mine	Idaho	2

1042 rows × 7 columns

```
# Create the map with initial parameters
folium_map = folium.Map(location=[29.2985, 42.5510],
                        zoom_start=2,
                        tiles='OpenStreetMap')

# Add more tile layers
folium.TileLayer('Stamen Terrain' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB positron' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB dark_matter' , attr="Map ").add_to(folium_map)

# Add the Esri Satellite tiles with attribution
esri_satellite = folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
    attr='Esri, Esri, DeLorme, NAVTEQ',
    name='Esri Satellite',
    overlay=False,
    control=True
)
esri_satellite.add_to(folium_map)

# Create a MarkerCluster
mc = MarkerCluster(name="Marker Cluster")

# Add markers to the map
for index, row in country_group.iterrows():
    popup_text = """
```

```
                <h1> Information Blog </h1><br>
                <li>Index Number: {} </li>
                <li>Site Name   : {} </li>
                <li>Country     : {} </li>
                <li>Commod1     : {} </li>
                <li>State       : {} </li>
            """.format(index,
                       row["site_name"],
                       row["country"],
                       row["commod1"],
                       row["state"])
        folium.CircleMarker(location=[row["latitude"], row["longitude"]],
                            color="red", popup=popup_text,
                            fill=True).add_to(mc)

# Add the marker cluster to the map
mc.add_to(folium_map)

# Add layer control to toggle between layers
folium.LayerControl().add_to(folium_map)

# Display the map
folium_map
```



Sand and Gravel, Construction

Description

Construction sand and gravel, one of the most accessible natural resources and a major basic raw material, is used mostly by the construction industry. Despite the low unit value of its basic products, the construction sand and gravel industry is a major contributor to and an indicator of the economic well-being of the Nation.

Sources

Construction sand and gravel are produced in all 50 states in the U.S. The states producing the most are California, Texas, Michigan, Ohio, Arizona, Colorado, Minnesota, Washington, and Utah. Together, they produce 52% of the total amount of construction sand and gravel mined and processed in the United States.

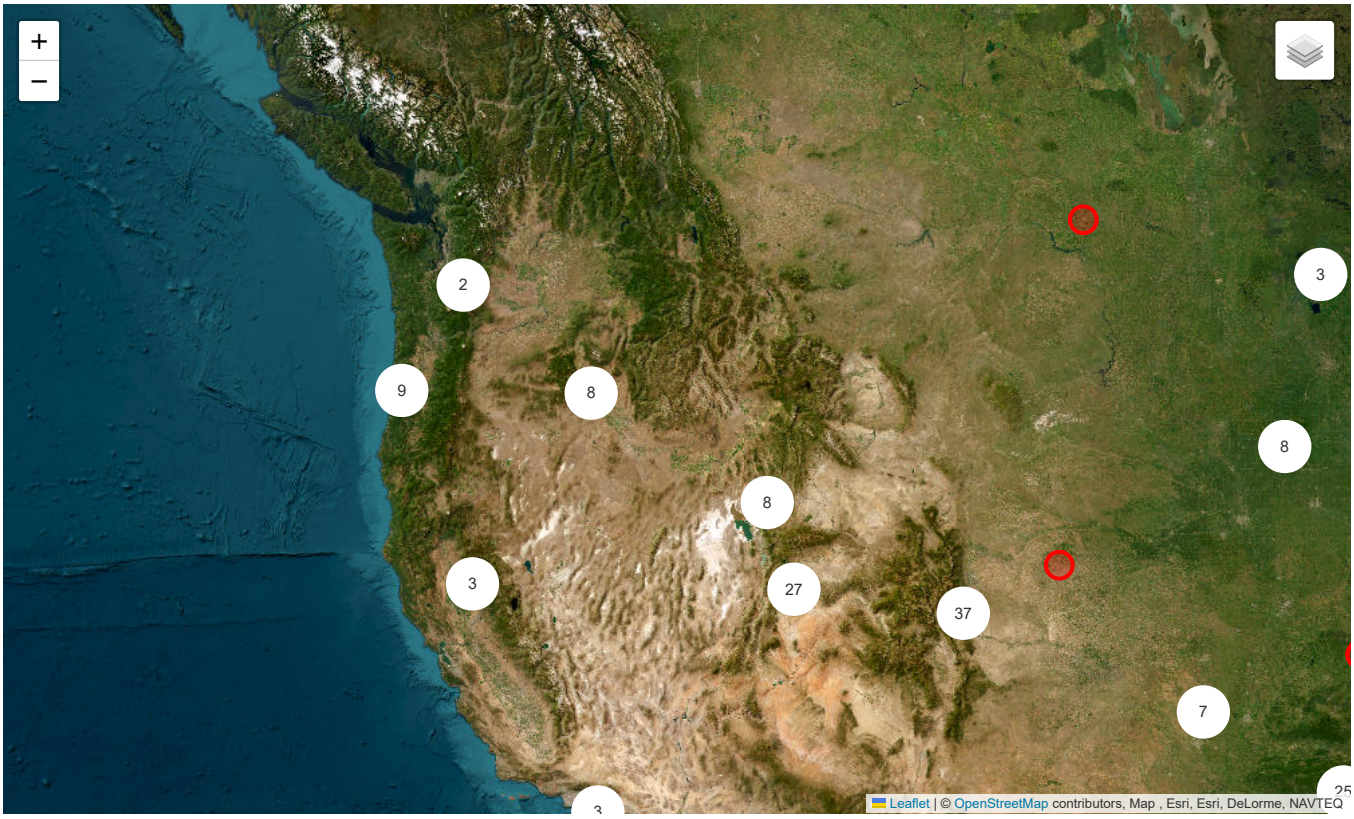
```
sand_gravel = df[df["commod1"] == "Sand and Gravel, Construction"]
sand_gravel = sand_gravel.groupby(['country','commod1','latitude','longitude','site_name','state'])['country'].count().reset_index(name=
sand_gravel.sort_values(by='count',ascending=False,inplace=True,ignore_index=True)
sand_gravel = sand_gravel[sand_gravel['count'] > 1]
sand_gravel
```

	country	commod1	latitude	longitude	site_name	state	count
0	United States	Sand and Gravel, Construction	37.56295	-105.78827	Unknown	Colorado	4
1	United States	Sand and Gravel, Construction	40.06639	-102.10548	Unknown	Colorado	4
2	United States	Sand and Gravel, Construction	39.57471	-105.04224	Unknown	Colorado	3
3	United States	Sand and Gravel, Construction	42.72440	-94.21604	Unknown	Iowa	3
4	United States	Sand and Gravel, Construction	33.94153	-92.47096	Unk	Arkansas	3
...	...	...	...	...	...	...	...
241	United States	Sand and Gravel, Construction	42.58188	-109.97288	Unknown Sand and Gravel pit	Wyoming	2
242	United States	Sand and Gravel, Construction	36.49259	-88.43282	Dixon Pit	Tennessee	2
243	United States	Sand and Gravel, Construction	33.89934	-92.15435	Unk	Arkansas	2
244	United States	Sand and Gravel, Construction	34.20153	-92.06455	Unk	Arkansas	2
245	United States	Sand and Gravel, Construction	36.31731	-89.25729	Cole Pit	Tennessee	2

246 rows × 7 columns

```
folium_map = folium.Map(location=[37.0902, -95.7129],
                        zoom_start=2,
                        tiles='OpenStreetMap')
# Add more tile layers
folium.TileLayer('Stamen Terrain' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB positron' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB dark_matter', attr="Map ").add_to(folium_map)

# Add the Esri Satellite tiles with attribution
esri_satellite = folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
```

```
        attr='Esri, Esri, DeLorme, NAVTEQ',
        name='Esri Satellite',
        overlay=False,
        control=True
    )
    esri_satellite.add_to(folium_map)

    mc = MarkerCluster(name="Marker Cluster")

    for index, row in sand_gravel.iterrows():
        popup_text="""
                    <h1> Information Blog </h1><br>
                    <li">Index Number: {} </li>
                    <li>Site Name   : {:} </li>
                    <li>Country     : {:} </li>
                    <li>Commod1     : {:} </li>
                    <li>State       : {:} </li>
            """.format(index,
                            row["site_name"],
                            row["country"],
                            row["commod1"],
                            row["state"])
        folium.CircleMarker(location=[row["latitude"],row["longitude"]],
                            color="red",popup=popup_text,
                            fill=True).add_to(mc)
    folium.Popup(html=popup_text,sticky=True,max_width='250%',show=False).add_to(mc)
    mc.add_to(folium_map)
    folium.LayerControl().add_to(folium_map)
    folium_map
```



Stone,Crushed/Broken

Description

Crushed stone is a popular material among construction professionals. It is used for plenty of different construction projects, and it is favoured for its versatility and accessibility. As the name suggests, crushed stone is a construction material made from rocks that were broken into smaller pieces. It is most often used as an aggregate in construction projects. The stones in their original size would not work for these projects, but upon being crushed, they are perfect for mixing or packing.

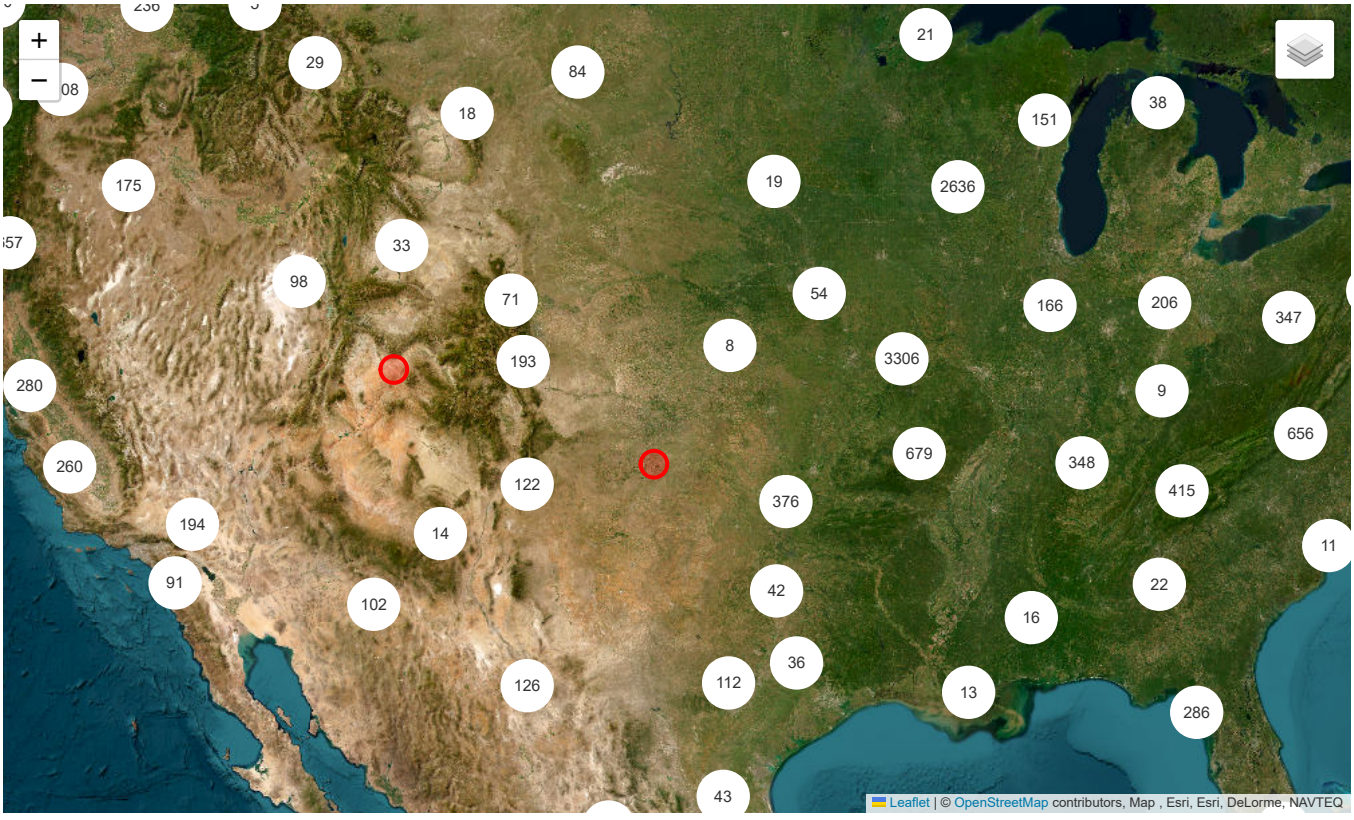
```
stone_crushed = df[df["commod1"] == "Stone, Crushed/Broken"]
stone_crushed = stone_crushed.groupby(['country','commod1','latitude','longitude','site_name','state'])['country'].count().reset_index(n
stone_crushed.sort_values(by='count',ascending=False,inplace=True,ignore_index=True)
stone_crushed
```

	country	commod1	latitude	longitude	site_name	state	count
0	United States	Stone, Crushed/Broken	41.70520	-112.85667	Monument Point West	Utah	2
1	United States	Stone, Crushed/Broken	42.49123	-122.82144	Unnamed Pit	Oregon	2
2	United States	Stone, Crushed/Broken	38.28945	-92.27236	Unknown-Ls	Missouri	2
3	Mexico	Stone, Crushed/Broken	29.65994	-104.58106	Unidentified Quarry	Chihuahua	2
4	United States	Stone, Crushed/Broken	37.14787	-92.74578	Unknown - Ls	Missouri	2
...	...	...	...	...	...	...	...
16398	United States	Stone, Crushed/Broken	38.53255	-90.56511	Unknown - Ls	Missouri	1
16399	United States	Stone, Crushed/Broken	38.53315	-92.10076	Unknown-Ls	Missouri	1
16400	United States	Stone, Crushed/Broken	38.53315	-91.20823	Plant # 1	Missouri	1
16401	United States	Stone, Crushed/Broken	38.53330	-106.00058	C J List	Colorado	1
16402	United States	Stone, Crushed/Broken	38.52895	-90.56821	Unknown - Ls	Missouri	1

16403 rows × 7 columns

```
folium_map = folium.Map(location=[37.0902, -95.7129],
                        zoom_start=2,
                        tiles='OpenStreetMap')
# Add more tile layers
folium.TileLayer('Stamen Terrain' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB positron' , attr="Map ").add_to(folium_map)
```

```
folium.TileLayer('CartoDB dark_matter', attr='Map ').add_to(folium_map)

# Add the Esri Satellite tiles with attribution
esri_satellite = folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
    attr='Esri, Esri, DeLorme, NAVTEQ',
    name='Esri Satellite',
    overlay=False,
    control=True
)
esri_satellite.add_to(folium_map)
mc = MarkerCluster(name="Marker Cluster")

for index, row in stone_crushed.iterrows():
    popup_text="""
                <h1> Information Blog </h1><br>
                <li">Index Number: {} </li>
                <li>Site Name   : {:} </li>
                <li>Country     : {:} </li>
                <li>Commod1     : {:} </li>
                <li>State       : {:} </li>
        """.format(index,
                        row["site_name"],
                        row["country"],
                        row["commod1"],
                        row["state"])
    folium.CircleMarker(location=[row["latitude"],row["longitude"]],
                        color="red",popup=popup_text,
                        fill=True).add_to(mc)
folium.Popup(html=popup_text,sticky=True,max_width='250%',show=False).add_to(mc)
mc.add_to(folium_map)
folium.LayerControl().add_to(folium_map)
folium_map
```



COPPER

Description

Copper is a naturally occurring element. Present in the earth's crust, in oceans, lakes and rivers, sources of copper range from minute trace amounts to rich mine deposits. Copper is an essential element, meaning that all plants, fish and animals need copper to function properly.

Copper is naturally present in the Earth's crust, at a concentration of about 67 parts per million. While most mines operate with copper concentrations of between 0.2 and 0.8 %, some of the richest ore bodies, located in central to southern Africa, can contain 5–6% copper.

Sources

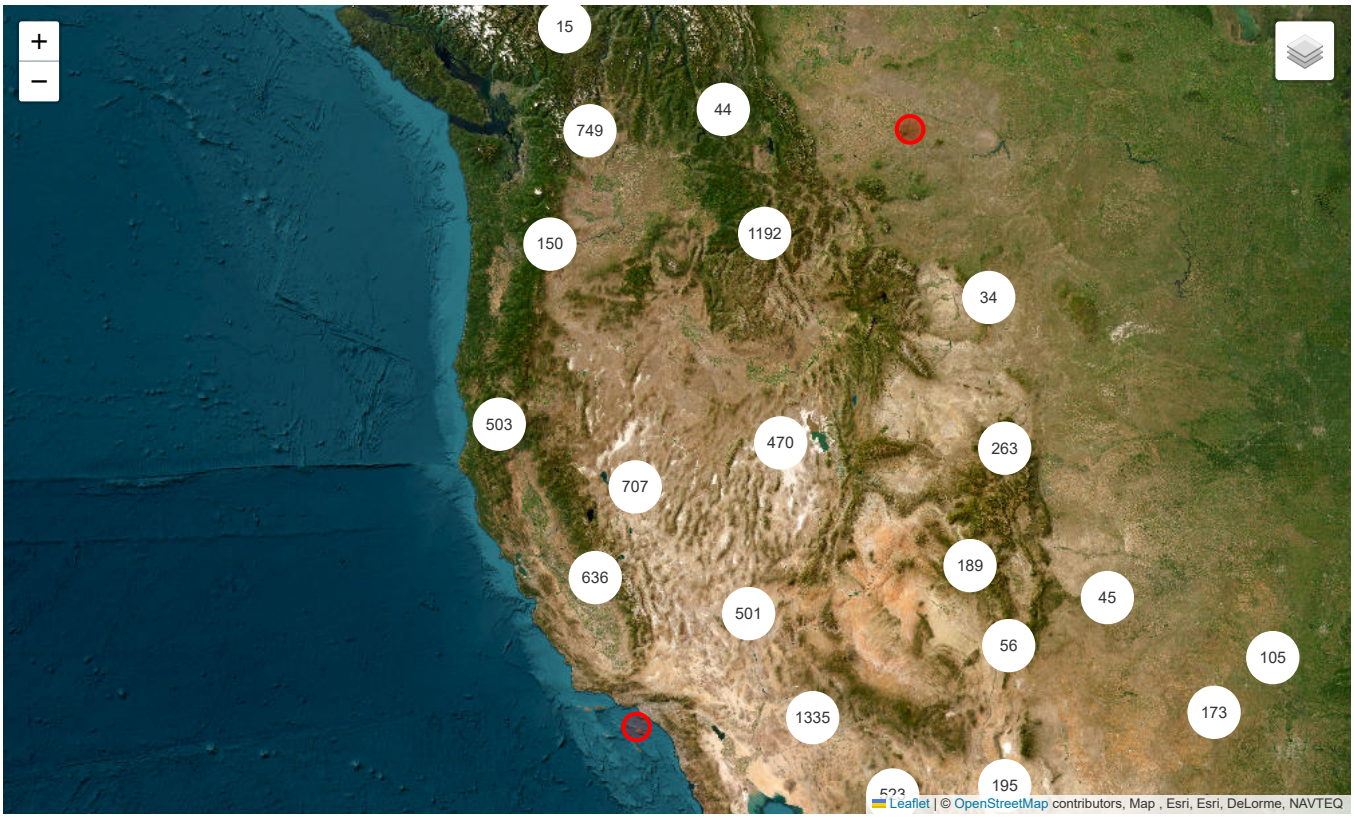
Within Europe, the largest known deposits are in Russia and Poland. Economic mines are also operating in Bulgaria, Finland, Portugal, Spain and Sweden. In 2010, EU mine production totalled 855,316 tonnes of copper (source: BGS), around 5.3% of world production.

```
copper_country = df[df["commod1"] == "Copper"]
copper_country = copper_country.groupby(['country','commod1','latitude','longitude','site_name','state'])['country'].count().reset_index
copper_country.sort_values(by='count',ascending=False,inplace=True,ignore_index=True)
copper_country
```

	country	commod1	latitude	longitude	site_name	state	count
0	United States	Copper	33.75696	-114.02108	Unknown	Arizona	3
1	United States	Copper	37.71414	-113.47414	Prospects East of Bullion Canyon Adit	Utah	2
2	United States	Copper	37.12081	-113.92803	Middle Spring Prospect	Utah	2
3	United States	Copper	45.43179	-113.99672	Copper Queen	Idaho	2
4	United States	Copper	37.11831	-113.95164	Reber Wash Prospects	Utah	2
...	...	...	...	...	...	...	...
12390	United States	Copper	35.37669	-117.89170	Copper Queen Claims	California	1
12391	United States	Copper	35.37691	-115.48164	Alta Crown	California	1
12392	United States	Copper	35.37721	-115.53834	Express	California	1
12393	United States	Copper	35.37721	-115.47724	Unnamed Prospect	California	1
12394	United States	Copper	35.37171	-115.47774	Allured Copper Mine	California	1

12395 rows × 7 columns

```
# Create the map with initial parameters
folium_map = folium.Map(location=[29.2985, 42.5510], zoom_start=2, tiles='OpenStreetMap')

# Add more tile layers
folium.TileLayer('Stamen Terrain' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB positron' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB dark_matter' , attr="Map ").add_to(folium_map)

# Add the Esri Satellite tiles with attribution
esri_satellite = folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
    attr='Esri, Esri, DeLorme, NAVTEQ',
    name='Esri Satellite',
    overlay=False,
    control=True
)
esri_satellite.add_to(folium_map)

mc = MarkerCluster(name="Marker Cluster")

for index, row in copper_country.iterrows():
    popup_text="""
                <h1> Information Blog </h1><br>
                <li">Index Number: {} </li>
                <li>Site Name   : {:} </li>
                <li>Country     : {:} </li>
                <li>Commod1     : {:} </li>
                <li>State       : {:} </li>
        """.format(index,
                           row["site_name"],
                           row["country"],
                           row["commod1"],
                           row["state"])
    folium.CircleMarker(location=[row["latitude"],row["longitude"]],
                        color="red",popup=popup_text,
                        fill=True).add_to(mc)
folium.Popup(html=popup_text,sticky=True,max_width='250%',show=False).add_to(mc)
mc.add_to(folium_map)
folium.LayerControl().add_to(folium_map)
folium_map
```



IRON

Description

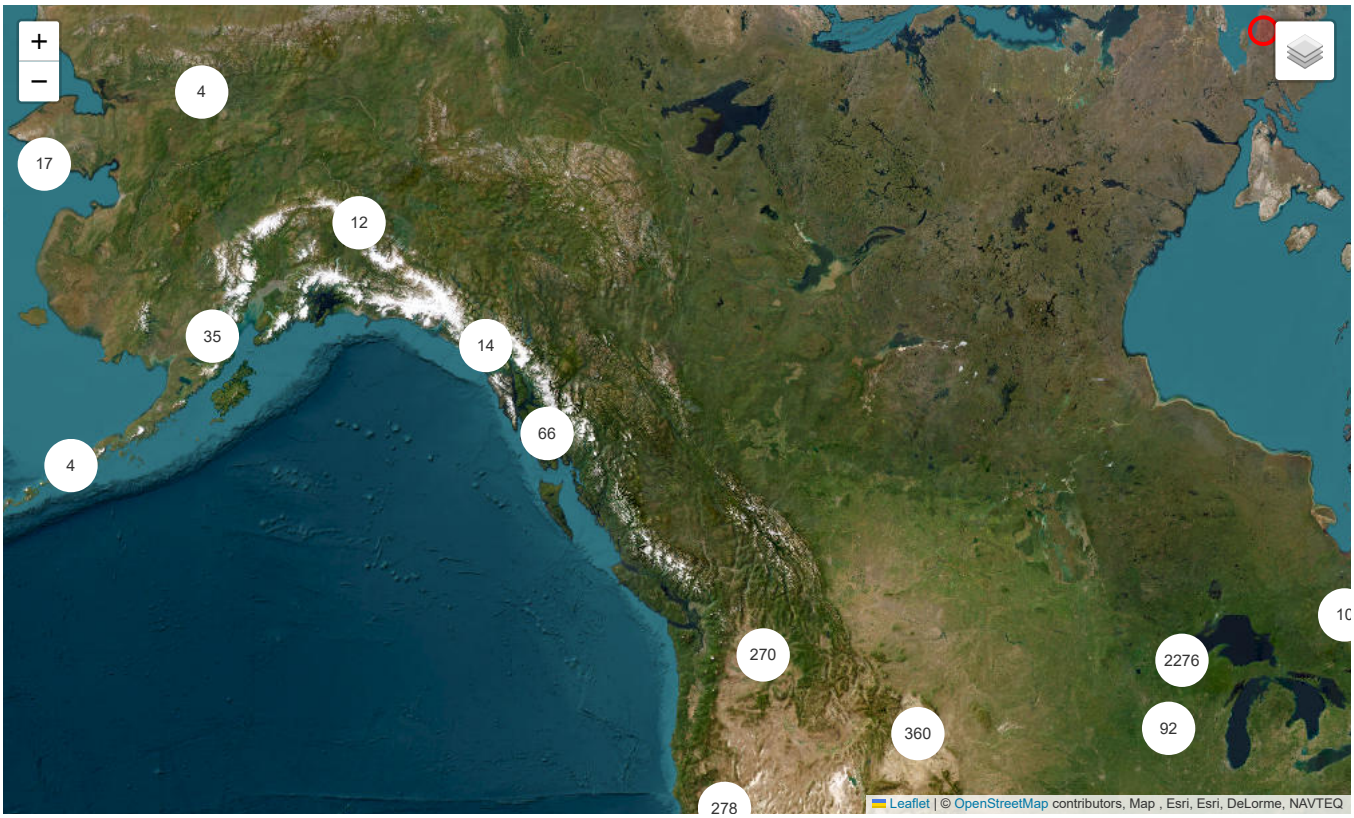
Iron is the backbone of the world we have built around us and it is the basic ingredient of steel (iron plus carbon). Iron is a very useful metal because it can be mixed with other metals to make a whole variety of 'alloys' which are even stronger and don't rust easily and can be shaped into products from cars to pins, household appliances to buildings, bridges to railways, food cans to tools. In short, we rely on iron (as steel) to make almost everything we need for living in the 21st century. Today we use twenty times more iron (in the form of steel) than all other metals put together.

```
iron_country = df[df["commod1"] == "Iron"]
iron_country = iron_country.groupby(['country','commod1','latitude','longitude','site_name','state'])['country'].count().reset_index(nar
iron_country.sort_values(by='count',ascending=False,inplace=True,ignore_index=True)
iron_country
```

	country	commod1	latitude	longitude	site_name	state	count
0	United States	Iron	33.97514	-86.40415	Taits Gap Mines	Alabama	2
1	United States	Iron	40.89306	-74.58959	Stirling Mine	New Jersey	2
2	United States	Iron	40.26945	-76.40298	Cornwall Mine	Pennsylvania	2
3	United States	Iron	41.52749	-73.88762	Mase's Limonite Mine	New York	2
4	United States	Iron	34.57482	-89.32897	Unnamed Prospect	Mississippi	2
...	...	...	...	...	...	...	...
12957	United States	Iron	36.59198	-92.15626	Unknown - Iron	Missouri	1
12958	United States	Iron	36.59318	-81.61543	Wiley Farmer Tract	Virginia	1
12959	United States	Iron	36.59398	-91.53594	Unknown - Iron	Missouri	1
12960	United States	Iron	36.59698	-91.62904	Unknown - Iron	Missouri	1
12961	United States	Iron	36.58398	-87.93782	Tharpe Prospect	Tennessee	1

12962 rows × 7 columns

```
# Create the map with initial parameters
folium_map = folium.Map(location=[29.2985, 42.5510], zoom_start=2, tiles='OpenStreetMap')

# Add more tile layers
folium.TileLayer('Stamen Terrain' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB positron' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB dark_matter', attr="Map ").add_to(folium_map)

# Add the Esri Satellite tiles with attribution
esri_satellite = folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
    attr='Esri, Esri, DeLorme, NAVTEQ',
    name='Esri Satellite',
    overlay=False,
    control=True
)
esri_satellite.add_to(folium_map)

mc = MarkerCluster(name="Marker Cluster")

for index, row in iron_country.iterrows():
    popup_text="""
                <h1> Information Blog </h1><br>
                <li">Index Number: {} </li>
                <li>Site Name   : {:} </li>
                <li>Country     : {:} </li>
                <li>Commod1     : {:} </li>
                <li>State       : {:} </li>
        """.format(index,
                           row["site_name"],
                           row["country"],
                           row["commod1"],
                           row["state"])
    folium.CircleMarker(location=[row["latitude"],row["longitude"]],
                        color="red",popup=popup_text,
                        fill=True).add_to(mc)
folium.Popup(html=popup_text,sticky=True,max_width='250%',show=False).add_to(mc)
mc.add_to(folium_map)
folium.LayerControl().add_to(folium_map)
folium_map
```



SILVER

Description

Silver is highly prized due to the fact that is a rare precious metal with high economic value. Silver is one of the seven metals of antiquity that has been valued since ancient civilizations. Today, silver is valued as an investment medium, as well as for many commercial and industrial uses like for film development in photography, coatings for mirrors, jewelry, electrical equipment, medical devices, and dental equipment.

Sources

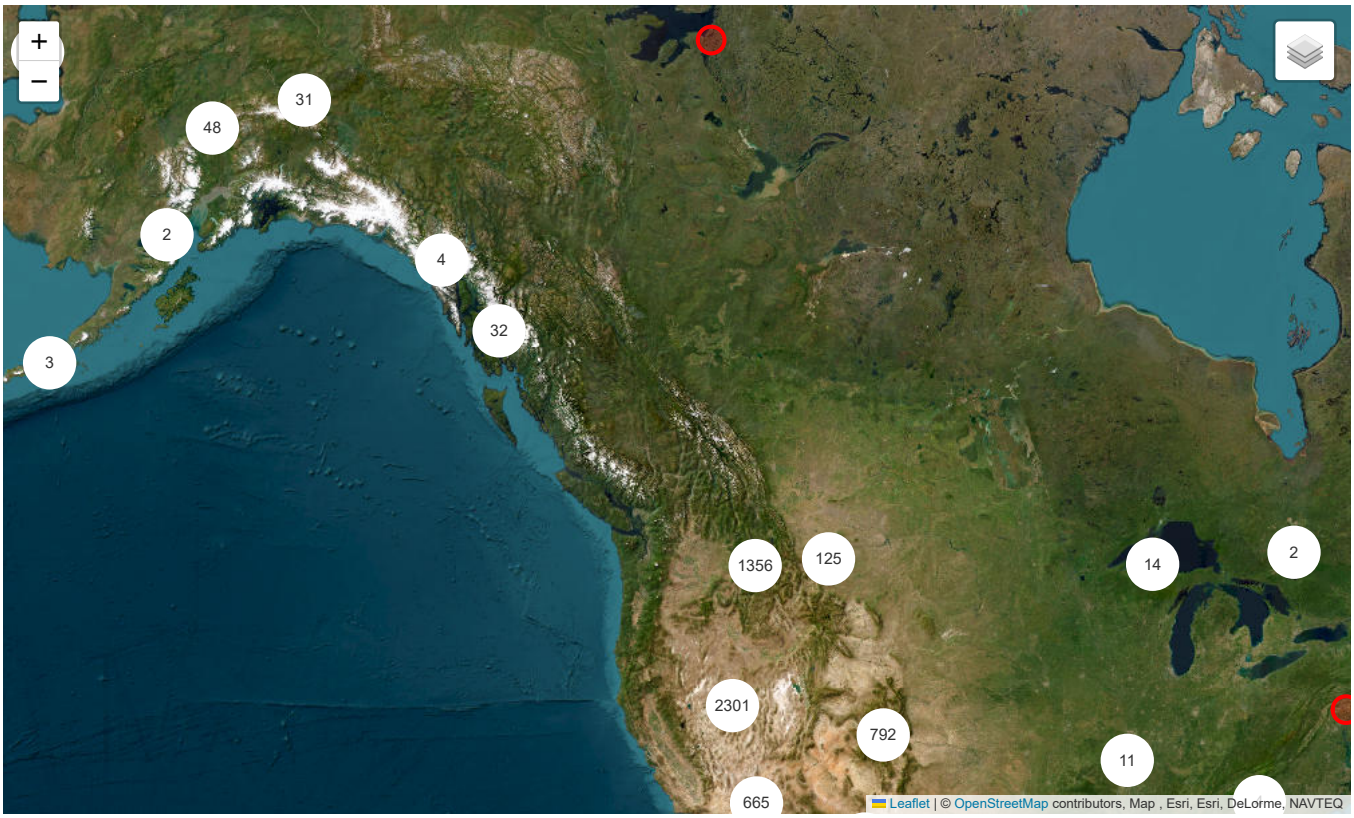
A majority of the world's silver mines are located in Peru, Bolivia, Mexico, China, Australia, Chile, Poland, and Serbia. The pure form of silver can be found in the Earth's crust, with the occurrence only being 0.08 parts per million. The top producing mines in order of production are Mexico (18.7%), China (15.1%), and Peru (14.1%). These mines are able to produce commercial fine-grade silver, which means that the silver is 99.9% pure.

```
silver_country = df[df["commod1"] == "Silver"]
silver_country = silver_country.groupby(['country','commod1','latitude','longitude','site_name','state'])['country'].count().reset_index
silver_country.sort_values(by='count',ascending=False,inplace=True,ignore_index=True)
silver_country
```

	country	commod1	latitude	longitude	site_name	state	count
0	United States	Silver	48.13346	-116.51026	Surprise Mine	Idaho	2
1	United States	Silver	47.52677	-115.98433	Silver Rock Prospect	Idaho	2
2	Mexico	Silver	28.51273	-110.75066	El Vidolin	Sonora	2
3	Mexico	Silver	30.85017	-107.65059	Tres Estrellas	Chihuahua	2
4	Mexico	Silver	28.65025	-104.15048	Alamos	Chihuahua	2
...	...	...	...	...	...	...	...
6251	United States	Silver	37.91217	-116.47756	Hillside Mine	Nevada	1
6252	United States	Silver	37.91136	-114.68804	Unnamed Mine	Nevada	1
6253	United States	Silver	37.91133	-117.18980	Silver King Claims	Nevada	1
6254	United States	Silver	37.91110	-107.70064	Unknown	Colorado	1
6255	United States	Silver	37.91688	-117.19424	Helen Claims	Nevada	1

6256 rows × 7 columns

```
# Create the map with initial parameters
folium_map = folium.Map(location=[29.2985, 42.5510], zoom_start=2, tiles='OpenStreetMap')

# Add more tile layers
folium.TileLayer('Stamen Terrain' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB positron' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB dark_matter' , attr="Map ").add_to(folium_map)

# Add the Esri Satellite tiles with attribution
esri_satellite = folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
    attr='Esri, Esri, DeLorme, NAVTEQ',
    name='Esri Satellite',
    overlay=False,
    control=True
)
esri_satellite.add_to(folium_map)

mc = MarkerCluster(name="Marker Cluster")

for index, row in silver_country.iterrows():
    popup_text="""
                <h1> Information Blog </h1><br>
                <li">Index Number: {} </li>
                <li>Site Name   : {:} </li>
                <li>Country     : {:} </li>
                <li>Commod1     : {:} </li>
                <li>State       : {:} </li>
        """.format(index,
                           row["site_name"],
                           row["country"],
                           row["commod1"],
                           row["state"])
    folium.CircleMarker(location=[row["latitude"],row["longitude"]],
                        color="red",popup=popup_text,
                        fill=True).add_to(mc)
folium.Popup(html=popup_text,sticky=True,max_width='250%',show=False).add_to(mc)
mc.add_to(folium_map)
folium.LayerControl().add_to(folium_map)
folium_map
```



India

```
India_country = df[df["country"] == "India"]
India_country = India_country.groupby(['country','commod1','latitude','longitude','site_name','state'])['country'].count().reset_index(n
India_country.sort_values(by='count',ascending=False,inplace=True,ignore_index=True)
India_country
```

	country	commod1	latitude	longitude	site_name	state	count
0	India	Manganese	21.97446	80.38367	Bijora	Madhya Pradesh	2
1	India	Aluminum	19.45026	76.13324	Satara Dist.	Maharashtra	2
2	India	Manganese	21.79947	80.10867	Barbaspur	Madhya Pradesh	1
3	India	Manganese	21.76615	80.11701	Amatola	Madhya Pradesh	1
4	India	Manganese	21.76615	80.04201	Waraseoni	Madhya Pradesh	1
...	...	...	...	...	...	...	...
334	India	Chromium	21.58278	85.58350	Boula Mining Area	Orissa	1
335	India	Chromium	21.28671	86.33018	Serrajuddin Nausahi-Low Grade	Orissa	1
336	India	Chromium	21.28671	86.33018	Serrajuddin Nausahi- High Grade	Orissa	1
337	India	Chromium	21.26621	86.31688	Nuasahi Chromite Mines	Orissa	1
338	India	Chromium, Ferrochrome	21.08283	86.60017	(Facility) Randia Baudpur Smelter	Orissa	1

339 rows × 7 columns

```
# Create the map with initial parameters
folium_map = folium.Map(location=[20.5937, 78.9629], zoom_start=4.5, tiles='OpenStreetMap')
```

```
# Add more tile layers
folium.TileLayer('Stamen Terrain' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB positron' , attr="Map ").add_to(folium_map)
folium.TileLayer('CartoDB dark_matter' , attr="Map ").add_to(folium_map)

# Add the Esri Satellite tiles with attribution
esri_satellite = folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}'
    attr='Esri, Esri, DeLorme, NAVTEQ',
    name='Esri Satellite',
    overlay=False,
    control=True
)
esri_satellite.add_to(folium_map)

mc = MarkerCluster(name="Marker Cluster")

for index, row in India_country.iterrows():
    popup_text="""
                <h1> Information Blog </h1><br>
                <li">Index Number: {} </li>
                <li>Site Name   : {:} </li>
                <li>Country     : {:} </li>
```