


## Healthcare Project - Important Questions

In this project, we aim to answer key questions related to various issues that will help improve our business and inform important decisions.

```
# Modules
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# load data
# Load data
path = "C:/Users/rvrei/Documents/Healthcare_df.csv"
healthcare_df = pd.read_csv(path)

# display all the columns
pd.set_option('display.max_columns',100)
healthcare_df.head(3)
```



	claim_id	patient_id	procedure_id	claim_date	claim_amount	claim_status	insurance_provider	procedure_type	pr
0	CLM0001	PAT0001	41	2024-03-29	1997.79	approved	Blue Shield	CT Scan	
1	CLM0001	PAT0001	41	2024-03-29	1997.79	approved	Blue Shield	CT Scan	
2	CLM0016	PAT0016	41	2023-09-16	1080.34	approved	Aetna	MRI	

### ▼ Project 1

Question: How are insurance claim reimbursements trending over time, and are delays or denials impacting cash flow?

Purpose: To evaluate the efficiency of the claims process, identify potential issues with payer contracts, and improve cash flow management. To track reimbursement trends, we can group claims by year and month.

```
# Convert 'claim_date' to datetime format
healthcare_df['claim_date'] = pd.to_datetime(healthcare_df['claim_date'])

# Extract year and month from claim_date
healthcare_df['claim_year'] = healthcare_df['claim_date'].dt.year
healthcare_df['claim_month'] = healthcare_df['claim_date'].dt.month

# Calculate total paid and denied claims over time
claims_trend_df = healthcare_df.groupby(['claim_year', 'claim_month']).agg(
    total_paid=('claim_amount', lambda x: sum(x[healthcare_df['claim_status'] == 'approved'])),
    total_denied=('claim_amount', lambda x: sum(x[healthcare_df['claim_status'] == 'denied'])),
    pending_claims=('claim_status', lambda x: (x == 'pending').sum())
).reset_index()

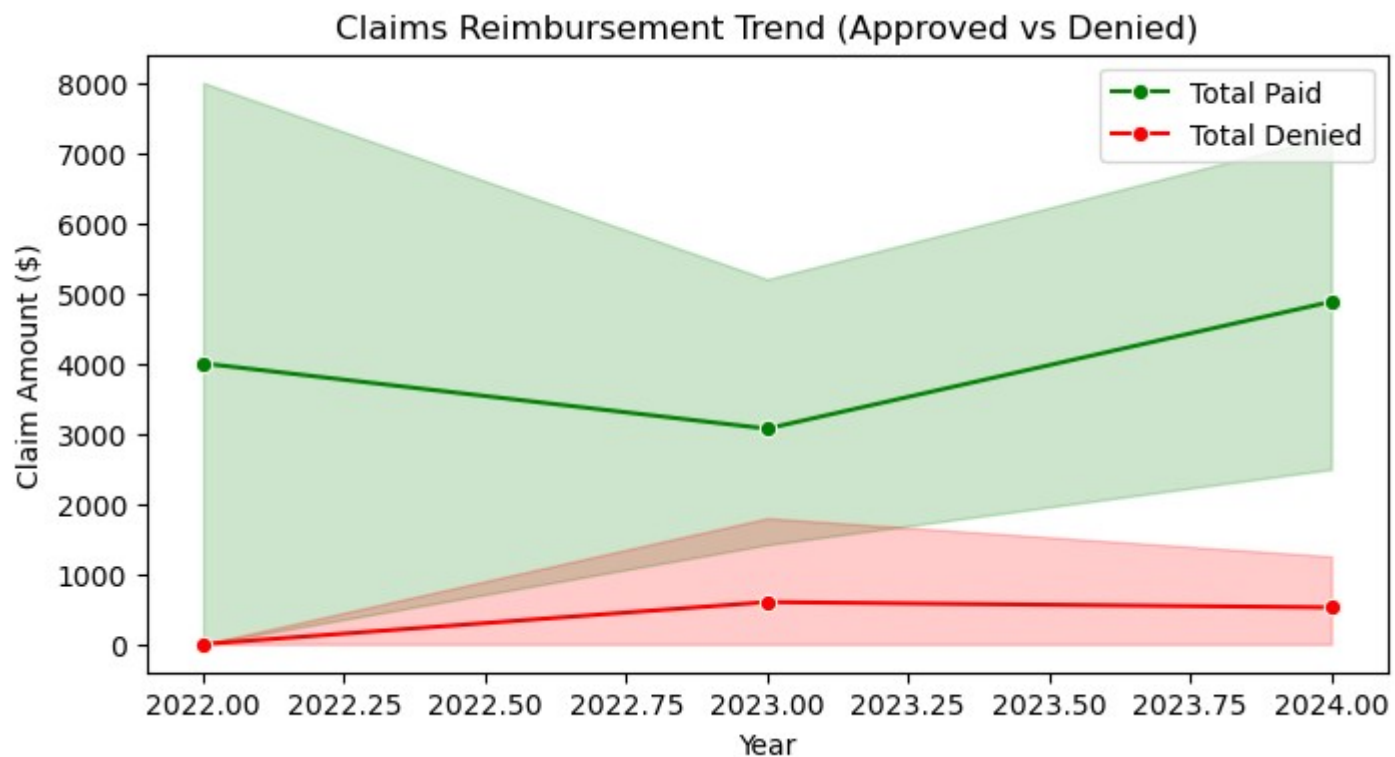
claims_trend_df.head()
```



	claim_year	claim_month	total_paid	total_denied	pending_claims
0	2022	11	8004.99	0.0	0
1	2022	12	0.00	0.0	1
2	2023	1	3598.75	0.0	0
3	2023	2	6147.12	0.0	0
4	2023	4	0.00	0.0	3

```
# Visualization: Plot trends for approved vs denied claims
plt.figure(figsize=(8, 4))
sns.lineplot(x='claim year', y='total paid', data=claims_trend_df, label='Total Paid', color='green', marker='o')
```

```
sns.lineplot(x='claim_year', y='total_denied', data=claims_trend_df, label='Total Denied', color='red', marker='o')
plt.title('Claims Reimbursement Trend (Approved vs Denied)')
plt.xlabel('Year')
plt.ylabel('Claim Amount ($)')
plt.legend()
plt.show()
```



We observe a declining trend in 2022, where the amount paid for approved claims decreased and denials increased. However, in 2023 this trend reversed, with an increase in the amount paid for approved claims and a decrease in denials.

This indicate that there is a shift in the approval and denial patterns for claims:

- 2022: Fewer claims are being approved. The increase in denials may reflect stricter approval processes or a higher rate of claims being rejected.

- 2023: The approval process may have become more lenient or efficient, leading to higher payouts for approved claims, while fewer claims are being denied.

This shift could be indicative of changes in policy, claims review processes, or overall business strategies aimed at improving claim approval rates and reducing denials.

## ▼ Project 2

Question: What is the return on investment (ROI) for various treatments, procedures, and medical technologies?

Purpose: To assess whether the organization's investment in specific treatments or technologies is generating adequate financial returns.

```
# Calculate ROI by dividing the revenue by the cost for each procedure
roi_df = healthcare_df.groupby('procedure_type').agg(
    total_revenue=('revenue', 'sum'),
    total_cost=('procedure_cost', 'sum')).reset_index()

roi_df['roi'] = roi_df['total_revenue'] / roi_df['total_cost']
roi_df = roi_df.sort_values(by='roi', ascending=False)
roi_df
```

	procedure_type	total_revenue	total_cost	roi
2	MRI	272910.28	151208.13	1.804865
4	X-Ray	111395.57	61889.94	1.799898
1	Lab Test	146980.83	83787.66	1.754206
0	CT Scan	55699.21	34156.26	1.630717
3	Physical Exam	73545.90	56477.52	1.302215

```
roi_df['total revenue ($)'] = roi_df['total_revenue'].apply(lambda x: f"${x:,.2f}")
roi_df['total cost ($)'] = roi_df['total_cost'].apply(lambda x: f"${x:,.2f}")
roi_df.drop(['total_revenue', 'total_cost'], axis=1)
```

	procedure_type	roi	total revenue (\$)	total cost (\$)
2	MRI	1.804865	\$272,910.28	\$151,208.13
4	X-Ray	1.799898	\$111,395.57	\$61,889.94
1	Lab Test	1.754206	\$146,980.83	\$83,787.66
0	CT Scan	1.630717	\$55,699.21	\$34,156.26
3	Physical Exam	1.302215	\$73,545.90	\$56,477.52

# Visualization: We can visualize the ROI of procedures.

```
plt.figure(figsize=(8, 4))
ax = sns.barplot(x='procedure_type', y='roi', data=roi_df, color='#9C27B0')
```

```
# Highest value to highlight
max_value = roi_df['roi'].max()
```

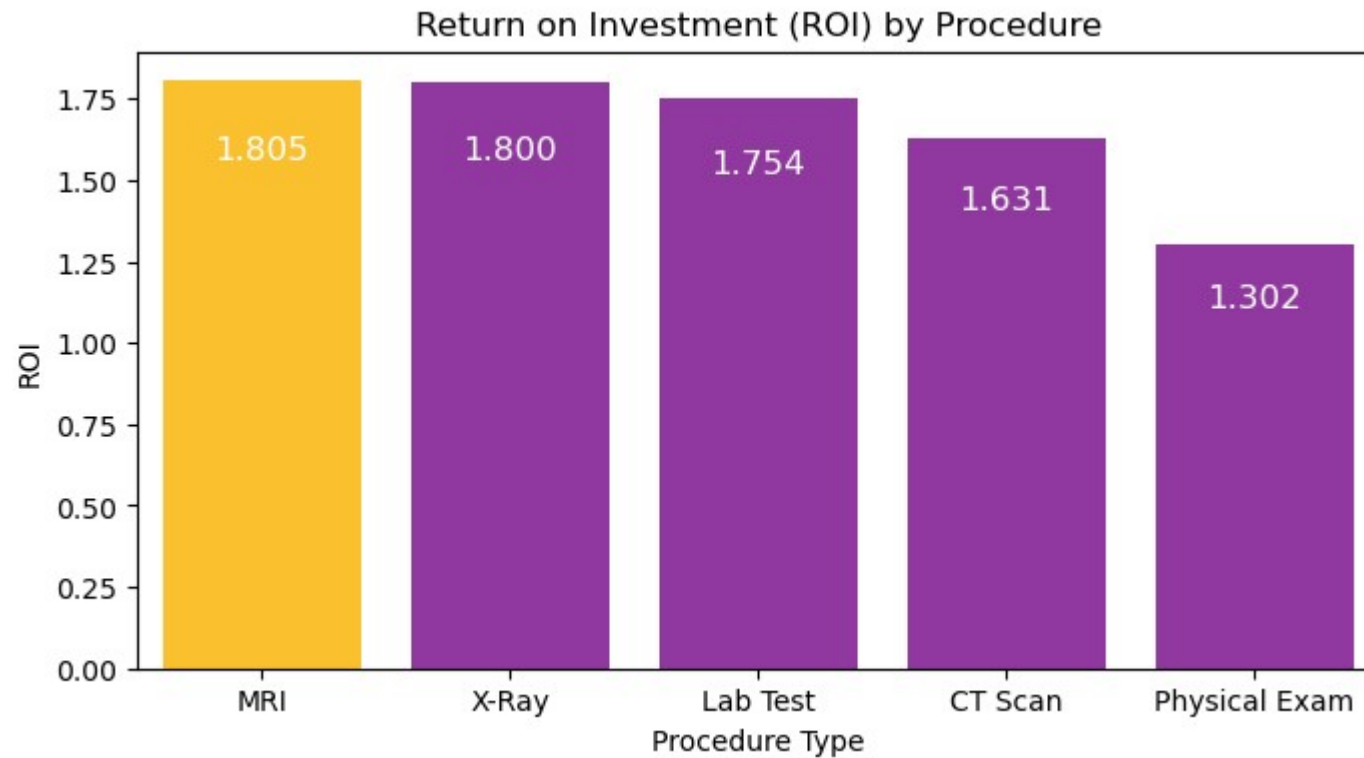
```
# Insert values inside each bar
for p in ax.patches:
    height = p.get_height() # Get the height of each bar (value)
```

```
percentage = f'{height:,.3f}' # Percentage values
```

```
if height == max_value: # Change the color of the highest bar
    p.set_facecolor('#FBC02D')
```

```
ax.text(p.get_x() + p.get_width() / 2, height/1.1, percentage, ha='center', va='top', color='white', fontsize=12)
```

```
plt.title('Return on Investment (ROI) by Procedure')  
plt.xlabel('Procedure Type')  
plt.ylabel('ROI')  
plt.show()
```



Based on the ROI calculation (revenue vs. cost), MRIs and X-rays are the most financially efficient treatments or procedures, meaning they generate the highest return relative to their costs. On the other hand, Physical Exams have the lowest ROI, suggesting that the revenue they generate does not justify their associated costs as effectively as MRIs or X-rays.

### ▼ Project 3

Question: What are the average patient acquisition costs, and how do they compare with the patient lifetime value?

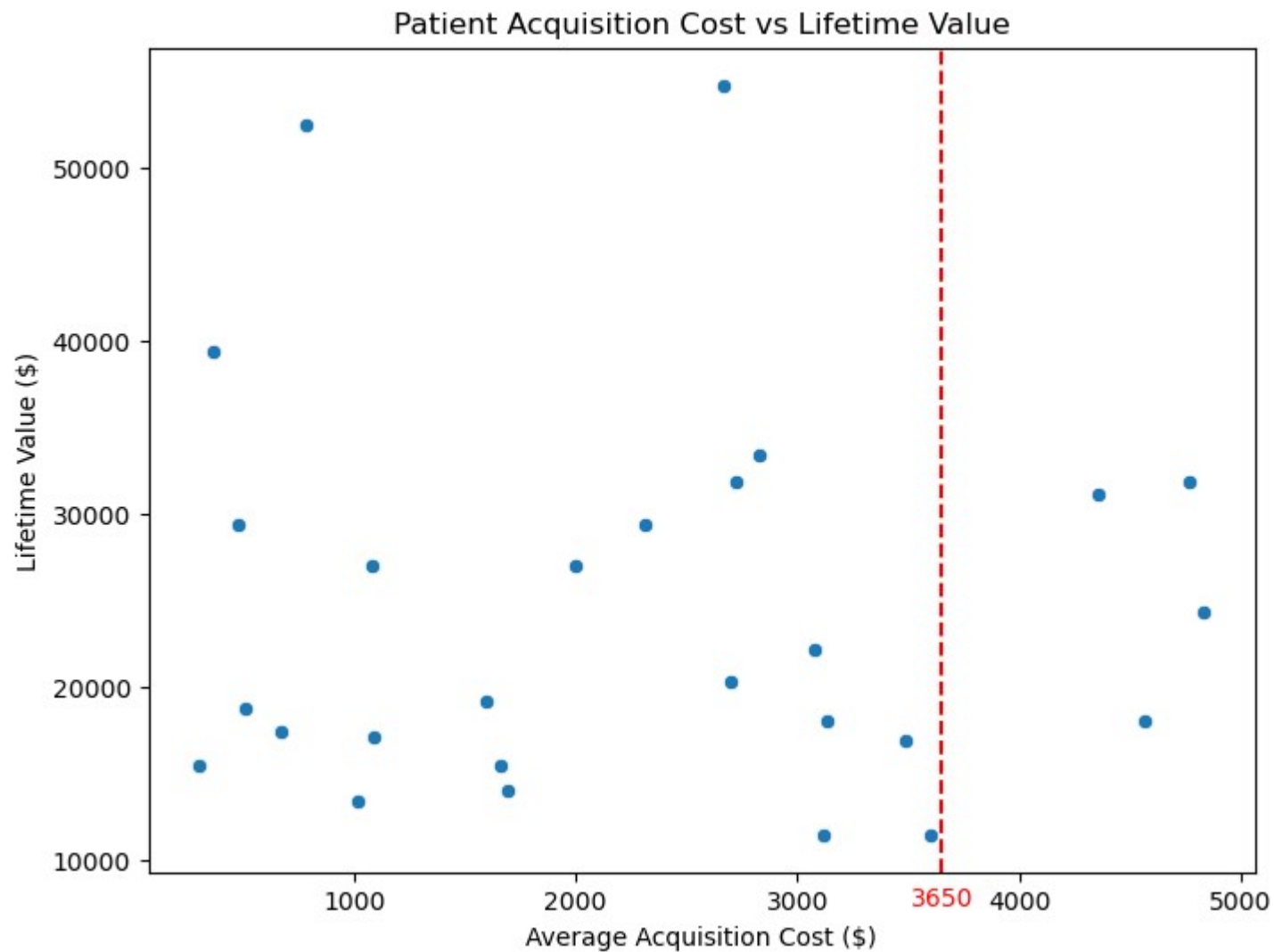
Purpose: To assess the balance between the costs of acquiring new patients and the long-term revenue each patient generates, helping to optimize marketing and operational strategies.

	patient_id	average acquisition cost (\$)	lifetime value (\$)
12	PAT0028	\$2,668.33	\$54,794.03
20	PAT0041	\$785.78	\$52,505.03
21	PAT0043	\$365.06	\$39,361.35

[illegible]

```
# Add horizontal line at a specific y-value (e.g., y=4000)
# plt.axhline(y=40000, color='blue', linestyle='--', linewidth=1.5)

plt.title('Patient Acquisition Cost vs Lifetime Value')
plt.xlabel('Average Acquisition Cost ($)')
plt.ylabel('Lifetime Value ($)')
plt.show()
```





With the average acquisition cost of \$3,650, we don't get an improvement lifetime value from our patients. The majority of the higher lifetime value patients acquisition cost is less than that amount.

## ▼ Project 4

Question: How much time are our doctors and medical staff spending on direct patient care compared to administrative tasks, and how can we optimize labor efficiency?

Purpose: To evaluate workforce productivity, identify inefficiencies, and ensure that medical staff spend the maximum amount of time on revenue-generating activities.

```
healthcare_df.head(2)
```

	claim_id	patient_id	procedure_id	claim_date	claim_amount	claim_status	insurance_provider	procedure_type	pr
0	CLM0001	PAT0001	41	2024-03-29	1997.79	approved	Blue Shield	CT Scan	
1	CLM0001	PAT0001	41	2024-03-29	1997.79	approved	Blue Shield	CT Scan	

```
# Calculate the care-to-administration ratio for each doctor
healthcare_df['care_to_admin_ratio'] = healthcare_df['time_spent_on_patient_care'] / healthcare_df['time_spent_on_admini

# Group by doctor to calculate the total care-to-administration ratio
doctor_efficiency_df = healthcare_df.groupby('doctor_id').agg(
    total_care_time=('time_spent_on_patient_care', 'sum'),
    total_admin_time=('time_spent_on_administration', 'sum'),
    care_to_admin_ratio=('care_to_admin_ratio', 'mean')
).reset_index()

# The result
```

```

doctor_efficiency_df = doctor_efficiency_df.sort_values(by='care_to_admin_ratio', ascending=False)
doctor_efficiency_df['doctor_id'] = doctor_efficiency_df['doctor_id'].apply(lambda x: f"{x:03d}")
doctor_efficiency_df.head()

```

	doctor_id	total_care_time	total_admin_time	care_to_admin_ratio
0	001	155	21	7.380952
5	006	741	249	3.699014
8	009	444	175	3.249807
1	002	191	65	2.995238
9	010	148	51	2.901961

```

# Visualization: Plotting care-to-admin ratio by doctor
plt.figure(figsize=(8, 4))
ax = sns.barplot(x='doctor_id', y='care_to_admin_ratio', data=doctor_efficiency_df, color='#009688')

# Highest value to highlight
max_value = doctor_efficiency_df['care_to_admin_ratio'].max()

# Insert values inside each bar
for p in ax.patches:
    height = p.get_height() # Get the height of each bar (value)

    percentage = f'{height:,.2f}' # Percentage values

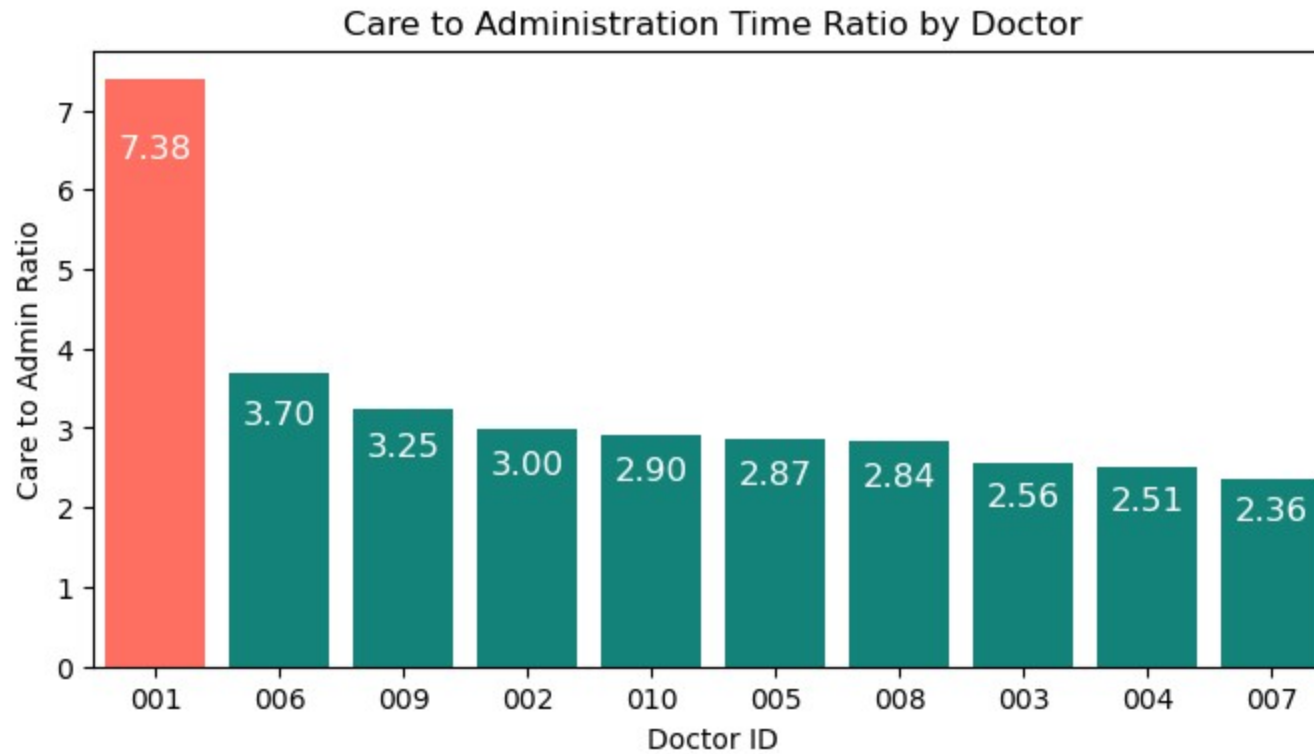
    if height == max_value: # Change the color of the highest bar
        p.set_facecolor('#FF6F61')

    ax.text(p.get_x() + p.get_width() / 2, height/1.1, percentage, ha='center', va='top', color='white', fontsize=12)

plt.title('Care to Administration Time Ratio by Doctor')
plt.xlabel('Doctor ID')
plt.ylabel('Care to Admin Ratio')
plt.show()

```

```
plt.show()
```



## ▼ Project 5

Question: What are the main insurance provider with denied claims and how many are they?

Purpose: To identify the insurance provider with the more amount in dollars for denial claims.

```
# Analyze claim denials
denied_claims_df = healthcare_df[healthcare_df['claim_status'] == 'denied']
denial_reasons_df = denied_claims_df.groupby('insurance_provider').agg(
    total_claims=('claim_amount', 'count'),
    total_value=('claim_amount', 'sum')).reset_index()
```

```
# Display the top denial reasons
denial_reasons_df.sort_values(by='total_value', ascending=False)
```

	insurance_provider	total_claims	total_value
1	Blue Shield	2	5392.66
0	Aetna	4	3096.76
2	Cigna	1	1087.30

```
# Total amount of denial claims
print(f"Total denial claims: ${denial_reasons_df['total_value'].sum():,.2f}")
```

Total denial claims: \$9,576.72

```
# Create a Seaborn barplot
sns.set(style='whitegrid')
ax = sns.barplot(x='insurance_provider', y='total_value', data=denial_reasons_df, color='#ADD8E6')
```

```
# Highest value to highlight
max_value = denial_reasons_df['total_value'].max()
```

```
# Insert values inside each bar
for p in ax.patches:
    height = p.get_height() # Get the height of each bar (value)
```

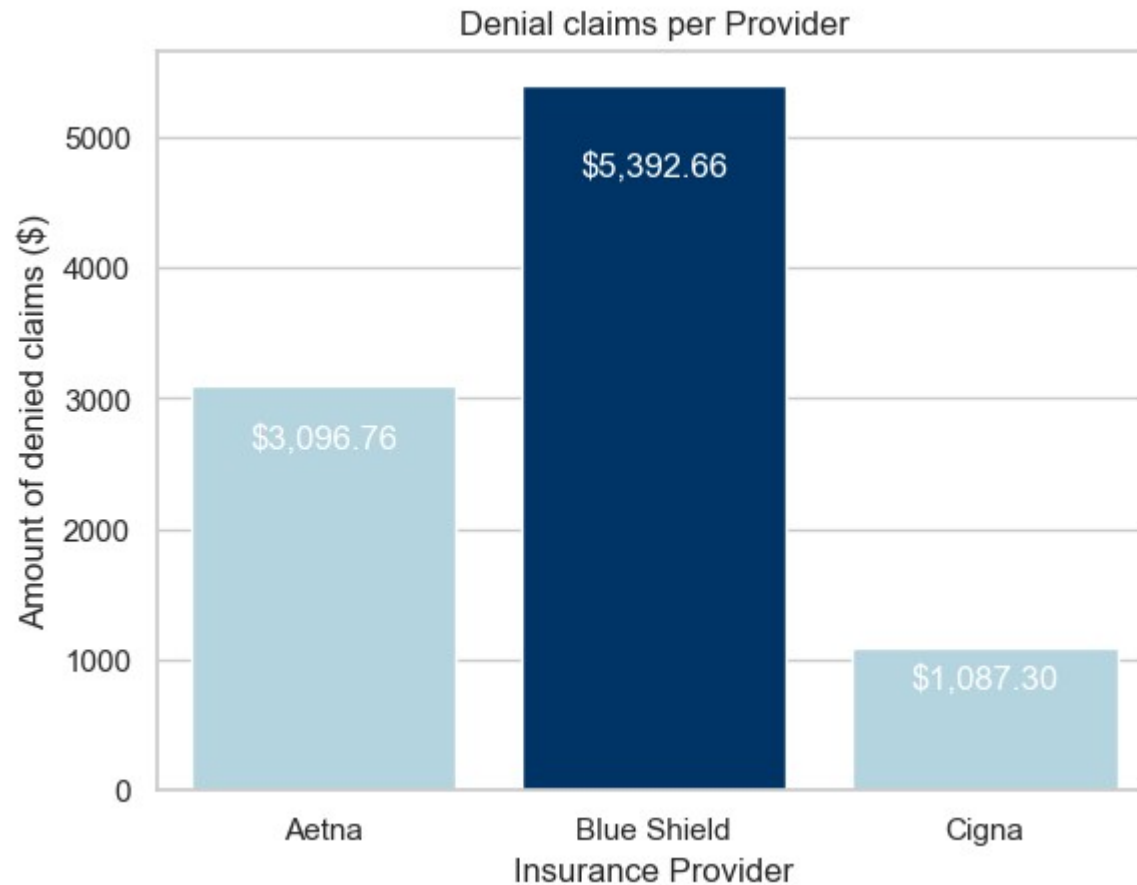
```
    percentage = f'${height:,.2f}' # Percentage values
```

```
    if height == max_value: # Change the color of the highest bar
        p.set_facecolor('#003366')
```

```
    ax.text(p.get_x() + p.get_width() / 2, height/1.1, percentage, ha='center', va='top', color='white', fontsize=12)
```

```
plt.title('Denial claims per Provider')
```

```
plt.xlabel('Insurance Provider')
plt.ylabel('Amount of denied claims ($)')
plt.show()
```



## ▼ Project 6

Question: Which patient demographics (such as age, location, insurance type, etc.) are driving the highest costs and revenues for the organization?

Purpose: To identify which patient populations are the most profitable or costly, and to adjust services, marketing, and care

management strategies accordingly.

```
# Calculate costs and revenues by patient demographics
healthcare_df['profit'] = healthcare_df['revenue'] - healthcare_df['procedure_cost']

demographics_summary_df = healthcare_df.groupby(['age', 'gender', 'insurance_provider']).agg(
    total_revenue=('revenue', 'sum'),
    total_cost=('procedure_cost', 'sum'),
    profit=('profit', 'sum')).reset_index()

# Display the result
demographics_summary_df.head()
```

	age	gender	insurance_provider	total_revenue	total_cost	profit
0	19	F	Aetna	26962.32	16795.34	10166.98
1	19	M	Cigna	54794.03	29288.82	25505.21
2	25	M	Blue Shield	22142.86	13150.24	8992.62
3	26	F	Aetna	29335.62	31895.12	-2559.50
4	26	F	Blue Shield	14006.17	4452.66	9553.51

```
demographics_summary_df.sort_values(by='profit', ascending=False).head()
```

	age	gender	insurance_provider	total_revenue	total_cost	profit
20	60	M	Blue Shield	52505.03	23970.32	28534.71
1	19	M	Cigna	54794.03	29288.82	25505.21
24	73	F	Cigna	39361.35	15041.12	24320.23
22	70	F	Cigna	33363.62	18063.77	15299.85
5	26	F	Blue Shield	14006.17	4452.66	9553.51

```
6    29    M    Blue Shield    31850.60    17143.05    14707.55
```

```
# Create ages groups:
demographics_summary_df['age_strata'] = pd.cut(x=demographics_summary_df['age'],\
        bins=[18,29,45,59,np.inf], labels=['19-29','30-45','46-59','+60'])
demographics_summary_df.head()
```

	age	gender	insurance_provider	total_revenue	total_cost	profit	age_strata
0	19	F	Aetna	26962.32	16795.34	10166.98	19-29
1	19	M	Cigna	54794.03	29288.82	25505.21	19-29
2	25	M	Blue Shield	22142.86	13150.24	8992.62	19-29
3	26	F	Aetna	29335.62	31895.12	-2559.50	19-29
4	26	F	Blue Shield	14006.17	4452.66	9553.51	19-29

```
demo_strata = demographics_summary_df.groupby('age_strata').sum()
demo_strata
```

	age	total_revenue	total_cost	profit
age_strata				
19-29	171	194500.85	124202.26	70298.59
30-45	252	134298.60	69108.70	65189.90
46-59	323	125816.96	86093.41	39723.55
+60	484	205915.38	108115.14	97800.24

```
# Visualization: Plotting care-to-admin ratio by doctor
```

```
plt.figure(figsize=(6, 4))

# Create a Seaborn barplot
sns.set(style='whitegrid')
ax = sns.barplot(x=demo_strata.index, y='profit', data=demo_strata, color='#4CAF50', errorbar=None)

# Highest value to highlight
max_value = demo_strata['profit'].max()

# Insert values inside each bar
for p in ax.patches:
    height = p.get_height() # Get the height of each bar (value)

    percentage = f'${height:,.2f}' # Percentage values

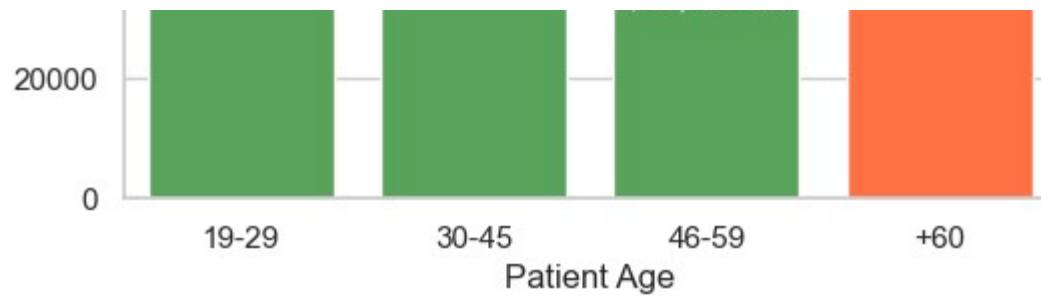
    if height == max_value: # Change the color of the highest bar
        p.set_facecolor('#FF7043')

    ax.text(p.get_x() + p.get_width() / 2, height/1.1, percentage, ha='center', va='top', color='white', fontsize=12)

plt.title('Profit vs Age')
plt.xlabel('Patient Age')
plt.ylabel('Profit ($)')
plt.show()
```







With the results of this study we see that the group of patients with age 60 and over are the ones that give more profit following by the younger group, from 19 to 29 years old.

## ▼ Project 7

Question: How do our financial performance metrics (e.g., profit margin, cost per procedure, payer mix) compare to industry benchmarks and competitors?

Purpose: To evaluate the organization's performance relative to the market, identifying areas for improvement and potential competitive advantages.

```
# Calculate profit margin and cost per procedure
performance_df = healthcare_df.groupby('procedure_type').agg(
    total_revenue=('revenue', 'sum'),
    total_cost=('procedure_cost', 'sum'))

performance_df['profit_margin (%)'] = round(((performance_df['total_revenue'] - performance_df['total_cost']) / performance_df['total_revenue']) * 100, 2)
performance_df['cost_per_procedure'] = performance_df['total_cost'] / healthcare_df['procedure_id'].nunique()

# Display the result
performance_df.sort_values(by='profit_margin (%)', ascending=False)
```

total_revenue	total_cost	profit_margin (%)	cost_per_procedure
---------------	------------	-------------------	--------------------

procedure_type				
<b>MRI</b>	272910.28	151208.13	44.59	6873.096818
<b>X-Ray</b>	111395.57	61889.94	44.44	2813.179091
<b>Lab Test</b>	146980.83	83787.66	42.99	3808.530000
<b>CT Scan</b>	55699.21	34156.26	38.68	1552.557273
<b>Physical Exam</b>	73545.90	56477.52	23.21	2567.160000

```
# Visual of profit margin vs procedure type
# Visualization: Plotting care-to-admin ratio by doctor
plt.figure(figsize=(6, 4))

# Create a Seaborn barplot
sns.set(style='whitegrid')
ax = sns.barplot(x=performance_df.index, y='profit_margin (%)', data=performance_df, color='#B0BEC5', errorbar=None)

# Highest value to highlight
max_value = performance_df['profit_margin (%)'].max()

# Insert values inside each bar
for p in ax.patches:
    height = p.get_height() # Get the height of each bar (value)

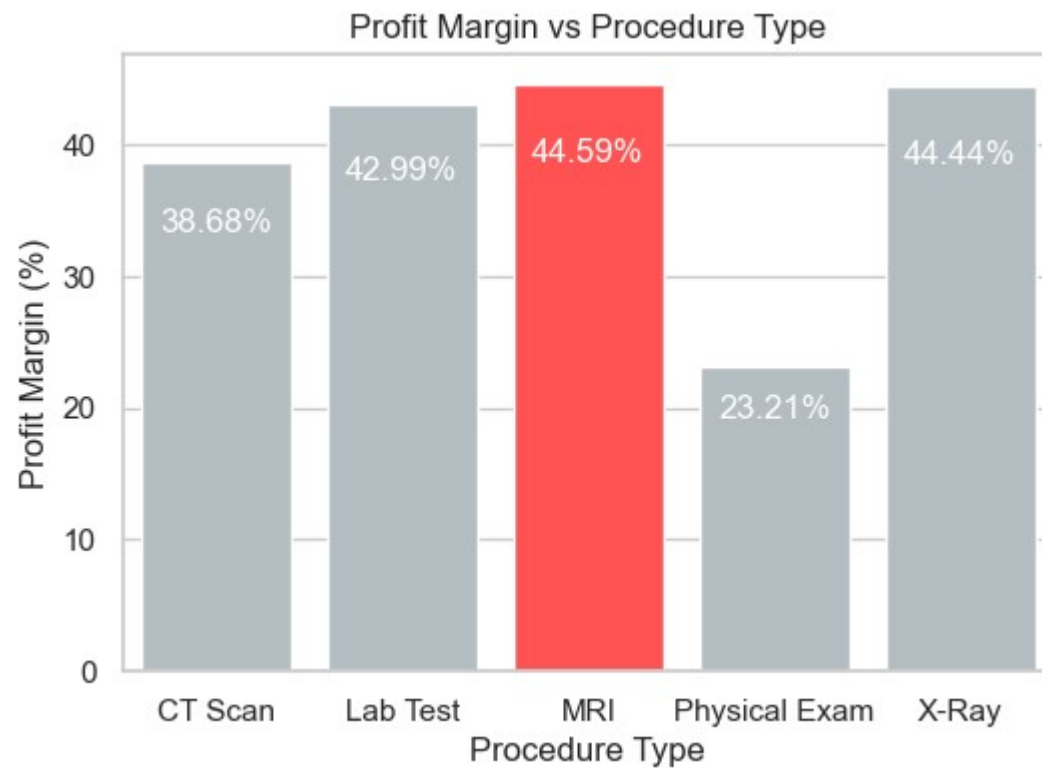
    percentage = f'{height:.2f}%' # Percentage values

    if height == max_value: # Change the color of the highest bar
        p.set_facecolor('#FF5252')

    ax.text(p.get_x() + p.get_width() / 2, height/1.1, percentage, ha='center', va='top', color='white', fontsize=12)

plt.title('Profit Margin vs Procedure Type')
plt.xlabel('Procedure Type')
plt.ylabel('Profit Margin (%)')
```

```
plt.show()
```



## ▼ Project 8

Question: What are the trends in patient readmission rates across different medical procedures?

Purpose: To identify which medical procedures are associated with higher readmission rates within 30 days of discharge. By evaluating the relationship between procedure types and patient readmission, the study aims to understand the impact on hospital operations, including claims processing, revenue generation, and resource availability.

```
# Patients that have been readmitted
```

```
patient_readmission = healthcare_d[healthcare_d['readmission']=="Yes"]
patient_readmission.head(3)
```

	claim_id	patient_id	procedure_id	claim_date	claim_amount	claim_status	insurance_provider	procedure_type	pr
0	CLM0001	PAT0001	41	2024-03-29	1997.79	approved	Blue Shield	CT Scan	
1	CLM0001	PAT0001	41	2024-03-29	1997.79	approved	Blue Shield	CT Scan	
2	CLM0016	PAT0016	41	2023-09-16	1080.34	approved	Aetna	MRI	

```
# Amount of precision per readmitted patient
patient_readmission_rate = patient_readmission.groupby('procedure_type').count()
patient_readmission_rate['readmission_rate'] = round(patient_readmission.groupby('procedure_type').count()[['claim_id']])
```

```
# Plotting readmission rates by procedure
plt.figure(figsize=(6, 4))
```

```
# Create a basic Seaborn barplot
sns.set(style="whitegrid")
ax = sns.barplot(x=patient_readmission_rate.index, y='readmission_rate', data=patient_readmission_rate, \
                 width=0.7, color='#4682B4')
```

```
# Find the highest value to highlight
max_value = patient_readmission_rate['readmission_rate'].max()
```

```
# Manually adjust the x-tick positions and bar alignment
ax.set_xticks(range(len(patient_readmission_rate))) # Align x-ticks at specific positions
ax.set_xticklabels(patient_readmission_rate.index, rotation=0) # Optional: adjust rotation of labels
```

```
# Adjust the alignment of bars with the labels
for tick in ax.get_xticklabels():
    tick.set_horizontalalignment('center') # Ensure that tick labels are centered
```

```
# Insert the values inside each bar
for p in ax.patches:
```

```
# Get the height of each bar (the value)
height = p.get_height()

# Convert the value to percentage and format it (e.g., 25% for 0.25)
# percentage = f'{height * 100:.0f}%' # Multiply by 100 and format as a string
percentage = f'{height:.1f}%' # simply one decimal digit and the percentage symbol

# If the bar is the highest, change its color to 'green'
if height == max_value:
    p.set_facecolor('#FF6347') # Change the color of the highest bar

# Annotate the bar with the value inside
ax.text(p.get_x() + p.get_width() / 2, height / 1.1, percentage,
        ha='center', va='top', color='white', fontsize=12)

plt.title('Readmission Rates by Procedure')
plt.xlabel('Procedure Type')
plt.ylabel('Readmission Rate (%)')

plt.show()
```



