In [ ]:
```
# Titanic_EDA
```

In [1]:
```
# imports
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# display settings
pd.set_option('display.max_columns', None)
sns.set(style='whitegrid')
```

In [10]:
```
# load data
df = pd.read_csv("C:/Users/akans/Downloads/train.csv")
```

In [11]:
```
# quick look
df.head()
df.info()
df.describe(include='all')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Out[11]:

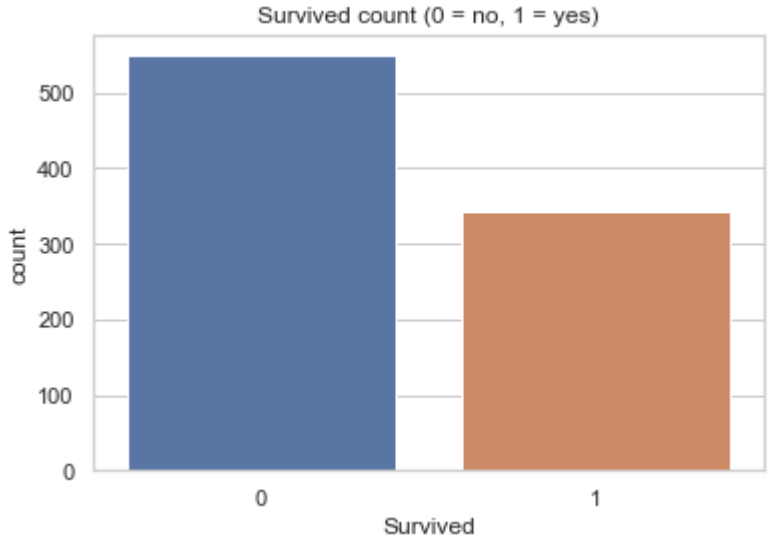
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch
count	891.000000	891.000000	891.000000	891	891	714.000000	891.000000	891.000000
unique	NaN	NaN	NaN	891	2	NaN	NaN	NaN
top	NaN	NaN	NaN	Braund, Mr. Owen Harris	male	NaN	NaN	NaN
freq	NaN	NaN	NaN	1	577	NaN	NaN	NaN
mean	446.000000	0.383838	2.308642	NaN	NaN	29.699118	0.523008	0.381594
std	257.353842	0.486592	0.836071	NaN	NaN	14.526497	1.102743	0.806057
min	1.000000	0.000000	1.000000	NaN	NaN	0.420000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	NaN	NaN	20.125000	0.000000	0.000000
50%	446.000000	0.000000	3.000000	NaN	NaN	28.000000	0.000000	0.000000
75%	668.500000	1.000000	3.000000	NaN	NaN	38.000000	1.000000	0.000000
max	891.000000	1.000000	3.000000	NaN	NaN	80.000000	8.000000	6.000000

In [12]:
```
# missing values summary
missing = df.isnull().sum().sort_values(ascending=False)
missing_pct = (missing / len(df) * 100).round(2)
pd.DataFrame({'missing_count': missing, 'missing_pct': missing_pct})
```

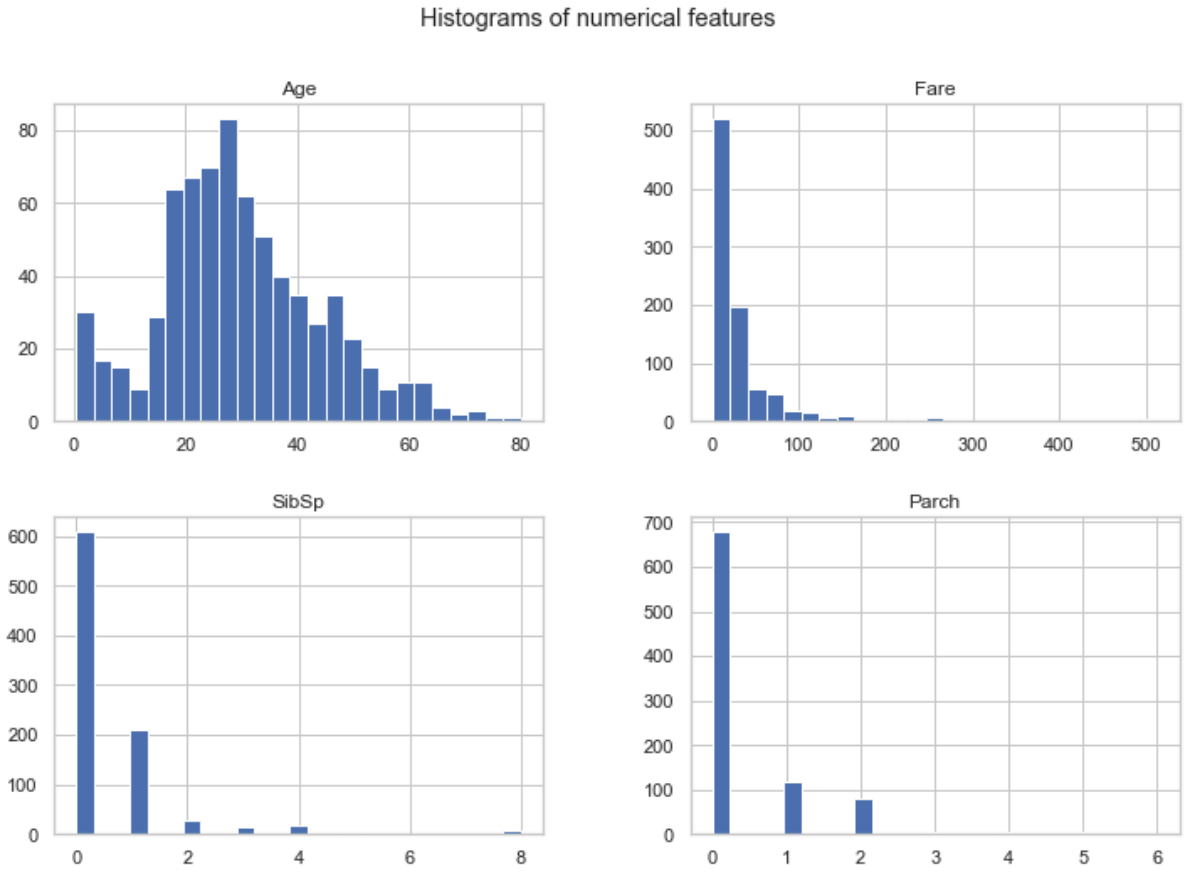
Out[12]:

	missing_count	missing_pct
Cabin	687	77.10
Age	177	19.87
Embarked	2	0.22
PassengerId	0	0.00
Survived	0	0.00
Pclass	0	0.00
Name	0	0.00
Sex	0	0.00
SibSp	0	0.00
Parch	0	0.00
Ticket	0	0.00
Fare	0	0.00

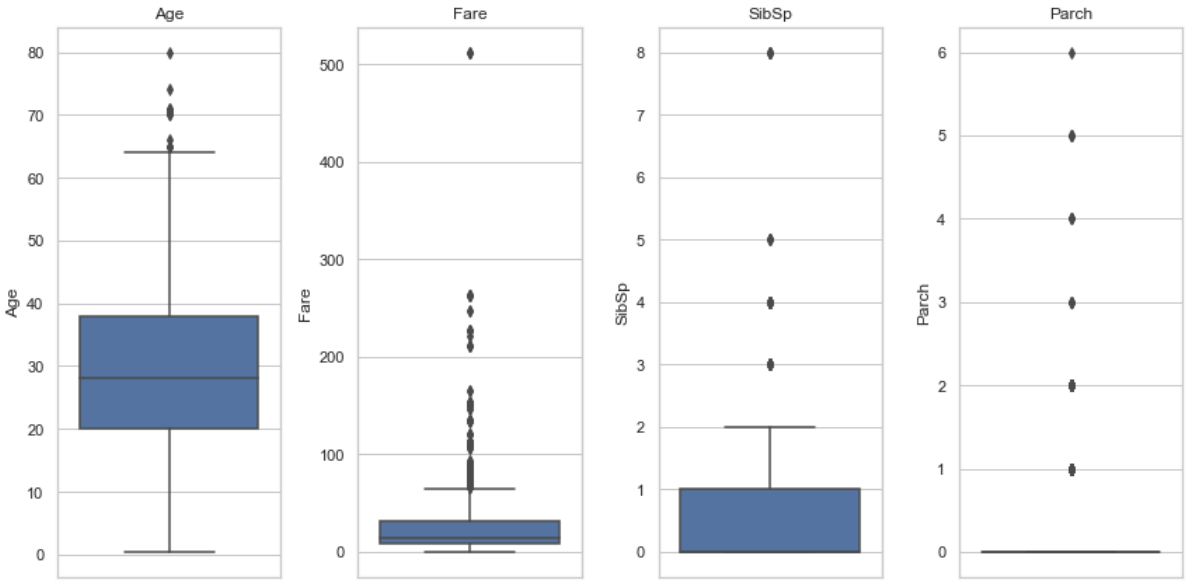
In [13]:
```
# target distribution
plt.figure(figsize=(6,4))
sns.countplot(x='Survived', data=df)
plt.title('Survived count (0 = no, 1 = yes)')
plt.show()
```



In [14]:
```
# Univariate analysis (numerical)
num_cols = ['Age', 'Fare', 'SibSp', 'Parch']
df[num_cols].hist(bins=25, figsize=(12,8))
plt.suptitle('Histograms of numerical features')
plt.show()
```



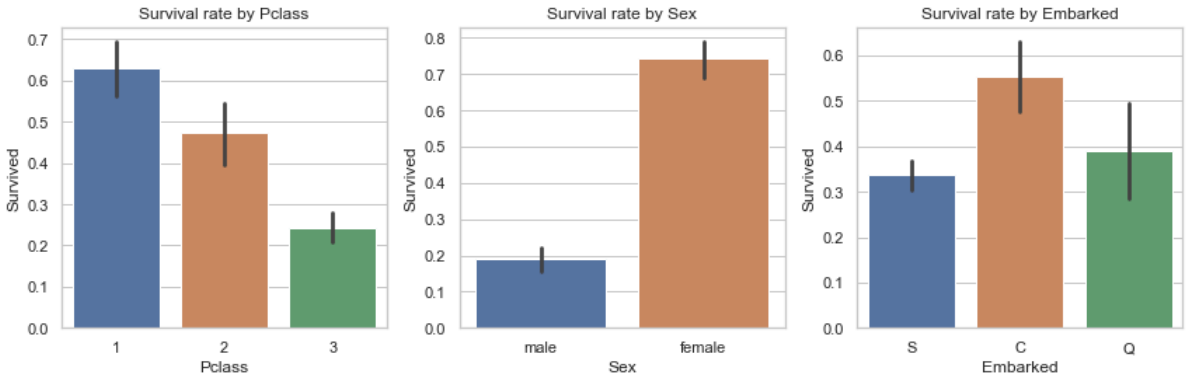
In [15]:
```
# boxplots to check outliers
plt.figure(figsize=(12,6))
for i, col in enumerate(num_cols, 1):
    plt.subplot(1,4,i)
    sns.boxplot(y=df[col])
    plt.title(col)
plt.tight_layout()
plt.show()
```



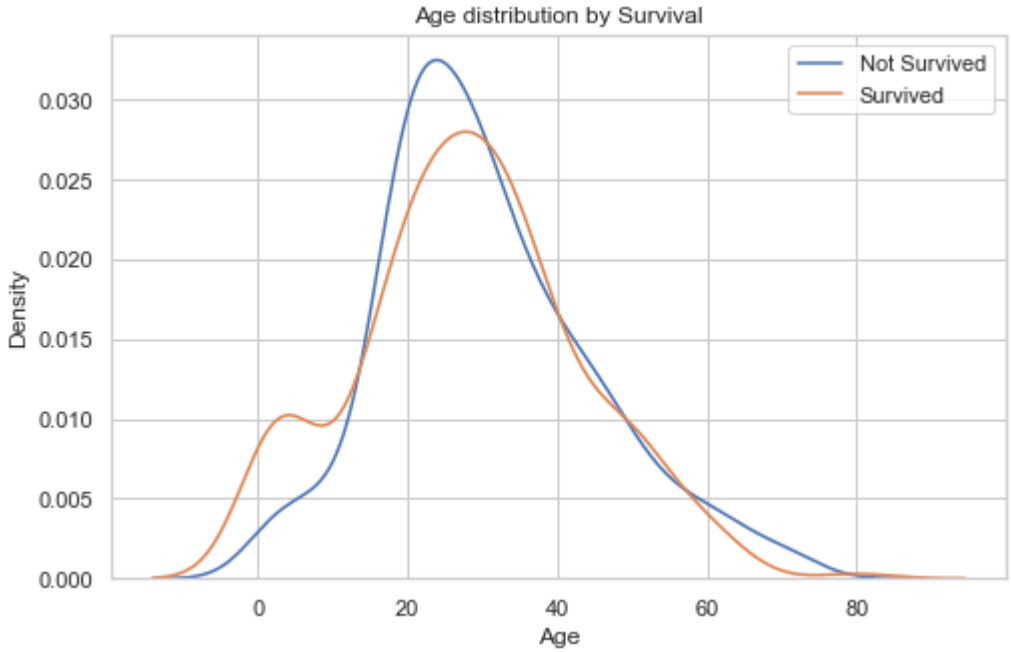
In [16]:
```
# categorical value counts (example)
cat_cols = ['Pclass', 'Sex', 'Embarked']
for col in cat_cols:
    print(f"\n{col} value counts:")
    display(df[col].value_counts())
```

```
Pclass value counts:
3    491
1    216
2    184
Name: Pclass, dtype: int64
Sex value counts:
male      577
female    314
Name: Sex, dtype: int64
Embarked value counts:
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

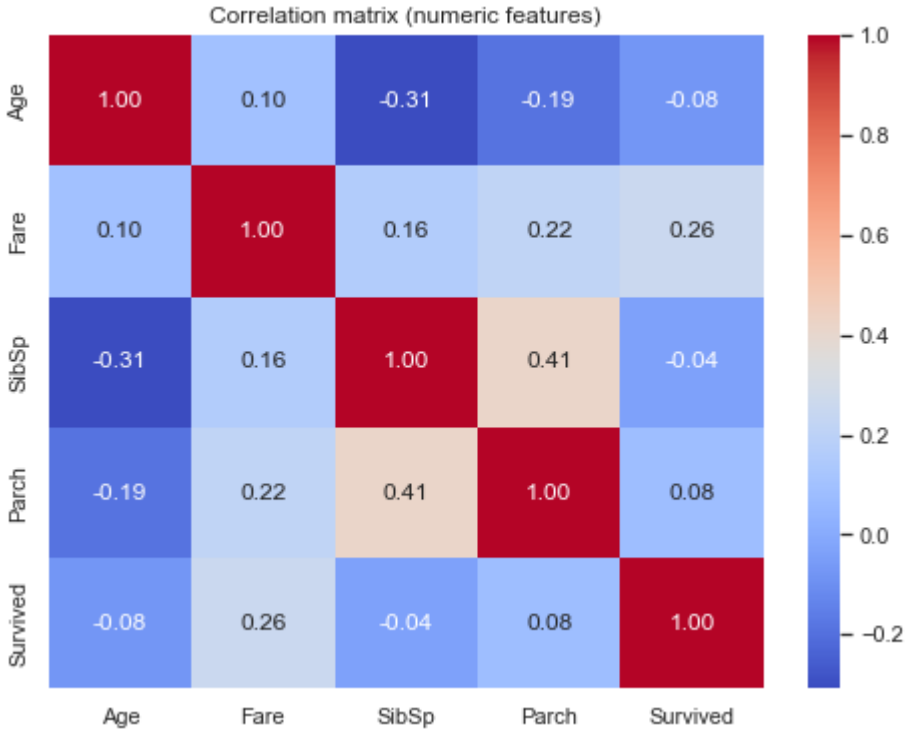
In [19]:
```
# survival rate by category (bivariate)
plt.figure(figsize=(12,4))
plt.subplot(1,3,1)
sns.barplot(x='Pclass', y='Survived', data=df)
plt.title('Survival rate by Pclass')

plt.subplot(1,3,2)
sns.barplot(x='Sex', y='Survived', data=df)
plt.title('Survival rate by Sex')

plt.subplot(1,3,3)
sns.barplot(x='Embarked', y='Survived', data=df)
plt.title('Survival rate by Embarked')
plt.tight_layout()
plt.show()
```



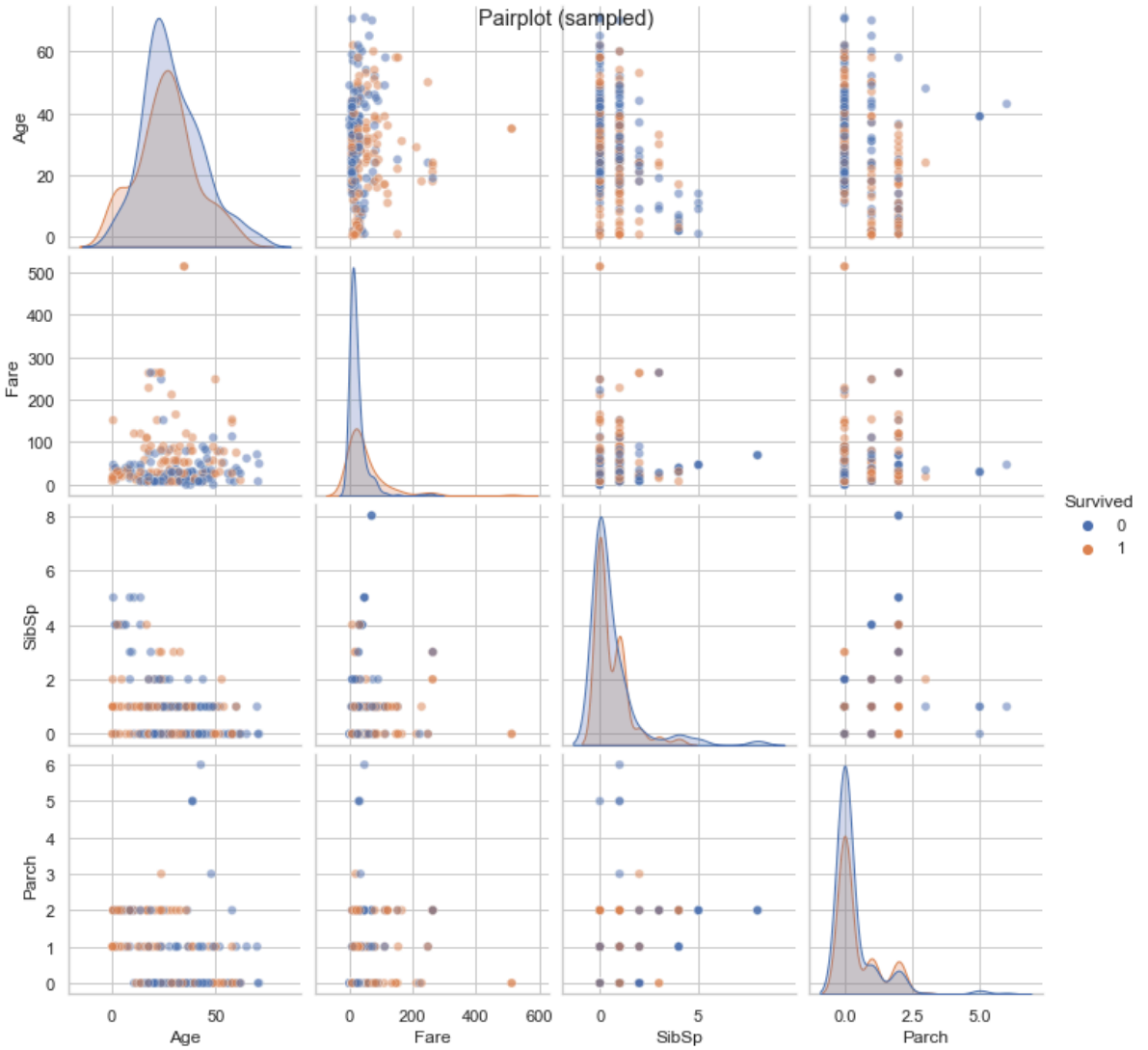
In [20]:
```
# Age distribution by survival
plt.figure(figsize=(8,5))
sns.kdeplot(df.loc[df.Survived==0,'Age'].dropna(), label='Not Survived')
sns.kdeplot(df.loc[df.Survived==1,'Age'].dropna(), label='Survived')
plt.title('Age distribution by Survival')
plt.legend()
plt.show()
```



In [21]:
```
# correlation matrix (numerical)
plt.figure(figsize=(8,6))
sns.heatmap(df[num_cols + ['Survived']].corr(), annot=True, fmt=".2f", cmap='coolwa
plt.title('Correlation matrix (numeric features)')
plt.show()
```



In [22]:
```
# pairplot
sample = df.sample(400, random_state=1)
sns.pairplot(sample[['Age','Fare','SibSp','Parch','Survived']], hue='Survived', dia
plt.suptitle('Pairplot (sampled)')
plt.show()
```



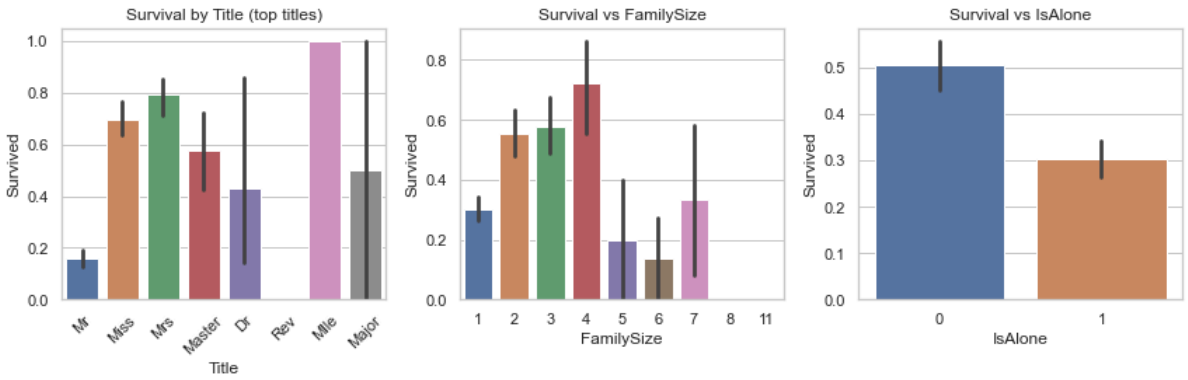
In [23]:
```
# Feature engineering examples
# Title extraction from Name
df['Title'] = df['Name'].str.extract(r',\s*([^\.]+)\.', expand=False)
df['FamilySize'] = df['SibSp'] + df['Parch'] + 1
df['IsAlone'] = (df['FamilySize'] == 1).astype(int)
```

In [24]:
```
# Show counts
df[['Title','FamilySize','IsAlone']].head()
```

Out[24]:

	Title	FamilySize	IsAlone
0	Mr	2	0
1	Mrs	2	0
2	Miss	1	1
3	Mrs	2	0
4	Mr	1	1

In [25]:
```
# survival by engineered features
plt.figure(figsize=(12,4))
plt.subplot(1,3,1)
sns.barplot(x='Title', y='Survived', data=df, order=df['Title'].value_counts().inde
plt.xticks(rotation=45)
plt.title('Survival by Title (top titles)')

plt.subplot(1,3,2)
sns.barplot(x='FamilySize', y='Survived', data=df)
plt.title('Survival vs FamilySize')

plt.subplot(1,3,3)
sns.barplot(x='IsAlone', y='Survived', data=df)
plt.title('Survival vs IsAlone')
plt.tight_layout()
plt.show()
```



In [26]:
```
# missing data handling suggestions (Age, Embarked, Cabin)
# show rows with missing Age or Embarked
df[df['Age'].isnull()].head()
df[df['Embarked'].isnull()]
```

Out[26]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Eml
61	62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	B28	
829	830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	B28	

In [27]:
```
# Example imputation: fill Embarked with mode, Age with median by Title
df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)
df['Age'] = df.groupby('Title')['Age'].apply(lambda x: x.fillna(x.median()))
```

In [28]:
```
# final summary table: survival rates by Pclass/Sex/Title
summary = df.groupby(['Pclass','Sex'])['Survived'].agg(['count','mean']).reset_inde
summary.sort_values('survival_rate', ascending=False).head(10)
```

Out[28]:

	Pclass	Sex	count	survival_rate
0	1	female	94	0.968085
2	2	female	76	0.921053
4	3	female	144	0.500000
1	1	male	122	0.368852
3	2	male	108	0.157407
5	3	male	347	0.135447