

AI COMPANY'S FINANCIAL ANALYSIS (2015 -2024)

OpenAI, Google Gemini & Meta AI: Comparing AI for Businesses



INTRODUCTION: -

This project focuses on analyzing financial market data to understand how AI-related investments and R&D spending influence company performance over time. Using **Python, MySQL, Power BI** the project performs data preprocessing, visualization, and correlation analysis to uncover meaningful insights from the dataset. The dataset spans from **2015 to 2024**, covering metrics such as R&D Spending (USD Mn), AI Revenue (USD Mn), Growth %, and Stock Impact %.

OBJECTIVES: -

- ❖ To analyze the relationship between R&D spending and AI revenue growth.
- ❖ To visualize company-wise stock performance over time.
- ❖ To analyze Company wise Yearly Profit Difference.
- ❖ To identify correlations between AI Revenue Growth and Stock Impact %.
- ❖ To provide data-driven insights for market trend prediction using Python.
- ❖ To identify how AI-related events influence stock market performance.
- ❖ To visualize long-term financial trends of companies like **OpenAI, Meta, and Google**.

DATASET DESCRIPTION: -

The dataset contains financial metrics from 2015 to 2024 for companies such as OpenAI ,Google, and Meta. It includes details such as R&D spending, AI revenue, revenue growth percentage, and stock impact percentage.

Columns:

Date	– Reporting date
Company	– Company name (e.g., OpenAI, Meta, Google)
R&D_Spending_USD_Mn	– Research and development spending (in million USD)
AI_Revenue_USD_Mn	– AI-related revenue (in million USD)
AI_Revenue_Growth_%	– Percentage growth in AI revenue
Event	– Notable AI-related events (if any)
Stock_Impact_%	– Stock performance percentage
Year	– Extracted year from Date
Month_Name	– Extracted month name from date

TOOLS & TECHNOLOGIES: -

Tool	Purpose
Python (Pandas, Matplotlib, Seaborn)	Data cleaning, preprocessing, and statistical analysis
MySQL	Data extraction, filtering, and aggregation
Power BI	Interactive dashboards and visual storytelling
CSV	Initial dataset source

METHODOLOGY: -

The project is divided into the following steps:

1. Data Loading and Inspection using Pandas
2. Data Cleaning and Preprocessing
3. Exploratory Data Analysis (EDA)
4. Correlation Analysis and Visualization
5. Data Storage and Analysis using SQL
6. Visualization using Power BI

DATA LOADING & INSPECTION USING PANDAS: -

Import the Data:

```
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
df = pd.read_csv(r"C:\Users\Genie\Downloads\AI_Financial.csv")
```

Show Data Types:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 10959 entries, 0 to 10958  
Data columns (total 7 columns):  
 #   Column           Non-Null Count  Dtype     
 ---  --    
 0   Date             10959 non-null    object    
 1   Company          10959 non-null    object    
 2   R&D_Spending_USD_Mn 10959 non-null    float64  
 3   AI_Revenue_USD_Mn 10959 non-null    float64  
 4   AI_Revenue_Growth_% 10959 non-null    float64  
 5   Event            233 non-null     object    
 6   Stock_Impact_%   10959 non-null    float64  
dtypes: float64(4), object(3)  
memory usage: 599.4+ KB
```

Statistical summary:

```
df.describe()
```

	Date	R&D_Spending_USD_Mn	AI_Revenue_USD_Mn	AI_Revenue_Growth_%	Stock_Impact_%
count	10959	10959.000000	10959.000000	10959.000000	10959.000000
mean	2020-01-01 00:00:00.000000256	65.184504	44.126571	159.395988	0.025560
min	2015-01-01 00:00:00	1.570000	-0.550000	-155.430000	-3.000000
25%	2017-07-02 00:00:00	8.640000	3.610000	43.870000	-0.500000
50%	2020-01-01 00:00:00	70.960000	35.220000	133.750000	0.000000
75%	2022-07-02 00:00:00	99.600000	71.680000	258.300000	0.510000
max	2024-12-31 00:00:00	163.830000	155.960000	565.500000	18.500000
std	NaN	47.918247	41.639356	135.462185	0.749513

DATA CLEANING AND PREPROCESSING: -

Convert Datatype of Date column into Datetime format:

```
df['Date'] = pd.to_datetime(df['Date'])
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10959 entries, 0 to 10958
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype  
--- 
 0   Date              10959 non-null   datetime64[ns]
 1   Company           10959 non-null   object  
 2   R&D_Spending_USD_Mn 10959 non-null   float64 
 3   AI_Revenue_USD_Mn  10959 non-null   float64 
 4   AI_Revenue_Growth_% 10959 non-null   float64 
 5   Event             10959 non-null   object  
 6   Stock_Impact_%    10959 non-null   float64 
 7   Year              10959 non-null   int32  
 8   Month_Name        10959 non-null   object  
dtypes: datetime64[ns](1), float64(4), int32(1), object(3)
memory usage: 727.9+ KB
```

Check Missing Values:

```
df.isnull().sum()
```

```
Date          0
Company       0
R&D_Spending_USD_Mn 0
AI_Revenue_USD_Mn  0
AI_Revenue_Growth_% 0
Event         10726
Stock_Impact_% 0
dtype: int64
```

Fill Missing Values:

```
df['Event'].fillna('No Event', inplace = True)
```

Check Null Values After Using Fillna Method:

```
df.isnull().sum()
```

```
Date          0
Company       0
R&D_Spending_USD_Mn 0
AI_Revenue_USD_Mn  0
AI_Revenue_Growth_% 0
Event         0
Stock_Impact_%  0
dtype: int64
```

Show company Names:

```
df['Company'].unique()  
array(['OpenAI', 'Google', 'Meta'], dtype=object)
```

Create new column for 'Year':

```
df['Year'] = df['Date'].dt.year  
df.head()
```

	Date	Company	R&D_Spending_USD_Mn	AI_Revenue_USD_Mn	AI_Revenue_Growth_%	Event	Stock_Impact_%	Year
0	2015-01-01	OpenAI	5.92	0.63	-36.82	NaN	-0.36	2015
1	2015-01-02	OpenAI	5.41	1.81	80.59	NaN	0.41	2015
2	2015-01-03	OpenAI	4.50	0.61	-38.88	NaN	0.23	2015
3	2015-01-04	OpenAI	5.45	0.95	-5.34	NaN	0.93	2015
4	2015-01-05	OpenAI	3.40	1.48	48.45	NaN	-0.09	2015

Create new column for 'Month':

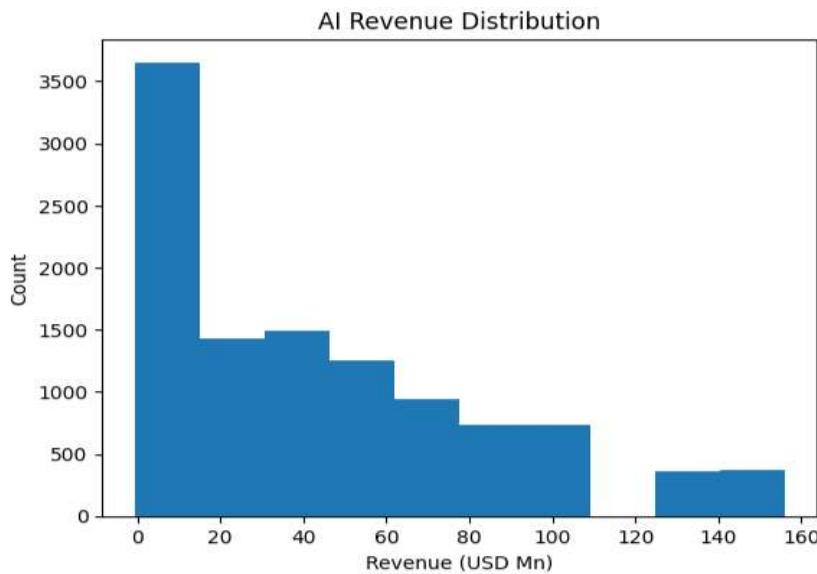
```
df['Month_Name'] = df['Date'].dt.month_name()  
df.head()
```

	Date	Company	R&D_Spending_USD_Mn	AI_Revenue_USD_Mn	AI_Revenue_Growth_%	Event	Stock_Impact_%	Year	Month_Name
0	2015-01-01	OpenAI	5.92	0.63	-36.82	NaN	-0.36	2015	January
1	2015-01-02	OpenAI	5.41	1.81	80.59	NaN	0.41	2015	January
2	2015-01-03	OpenAI	4.50	0.61	-38.88	NaN	0.23	2015	January
3	2015-01-04	OpenAI	5.45	0.95	-5.34	NaN	0.93	2015	January
4	2015-01-05	OpenAI	3.40	1.48	48.45	NaN	-0.09	2015	January

EXPLORATORY DATA ANALYSIS (EDA): -

Revenue Distribution:

```
plt.hist(df['AI_Revenue_USD_Mn'])  
plt.title("AI Revenue Distribution")  
plt.xlabel("Revenue (USD Mn)")  
plt.ylabel("Count")  
plt.show()
```



R&D Expenses Distribution:

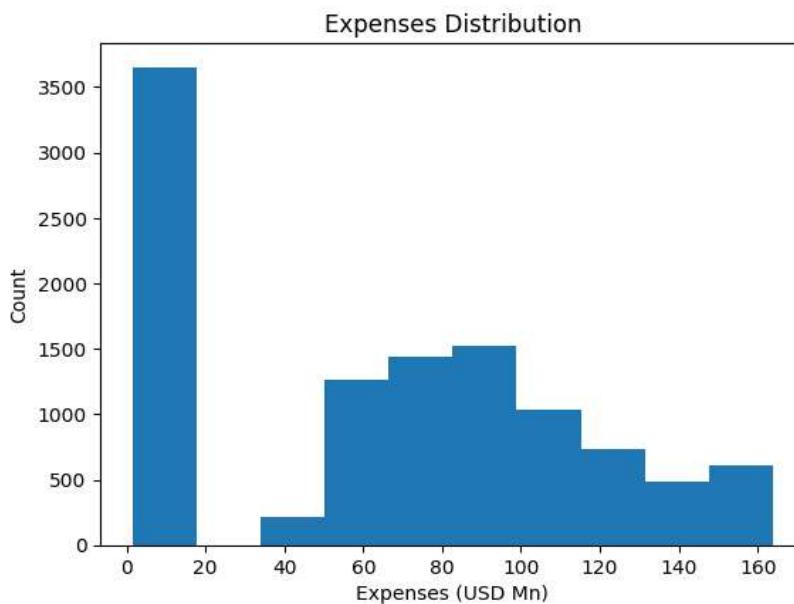
```
plt.hist(df['R&D_Spending_USD_Mn'])

plt.title("Expenses Distribution")

plt.xlabel("Expenses (USD Mn)")

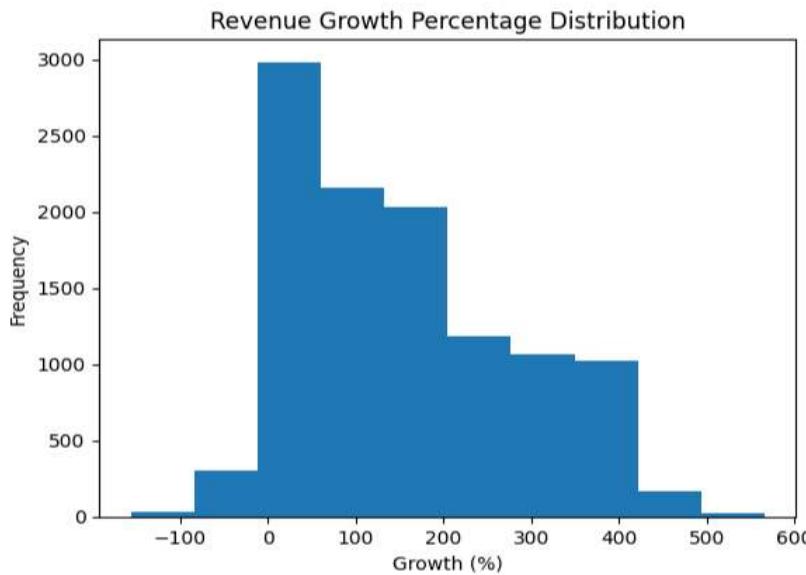
plt.ylabel("Count")

plt.show()
```



Revenue Growth Percentage Distribution:

```
plt.hist(df['AI_Revenue_Growth_%'])  
plt.title("Revenue Growth Percentage Distribution")  
plt.xlabel("Growth (%)")  
plt.ylabel("Frequency")  
plt.show()
```



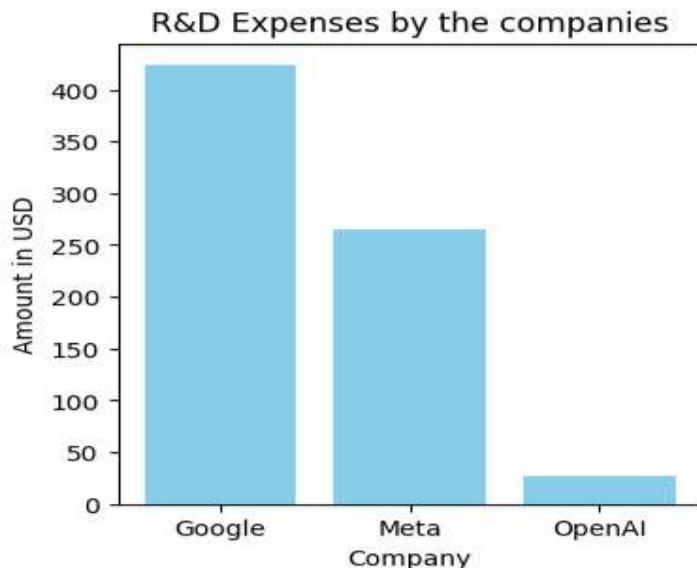
CORRELATION ANALYSIS AND VISUALIZATION: -

How much amount companies Expenses on R&D(Convert Million USD to Billion USD):

```
print("Companies Expenses for R&D in Billion USD")  
  
RD = df.groupby('Company')['R&D_Spending_USD_Mn'].sum()/1000  
  
RD  
  
Companies Expenses for R&D in Billion USD  
Company  
Google      423.34114  
Meta        264.53307  
OpenAI      26.48277  
Name: R&D_Spending_USD_Mn, dtype: float64
```

Bar plot to show the amount spent on R&D by companies:

```
plt.bar(RD.index,RD.values, color= ['SkyBlue'])  
plt.title('R&D Expenses by the companies')  
plt.xlabel('Company')  
plt.ylabel('Amount in USD')  
plt.show()
```



Revenue Earned by the companies (Convert Million USD to Billion USD):

```
print("Companies AI Revenue in Billion USD")  
REV = df.groupby('Company')['AI_Revenue_USD_Mn'].sum()/1000  
REV
```

Companies AI Revenue in Billion USD

Company	Revenue (Billion USD)
Google	284.49838
Meta	189.62182
OpenAI	9.46289

Name: AI_Revenue_USD_Mn, dtype: float64

Bar plot to show the Revenue by companies:

```
plt.figure(figsize=(4, 4))

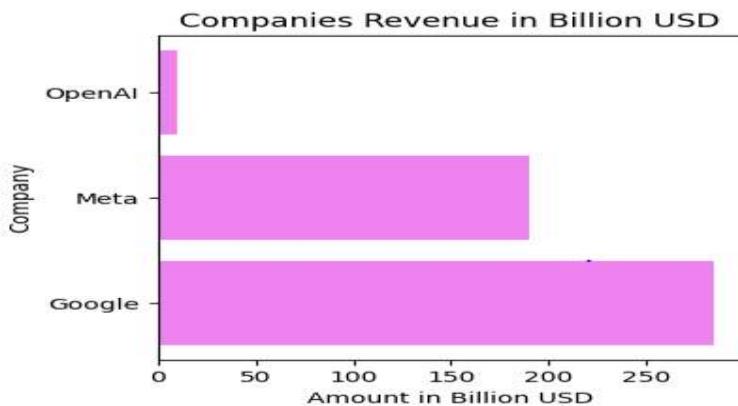
plt.barh(REV.index,REV.values, color= ['Violet'])

plt.title('Companies Revenue in Billion USD')

plt.xlabel('Amount in Billion USD')

plt.ylabel('Company')

plt.show()
```



Subplot:

```
plt.figure(figsize =(10,5))

plt.subplot(1,2,1)

plt.bar(RD.index,RD.values, color= ['SkyBlue'])

plt.title('R&D spending by the companies')

plt.xlabel('Company')

plt.ylabel('Amount in USD')

plt.subplot(1,2,2)

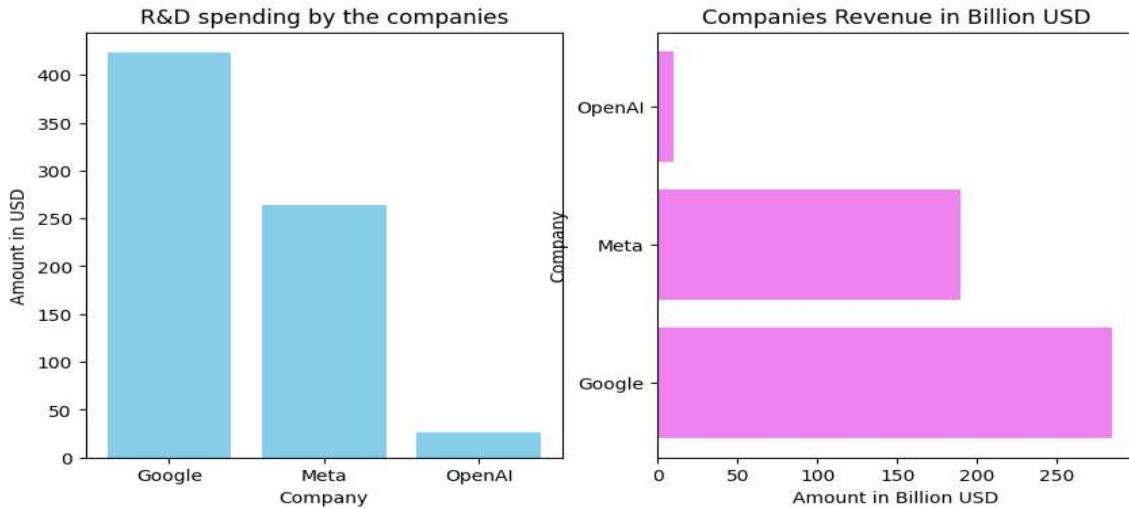
plt.barh(REV.index,REV.values, color= ['Violet'])

plt.title('Companies Revenue in Billion USD')

plt.xlabel('Amount in Billion USD')

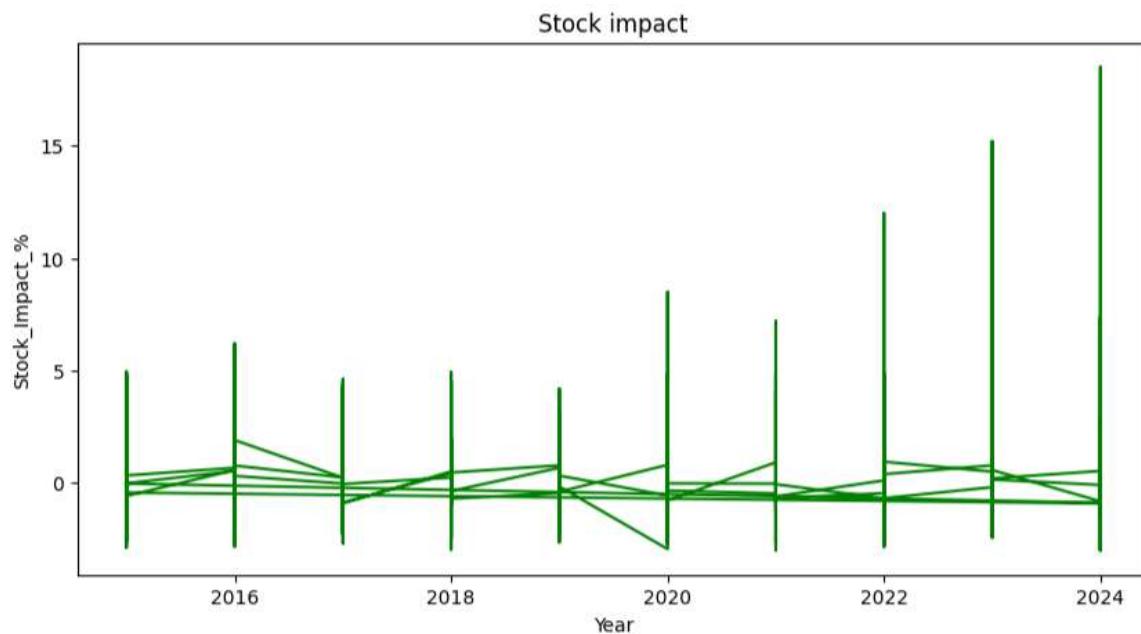
plt.ylabel('Company')

plt.show()
```



Year wise Impact on the Stock:

```
plt.figure(figsize = (10,5))
plt.plot(df['Year'],df['Stock_Impact_%'],color = 'Green')
plt.title('Stock impact')
plt.xlabel('Year')
plt.ylabel('Stock_Impact_%')
plt.show()
```



Line plot to show the stock value of companies Year wise:

```
data_Company = df[df['Company'].isin(['OpenAI','Google','Meta'])]

plt.figure(figsize = (10,5))

plt.plot(data_Company['Date'],data_Company['Stock_Impact_%'])

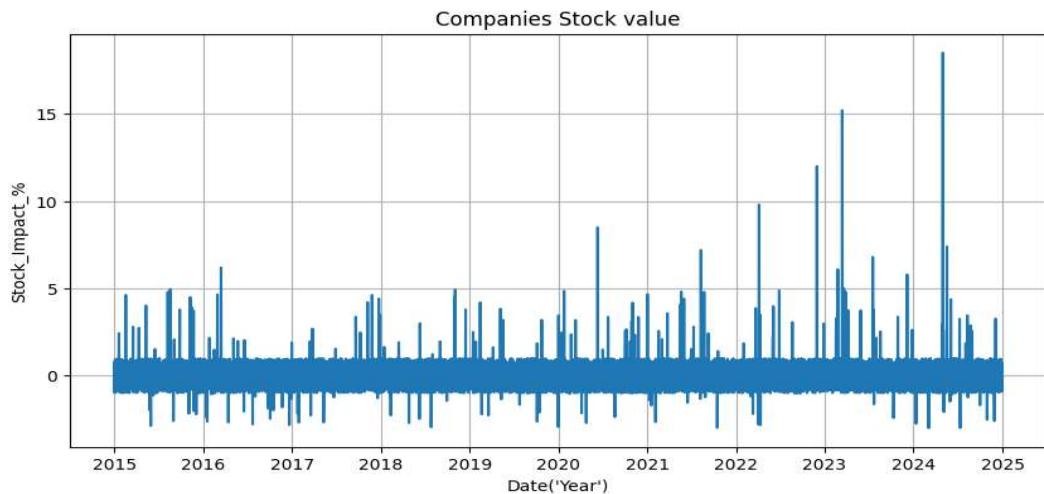
plt.title('Companies Stock value')

plt.xlabel("Date('Year')")

plt.ylabel('Stock_Impact_%')

plt.grid(True)

plt.show()
```



Pie chart of Stock Impact by Company wise:

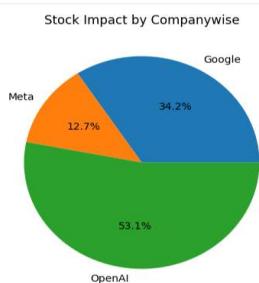
```
data = df.groupby('Company')['Stock_Impact_%'].sum()

plt.figure(figsize =(5,8))

plt.pie( data,labels = data.index ,autopct='%.1f%%')

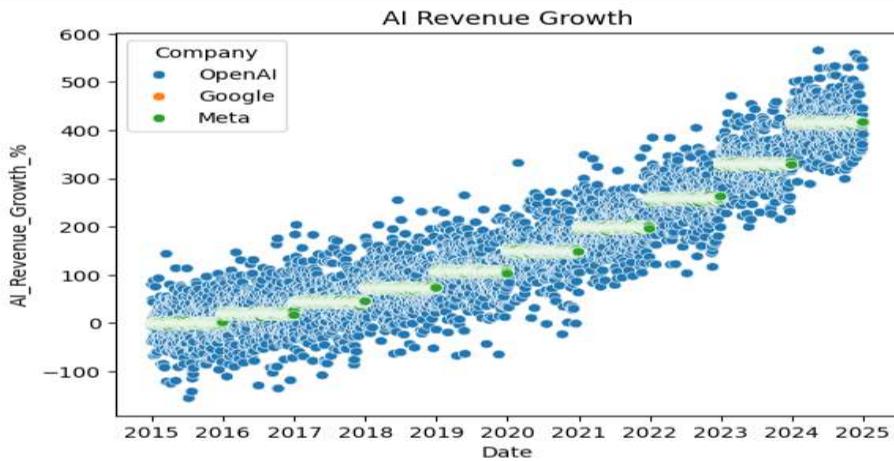
plt.title("Stock Impact by Companywise")

plt.show()
```



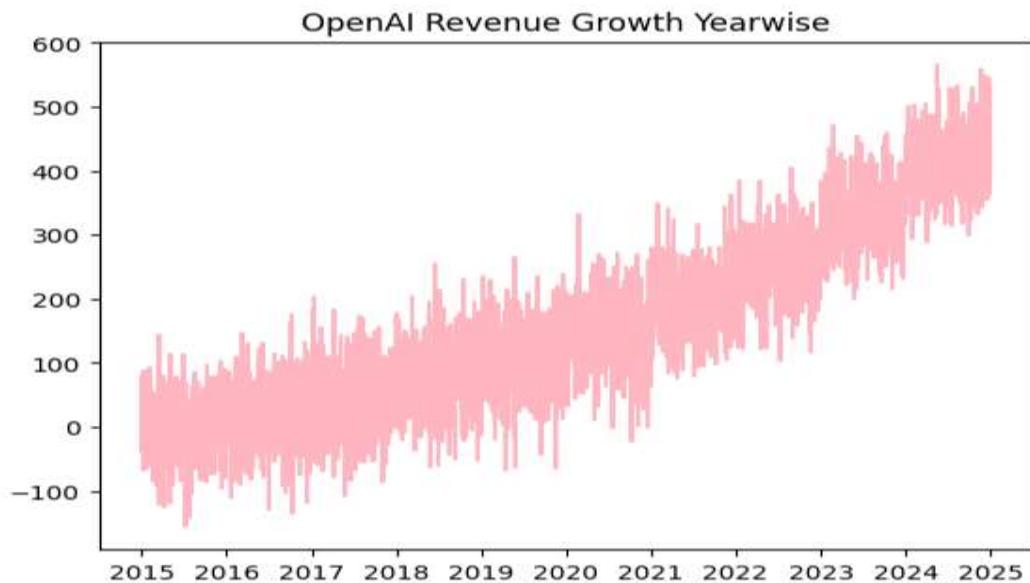
Scatterplot AI Revenue growth of the company:

```
sns.scatterplot(x ='Date',y = "AI_Revenue_Growth_%",data = df,hue = 'Company')  
plt.title('AI Revenue Growth')  
plt.show()
```



OpenAI Revenue Growth Year by Year:

```
plt.plot(data_OpenAI['Date'],data_OpenAI['AI_Revenue_Growth_%'],color= 'lightpink')  
plt.title ('OpenAI Revenue Growth Yearwise')  
plt.show()
```

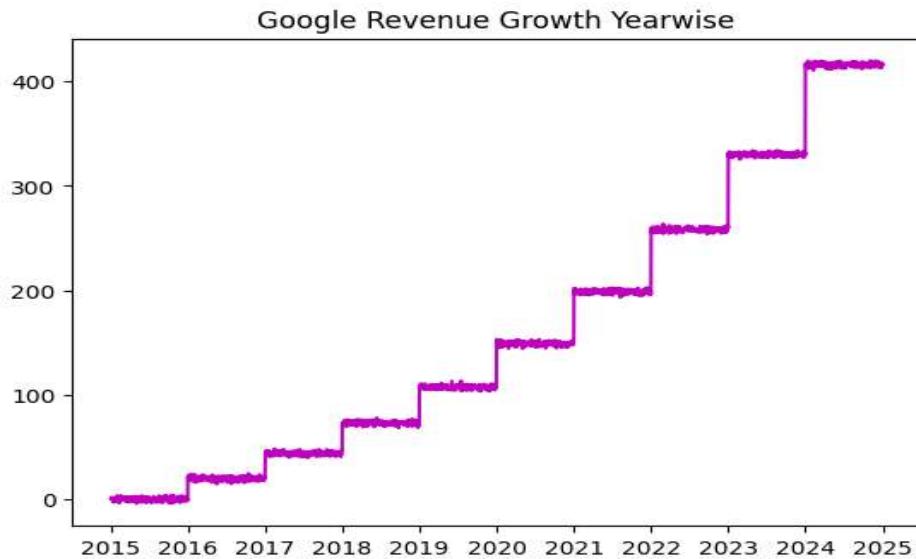


Google Revenue Growth Year by Year:

```
plt.plot(data_Google['Date'],data_Google['AI_Revenue_Growth_%'],color= 'm')

plt.title ('Google Revenue Growth Yearwise')

plt.show()
```

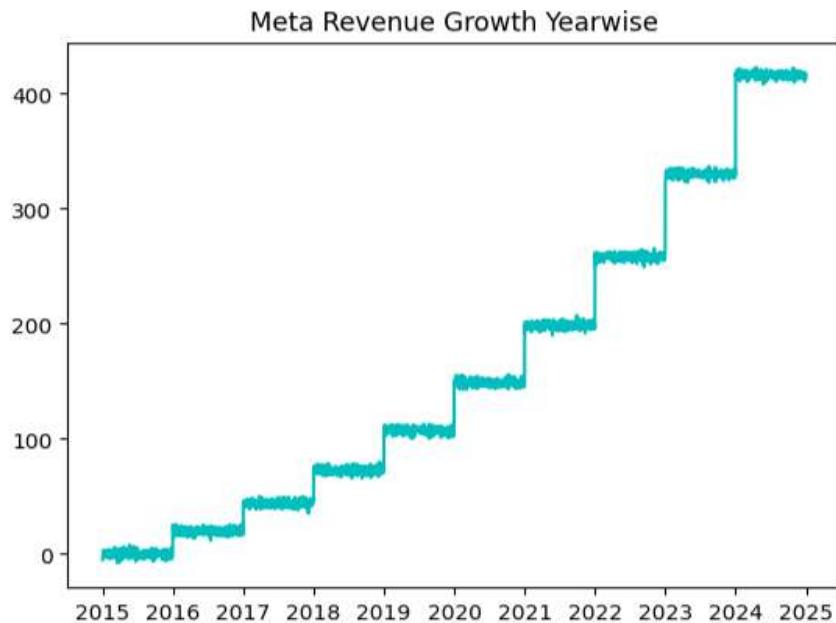


Meta Revenue Growth Year by Year:

```
plt.plot(data_Meta['Date'],data_Meta['AI_Revenue_Growth_%'],color= 'c')

plt.title ('Meta Revenue Growth Yearwise')

plt.show()
```



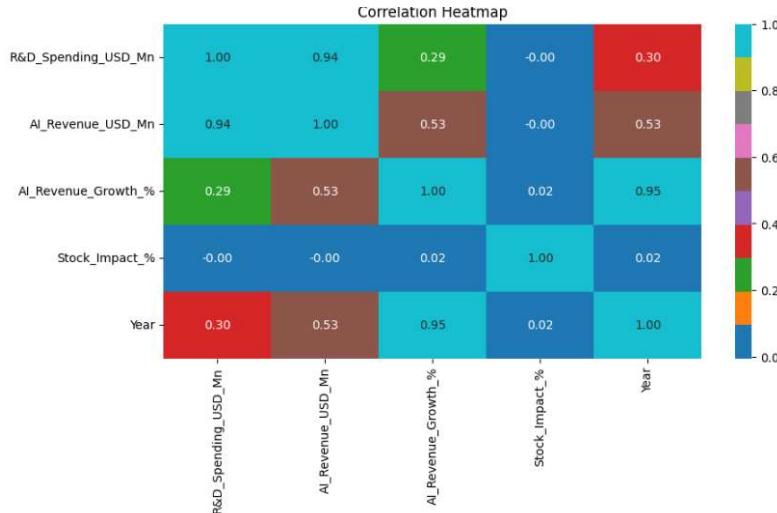
Heatmap:

```
plt.figure(figsize=(10,5))

sns.heatmap(df.corr(numeric_only=True), cmap='tab10', annot=True, fmt='.2f')

plt.title("Correlation Heatmap")

plt.show()
```



Maximum Stock impact on companies & Year:

```
stocks = df.groupby(['Year','Company'])['Stock_Impact_%'].max()
```

```
stocks
```

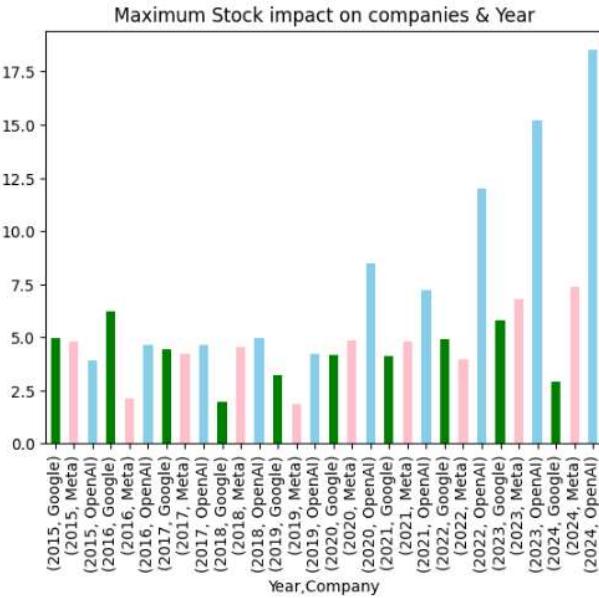
```
Year Company
2015 Google 4.96
      Meta 4.80
      OpenAI 3.92
2016 Google 6.20
      Meta 2.14
      OpenAI 4.66
2017 Google 4.42
      Meta 4.21
      OpenAI 4.63
2018 Google 1.97
      Meta 4.54
      OpenAI 4.94
2019 Google 3.20
      Meta 1.86
      OpenAI 4.20
2020 Google 4.18
      Meta 4.86
      OpenAI 8.50
2021 Google 4.10
      Meta 4.80
      OpenAI 7.20
2022 Google 4.88
      Meta 3.98
      OpenAI 12.00
2023 Google 5.80
      Meta 6.80
      OpenAI 15.20
2024 Google 2.89
      Meta 7.40
      OpenAI 18.50
Name: Stock_Impact_%, dtype: float64
```

Bar Plot for Maximum Stock impact on companies & Year:

```
stocks.plot(kind = 'bar',color = ['GREEN','PINK','SKYBLUE'])

plt.title("Maximum Stock impact on companies & Year")

plt.show()
```



Area Chart for Companies Spending R&D in USD_Mn:

```
data = df.groupby('Company')['R&D_Spending_USD_Mn'].sum()

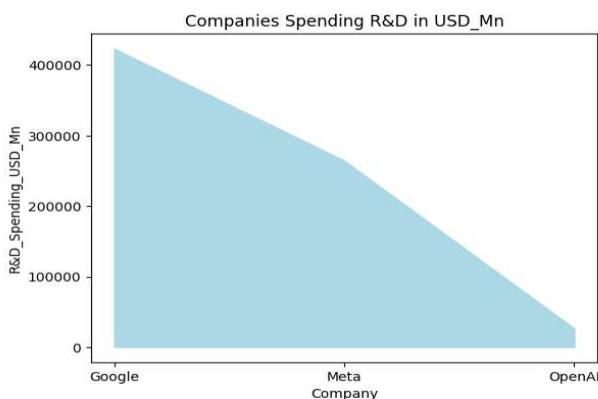
plt.fill_between(data.index,data.values, color= "lightblue")

plt.xlabel('Company')

plt.ylabel('R&D_Spending_USD_Mn')

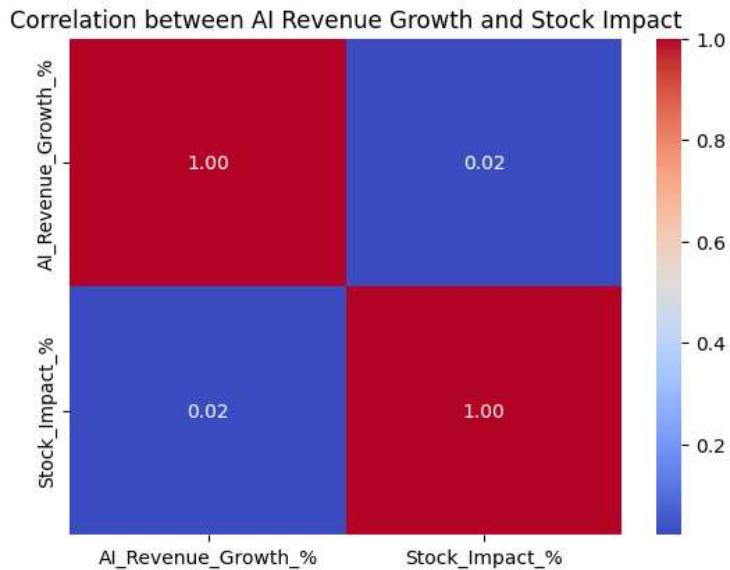
plt.title("Companies Spending R&D in USD_Mn")

plt.show()
```



HeatMap for Revenue Growth and Stock Impact:

```
corr_data = df[['AI_Revenue_Growth_%', 'Stock_Impact_%']]  
  
corr_matrix = corr_data.corr()  
  
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f")  
  
plt.title('Correlation between AI Revenue Growth and Stock Impact')  
  
plt.show()
```



Daily Average stock impact of the companies:

```
df.groupby('Company')['Stock_Impact_%'].mean()*100
```

```
Company  
Google      2.620860  
Meta        0.976184  
OpenAI      4.070901  
Name: Stock_Impact_%, dtype: float64
```

Daily Average Expenses on R&D by the companies:

```
df.groupby('Company')['R&D_Spending_USD_Mn'].mean()
```

```
Company  
Google      115.888623  
Meta        72.415294  
OpenAI      7.249595  
Name: R&D_Spending_USD_Mn, dtype: float64
```

Maximum impact % on a company Stocks:

```
df.groupby('Company')['Stock_Impact_%'].max()
```

```
Company
Google      6.2
Meta        7.4
OpenAI     18.5
Name: Stock_Impact_%, dtype: float64
```

Maximum Revenue Growth Percentage:

```
df.sort_values(by= 'AI_Revenue_Growth_%', ascending =False)
```

	Date	Company	R&D_Spending_USD_Mn	AI_Revenue_USD_Mn	AI_Revenue_Growth_%	Event	Stock_Impact_%	Year	Month_Name
3423	2024-05-16	OpenAI	8.47	6.65	565.50	No Event	0.51	2024	May
3612	2024-11-21	OpenAI	10.16	6.59	558.70	No Event	0.63	2024	November
3627	2024-12-06	OpenAI	8.51	6.49	548.83	No Event	-0.15	2024	December
3645	2024-12-24	OpenAI	8.79	6.46	546.07	No Event	0.45	2024	December
3508	2024-08-09	OpenAI	9.92	6.33	532.86	No Event	0.77	2024	August
...
98	2015-04-09	OpenAI	4.80	-0.25	-125.50	No Event	-0.56	2015	April
548	2016-07-02	OpenAI	5.19	-0.29	-129.11	No Event	-0.31	2016	July
649	2016-10-11	OpenAI	5.37	-0.35	-135.31	No Event	0.12	2016	October
205	2015-07-25	OpenAI	5.23	-0.42	-141.61	No Event	-0.62	2015	July
189	2015-07-09	OpenAI	5.66	-0.55	-155.43	No Event	0.70	2015	July

10959 rows × 9 columns

Maximum Stock Impact on OpenAI Events:

```
data_OpenAI.sort_values(by = 'Stock_Impact_%' ,ascending = False)
```

:	Date	Company	R&D_Spending_USD_Mn	AI_Revenue_USD_Mn	AI_Revenue_Growth_%	Event	Stock_Impact_%	Year	Month_Name
3408	2024-05-01	OpenAI	10.91	5.34	434.27	GPT-5 release (predicted)	18.50	2024	May
2994	2023-03-14	OpenAI	7.78	4.05	304.57	GPT-4 release	15.20	2023	March
2890	2022-11-30	OpenAI	10.60	3.18	217.72	ChatGPT (GPT-3.5) launch	12.00	2022	November
2652	2022-04-06	OpenAI	9.24	3.48	247.93	DALL-E 2 release	9.80	2022	April
1988	2020-06-11	OpenAI	5.90	2.62	161.56	GPT-3 release	8.50	2020	June
...
3590	2024-10-30	OpenAI	11.05	5.27	427.50	AI-powered search update	-2.52	2024	October
1212	2018-04-27	OpenAI	5.38	1.76	76.47	AI ethics policy update	-2.70	2018	April
1303	2018-07-27	OpenAI	7.44	0.97	-2.63	AI-powered search update	-2.95	2018	July
2480	2021-10-16	OpenAI	8.39	2.78	178.28	AI Ads Optimization upgrade	-2.98	2021	October
3350	2024-03-04	OpenAI	10.82	4.77	376.53	AI Ads Optimization upgrade	-3.00	2024	March

3653 rows × 9 columns

Maximum Stock Impact on Google Events:

data_Google.sort_values(by = 'Stock_Impact_%' ,ascending = False)

	Date	Company	R&D_Spending_USD_Mn	AI_Revenue_USD_Mn	AI_Revenue_Growth_%	Event	Stock_Impact_%	Year	Month_Name
4092	2016-03-15	Google	84.56	36.22	20.73	AlphaGo beats Lee Sedol	6.20	2016	March
6914	2023-12-06	Google	146.59	129.17	330.55	Gemini AI release	5.80	2023	December
6654	2023-03-21	Google	149.34	129.76	332.53	Bard chatbot launch	5.00	2023	March
3883	2015-08-19	Google	79.27	30.89	2.98	AI partnership deal	4.96	2015	August
6388	2022-06-28	Google	137.71	107.96	259.85	AI ethics policy update	4.88	2022	June
...
4122	2016-04-14	Google	87.19	36.08	20.28	AI partnership deal	-2.67	2016	April
4412	2017-01-29	Google	92.21	43.25	44.17	AI Ads Optimization upgrade	-2.68	2017	January
6303	2022-04-04	Google	138.00	106.81	256.04	AI Video Recommendation upgrade	-2.80	2022	April
3803	2015-05-31	Google	81.14	29.30	-2.32	Cloud AI launch	-2.87	2015	May
5479	2020-01-01	Google	117.24	74.82	149.41	AI partnership deal	-2.93	2020	January

3653 rows × 9 columns

Maximum Stock Impact on Meta Events:

data_Meta.sort_values(by = 'Stock_Impact_%' ,ascending = False)

	Date	Company	R&D_Spending_USD_Mn	AI_Revenue_USD_Mn	AI_Revenue_Growth_%	Event	Stock_Impact_%	Year	Month_Name
10731	2024-05-18	Meta	103.64	103.05	415.23	LLaMA 3 release (predicted)	7.40	2024	May
10426	2023-07-18	Meta	92.44	85.67	328.37	LLaMA 2 release	6.80	2023	July
10282	2023-02-24	Meta	93.71	86.98	334.89	LLaMA 1 release	6.10	2023	February
9156	2020-01-25	Meta	72.73	49.10	145.49	Cloud AI launch	4.86	2020	January
7526	2015-08-09	Meta	48.97	19.95	-0.27	AI Video Recommendation upgrade	4.80	2015	August
...
10604	2024-01-12	Meta	99.37	104.28	421.42	AI Video Recommendation upgrade	-2.75	2024	January
7875	2016-07-23	Meta	53.76	23.97	19.83	AI-powered search update	-2.79	2016	July
8026	2016-12-21	Meta	53.18	23.42	17.11	AI-powered search update	-2.82	2016	December
9964	2022-04-12	Meta	84.90	71.99	259.95	AI ethics policy update	-2.83	2022	April
10785	2024-07-11	Meta	100.78	102.92	414.60	AI ethics policy update	-2.99	2024	July

3653 rows × 9 columns

DATA STORAGE AND ANALYSIS USING SQL: -

FIND YEARWISE COMPANIES POSITIVE STOCK IMPACT:

```
select distinct `Year`, Company, `Stock_Impact_%` from ai_data where  
`Stock_Impact_%` >=0 order by `Stock_Impact_%` desc ;
```

The screenshot shows a database query results grid titled "Result Grid". The grid has three columns: "Year", "Company", and "Stock_Impact_%". The data is as follows:

Year	Company	Stock_Impact_%
2024	OpenAI	18.5
2023	OpenAI	15.2
2022	OpenAI	12
2022	OpenAI	9.8
2020	OpenAI	8.5
2024	Meta	7.4
2021	OpenAI	7.2
2023	Meta	6.8
2016	Google	6.2
2023	Meta	6.1
2023	Google	5.8
2023	Google	5
2015	Google	4.96

FIND R&D_EXPENSES PER COMPANY:

```
select Company,sum(`R&D_Spending_USD_Mn`) as RD_Expenses from ai_data group  
by company ;
```

The screenshot shows a database query results grid titled "Result Grid". The grid has two columns: "Company" and "RD_Expenses". The data is as follows:

Company	RD_Expenses
OpenAI	26482.77000000004
Google	423341.140000005
Meta	264533.070000001

FIND COMPANIES TOTAL AI REVENUE:

```
select Company,sum(`AI_Revenue_USD_Mn`) as AI_Revenue from ai_data group by  
company;
```

The screenshot shows a database query results grid titled "Result Grid". The grid has two columns: "Company" and "AI_Revenue". The data is as follows:

Company	AI_Revenue
OpenAI	9462.8899999998
Google	284498.3800000003
Meta	189621.8200000018

FIND AI_REVENUE GROWTH PERCENTAGE PER COMPANY:

```
select Company,sum(`AI_Revenue_Growth_%`) as AI_Revenue_Growth from ai_data  
group by company order by AI_Revenue_Growth desc;
```

The screenshot shows a database result grid titled 'Result Grid' with the following data:

Company	AI_Revenue_Growth
Google	583027.2499999997
Meta	582807.7400000001
OpenAI	580985.6399999996

FIND AI REVENUE GROWTH LEVEL USING CASE STATEMENT:

```
select Company,'Year','AI_Revenue_Growth_%',  
case  
when `AI_Revenue_Growth_%` <=0 then 'NEGATIVE'  
when `AI_Revenue_Growth_%` <=200 then 'LOW'  
when `AI_Revenue_Growth_%` <350 then 'MEDIUM'  
when `AI_Revenue_Growth_%` >350 then 'HIGH'  
end as growth_level from ai_data order by `AI_Revenue_Growth_%` ;
```

The screenshot shows a database result grid titled 'Result Grid' with the following data:

Company	Year	AI_Revenue_Growth_%	growth_level
OpenAI	2015	-155.43	NEGATIVE
OpenAI	2015	-141.61	NEGATIVE
OpenAI	2016	-135.31	NEGATIVE
OpenAI	2016	-129.11	NEGATIVE
OpenAI	2015	-125.5	NEGATIVE
OpenAI	2015	-121.12	NEGATIVE
OpenAI	2015	-119.29	NEGATIVE
OpenAI	2016	-118.28	NEGATIVE
OpenAI	2015	-114.87	NEGATIVE
OpenAI	2016	-110.58	NEGATIVE
OpenAI	2017	-107.93	NEGATIVE
OpenAI	2015	-106.06	NEGATIVE
OpenAI	2016	-102.96	NEGATIVE

FIND MINIMUM AND MAXIMUM AI REVENUE BY COMPANYWISE:

```
select Company,min(`AI_Revenue_USD_Mn`) as Low_Revenue,  
max(`AI_Revenue_USD_Mn`) as High_Revenue from ai_data group by Company;
```

The screenshot shows a database result grid titled 'Result Grid' with the following data:

Company	Low_Revenue	High_Revenue
OpenAI	-0.515	104.6
Google	28.7	155.96
Meta	18.44	104.6

FIND PROFIT DIFFERENCE OF R&D EXPENSES VS AI REVENUE BY COMPANYWISE:

select Company,

 SUM(`AI_Revenue_USD_Mn`) as Total_Revenue,

 SUM(`R&D_Spending_USD_Mn`) as Total_Expenses,

 (SUM(`AI_Revenue_USD_Mn`) - SUM(`R&D_Spending_USD_Mn`)) as

 Profit_Difference from ai_data group by Company;

The screenshot shows a database query results grid titled "Result Grid". The grid has columns: Company, Total_Revenue, Total_Expenses, and Profit_Difference. The data is as follows:

Company	Total_Revenue	Total_Expenses	Profit_Difference
OpenAI	9462.889999999998	26482.770000000004	-17019.880000000005
Google	284498.3800000003	423341.1400000005	-138842.76000000018
Meta	189621.82000000018	264533.0700000001	-74911.24999999994

Below the grid, it says "Result 13" and there is a "Read Only" status indicator.

FIND COMPANIES EVENT IN THE YEAR OF “2024”:

select Company,Event,Year from ai_data where `Year` = 2024 and Event <> 'No Event' order by Company;

The screenshot shows a database query results grid titled "Result Grid". The grid has columns: Company, Event, and Year. The data is as follows:

Company	Event	Year
Google	AI-powered search update	2024
Google	AI speech recognition release	2024
Google	AI-powered search update	2024
Google	AI partnership deal	2024
Google	AI-powered search update	2024
Meta	AI Video Recommendation upgrade	2024
Meta	Cloud AI launch	2024
Meta	AI Video Recommendation upgrade	2024
Meta	AI Video Recommendation upgrade	2024
Meta	AI-powered search update	2024
Meta	AI Ads Optimization upgrade	2024
Meta	AI speech recognition release	2024
Meta	LLaMA 3 release (predicted)	2024
Meta	AI Ads Optimization upgrade	2024
Meta	AI speech recognition release	2024
Meta	AI speech recognition release	2024
Meta	AI ethics policy update	2024
Meta	AI ethics policy update	2024
OpenAI	AI Ads Optimization upgrade	2024

Below the grid, it says "ai_data 20" and there is a "Read Only" status indicator.

FIND COMPANYWISE YEARLY PROFIT DIFFERENCE:

Select Company,Year,

```
SUM(`AI_Revenue_USD_Mn`) - SUM(`R&D_Spending_USD_Mn`) AS Profit_Diff from  
ai_data group by Year, Company order by Year, Company;
```

The screenshot shows a Power BI result grid titled "Result Grid". The data is presented in a table with columns: Company, Year, and Profit_Diff. The table contains 15 rows of data, each representing a company and its profit difference for a specific year. The data includes entries for Google, Meta, OpenAI, and other companies across years 2015 to 2019. The interface includes standard Power BI navigation elements like "Filter Rows", "Export", and "Wrap Cell Content". A sidebar on the right provides access to "Result Grid", "Form Editor", and "Field Types".

Company	Year	Profit_Diff
Google	2015	-18227.25
Meta	2015	-10943.959999999997
OpenAI	2015	-1509.809999999995
Google	2016	-18487.270000000008
Meta	2016	-10995.679999999978
OpenAI	2016	-1505.790000000004
Google	2017	-18313.209999999985
Meta	2017	-10758.930000000022
OpenAI	2017	-1626.180000000007
Google	2018	-17869.589999999982
Meta	2018	-10351.59000000001
OpenAI	2018	-1616.54
Google	2019	-17017.890000000032

FIND COMPANIES AVG REVENUE GROWTH:

```
select Company,avg(`AI_Revenue_Growth %`) as AVG_Revenue_Growth from  
ai_data group by Company order by AVG_Revenue_Growth desc ;
```

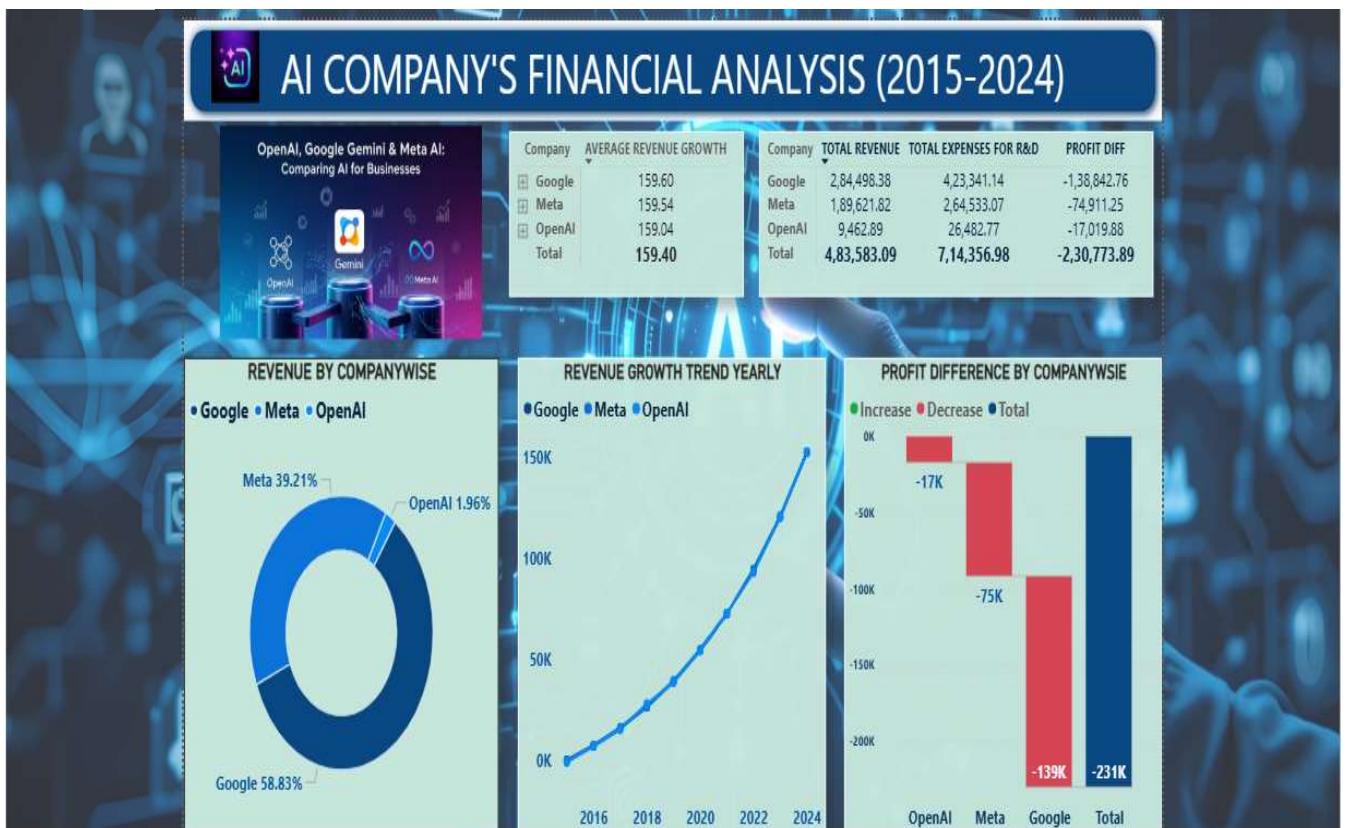
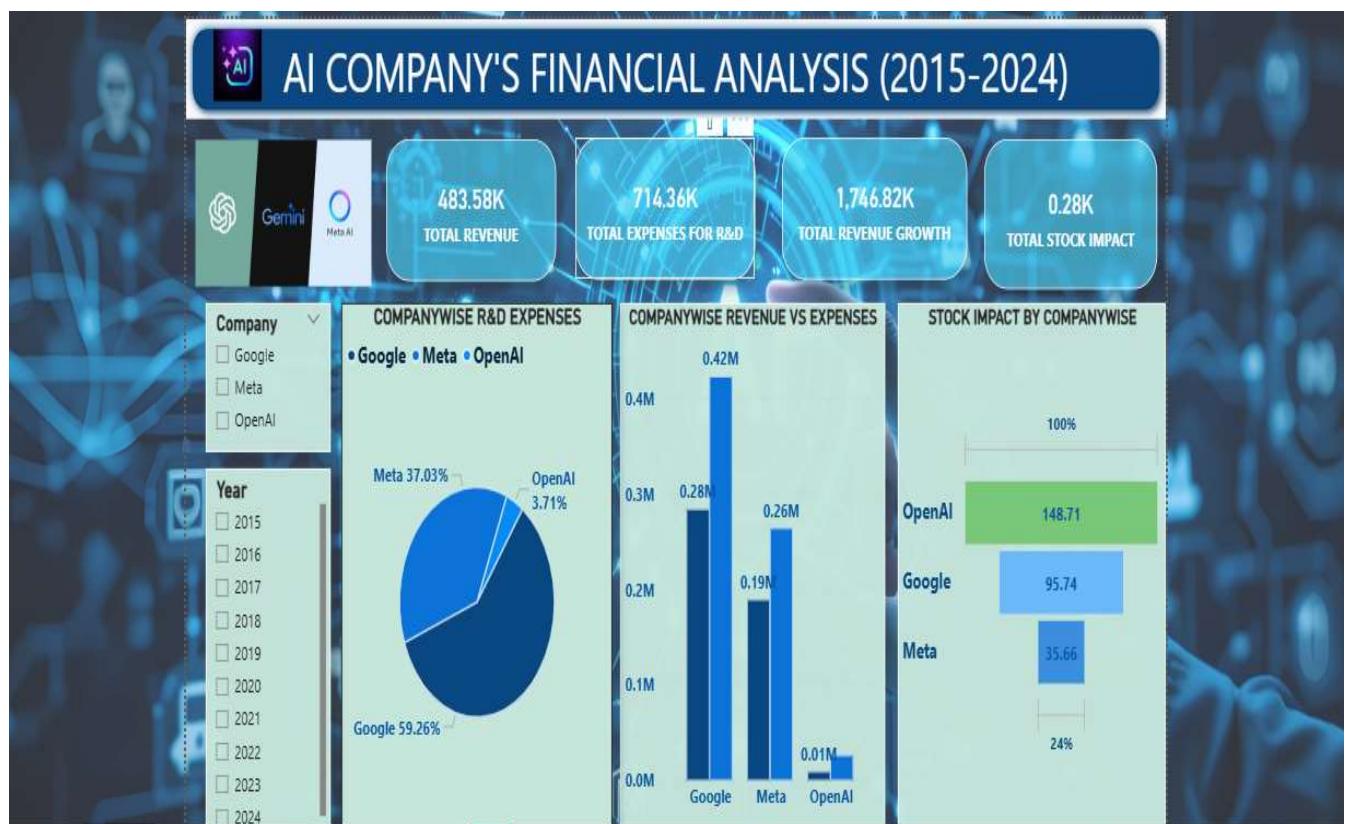
The screenshot shows a Power BI result grid titled "Result Grid". The data is presented in a table with columns: Company and AVG_Revenue_Growth. The table contains 3 rows of data, each representing a company's average revenue growth percentage. The data includes entries for Google, Meta, and OpenAI. The interface includes standard Power BI navigation elements like "Filter Rows", "Export", and "Wrap Cell Content". A sidebar on the right provides access to "Result Grid", "Form Editor", and "Field Types".

Company	AVG_Revenue_Growth
Google	159.6023131672597
Meta	159.5422283055026
OpenAI	159.04342732001058

VISUALIZATION USING POWER BI :-

VISUALS:

- ❖ KPI Cards — Total Revenue, Total Expenses, Total Stock Impact, Total Revenue Growth
- ❖ Pie chart — Total R&D Expenses by Company
- ❖ Clustered Column Chart — Company wise Revenue & Expenses by Company
- ❖ Funnel Chart — Stock Impact by Company wise
- ❖ Donut Chart — Total Revenue by Company wise
- ❖ Line chart — Revenue Growth Trend Yearly
- ❖ Waterfall chart — Profit Difference Analysis



CONCLUSION: -

The AI Financial Analysis project successfully provided meaningful insights into the financial performance, revenue trends, R&D investments, and market impact of leading AI companies. Through data cleaning, SQL queries, Python visualizations, and Power BI dashboards, we identified how AI revenue growth and R&D spending influence long-term profitability and stock performance.

- ❖ **OpenAI achieved the highest revenue growth in 2024**, reflecting strong innovation and fast market adoption.
- ❖ **Google continues to lead in total AI revenue and R&D expenses**, showing financial strength and long-term commitment.
- ❖ **OpenAI also shows a strong stock impact**, indicating high investor confidence and market influence.
- ❖ **Google's high expenses correlate with stable revenue growth**, proving the importance of continuous R&D investment.
- ❖ **Company-wise comparison shows clear performance gaps**, making it easy to identify leaders in growth, revenue, and spending.
- ❖ **Revenue vs. R&D visuals highlight how investment directly drives higher market performance.**
- ❖ Overall **OpenAI leads in growth momentum**, while **Google leads in financial scale and investment power**.