In [6]:

```
df = pd.read_excel('Insurance Policies.xlsx', sheet_name= 'Clean Insurance Policies')
```

In [ ]:

We answer some questions of the data to gain more insights

1. What are the average claim frequencies and amounts for different demographic groups (e.g., gender, marital status, education etc)?

In [7]:

```
#Group by 'gender'
avg_claims_by_gender = df.groupby('gender').agg(
    avg_claim_freq=('claim_freq', 'mean'),
    avg_claim_amt=('claim_amt', 'mean')
).reset_index()

print('Average Claims by Gender: ')
print(avg_claims_by_gender)


# Group by 'marital_status'
avg_claims_by_marital_status = df.groupby('marital_status').agg(
    avg_claim_freq=('claim_freq', 'mean'),
    avg_claim_amt=('claim_amt', 'mean')
).reset_index()

print("Average Claims by Marital Status: ")
print(avg_claims_by_marital_status)


# Group by 'education'
avg_claims_by_education = df.groupby('education').agg(
    avg_claim_freq=('claim_freq', 'mean'),
    avg_claim_amt=('claim_amt', 'mean')
).reset_index()

print("Average Claims by Education: ")
print(avg_claims_by_education)
print(avg_claims_by_parent)

# Group by 'coverage_zone'
avg_claims_by_coverage_zone = df.groupby('coverage_zone').agg(
    avg_claim_freq=('claim_freq', 'mean'),
    avg_claim_amt=('claim_amt', 'mean')
).reset_index()

print("Average Claims by Coverage Zone: ")
print(avg_claims_by_coverage_zone)
```

```
0  Commercial        0.498663   50003.247874
1     Private        0.513207   50034.305110
Average Claims by Parent:
  parent  avg_claim_freq  avg_claim_amt
0     No        0.510032   50084.902634
1    Yes        0.510656   49957.452996
Average Claims by Coverage Zone:
  coverage_zone  avg_claim_freq  avg_claim_amt
0  Highly Rural        0.500403   49998.132178
1  Highly Urban        0.516503   49861.036665
2         Rural        0.506381   49778.020247
3      Suburban        0.520091   50124.843185
4         Urban        0.508171   50377.730389
```

The average claim frequency and average claim amount are all around the median

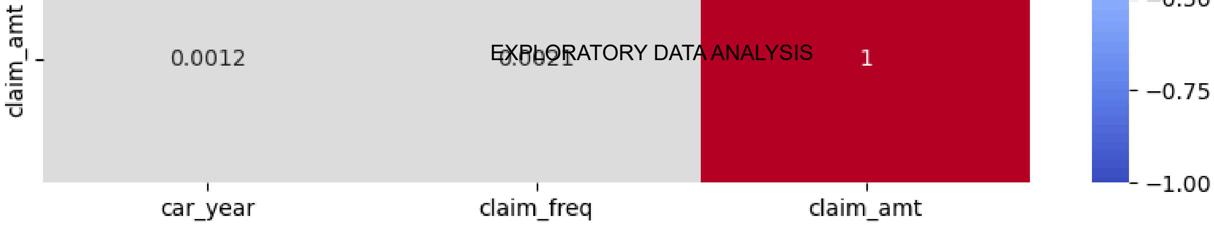
In [ ]:

2. Are there any specific vehicle characteristics (e.g., make, model and year) that correlate with higher claim frequencies or amounts?

for Car Year

In [8]:

```python
# Calculate correlation matrix for vehicle characteristics and claim metrics

vehicle_year = df[['car_year', 'claim_freq', 'claim_amt']]
correlation_year = vehicle_year.corr()


# Correlation heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_year, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Car Year')
plt.show()
```



There is no correlation between the car year, claim frequency, and claim amount

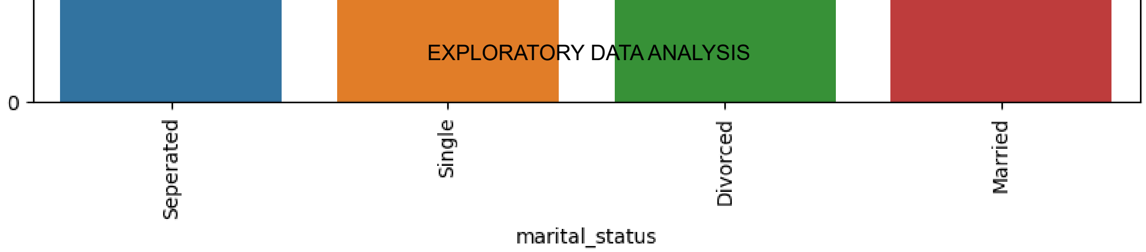
```
In [ ]:
```

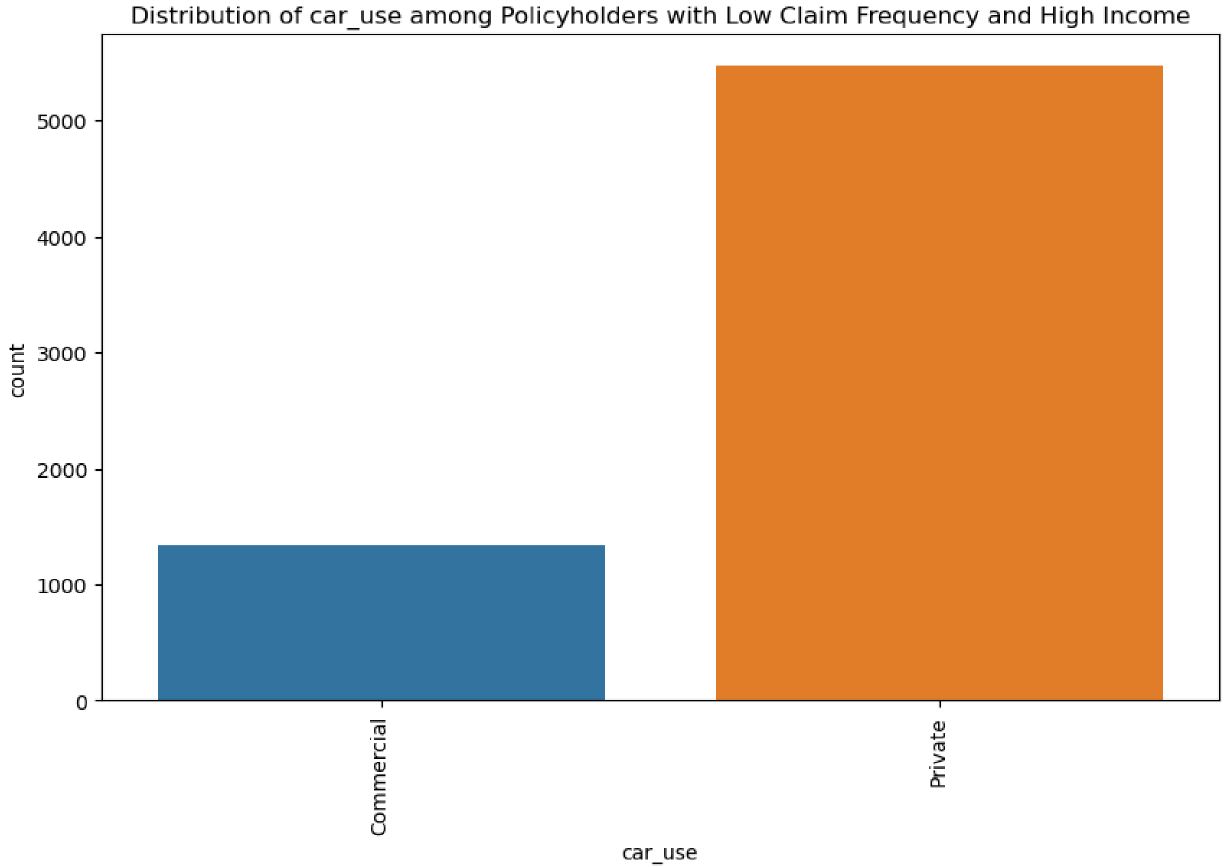
For Car Make

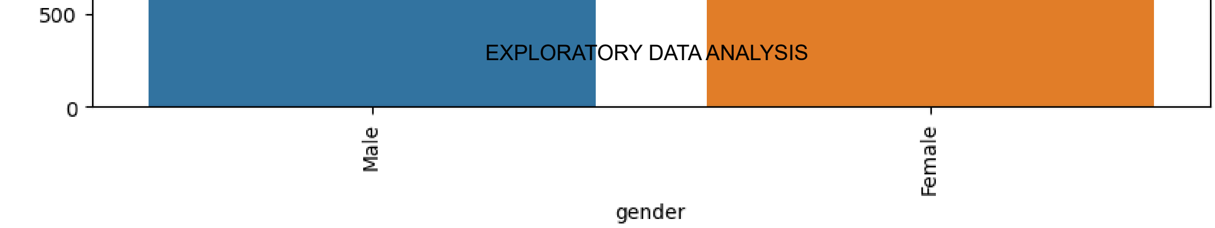
```
In [9]:
# Define a function to calculate chi-square for categorical variables
def chi_square_test(df, column, target):
    contingency_table = pd.crosstab(df[column], df[target]>df[target].median())
    chi2, p, dof, expected = stats.chi2_contingency(contingency_table)
    return chi2, p

# Apply chi-square test for 'car_make' and 'car_model'

chi2_car_make_freq, p_car_make_freq = chi_square_test(df, 'car_make', 'claim_freq')
chi2_car_make_amt, p_car_make_amt = chi_square_test(df, 'car_make', 'claim_amt')
chi2_car_model_freq, p_car_model_freq = chi_square_test(df, 'car_model', 'claim_freq')
chi2_car_model_amt, p_car_model_amt = chi_square_test(df, 'car_model', 'claim_amt')

print(f"Chi-square test for car make frequeny claim: chi2={chi2_car_make_freq}, p-valu
print(f"Chi-square test for car make amount claim: chi2={chi2_car_make_amt}, p-value={
print(f"Chi-square test for car model frequency claim: chi2={chi2_car_model_freq}, p-v
print(f"Chi-square test for car model amount claim: chi2={chi2_car_model_amt}, p-value

# Calculate the critical value

alpha = 0.05
shape = 563115
critical_value = stats.chi2.ppf(1 - alpha, shape)
print(f'Critical value for the Chi-square test = {critical_value}')
```

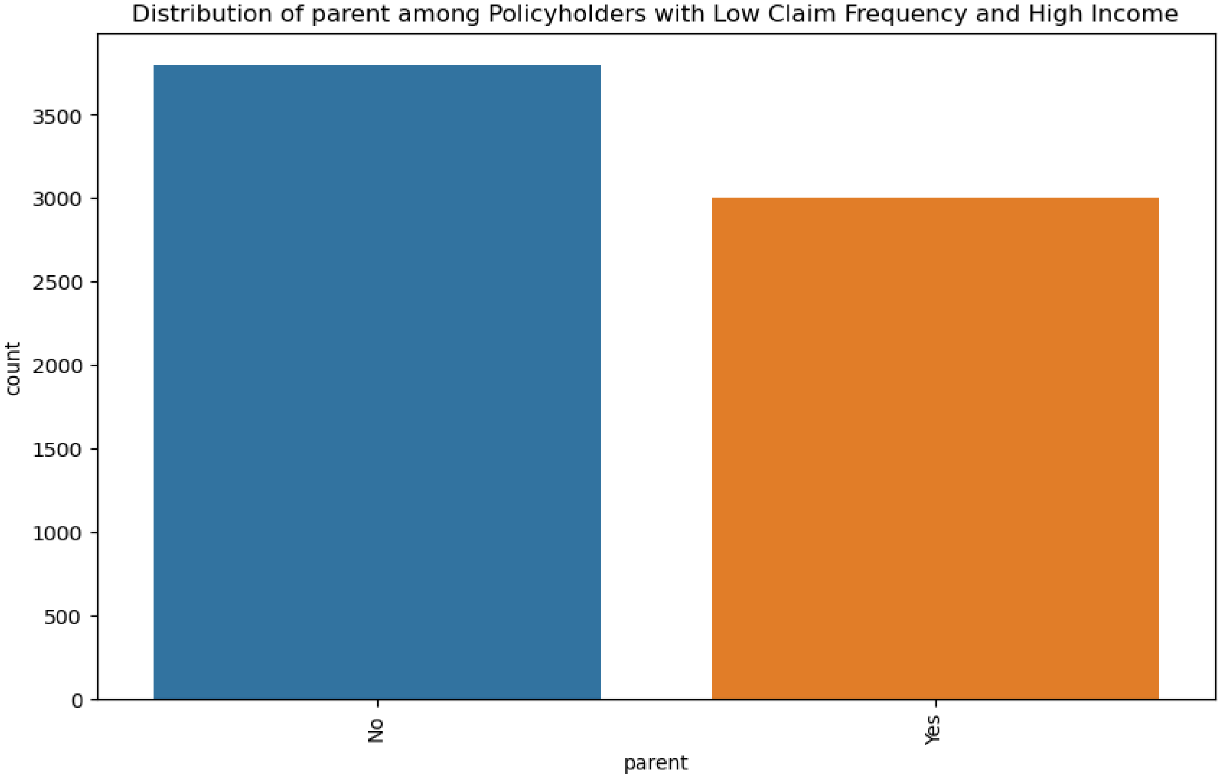
In [10]:

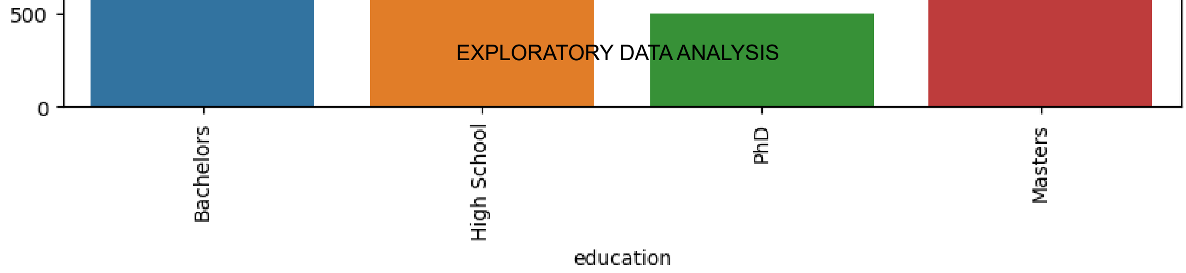
```
# Define thresholds
low_claim_freq_threshold = df['claim_freq'].quantile(0.25)
high_income_threshold = df['household_income'].quantile(0.75)

# Filter policyholders with low claim frequencies and high household incomes
low_claim_high_income = df[(df['claim_freq'] <= low_claim_freq_threshold) & (df['house


# Distribution of key categorical characteristics
categorical_vars = ['marital_status', 'car_use', 'gender', 'parent', 'education', 'car
                    'coverage_zone']

for var in categorical_vars:
    plt.figure(figsize=(10, 6))
    sns.countplot(x=low_claim_high_income[var])
    plt.title(f'Distribution of {var} among Policyholders with Low Claim Frequency anc
    plt.xticks(rotation=90)
    plt.show()

# Distribution of key numerical characteristics
numerical_vars = ['kids_driving', 'car_year']

for var in numerical_vars:
    plt.figure(figsize=(10, 6))
    sns.histplot(low_claim_high_income[var], bins=30, kde=True)
    plt.title(f'Distribution of {var} among Policyholders with Low Claim Frequency anc
    plt.show()
```



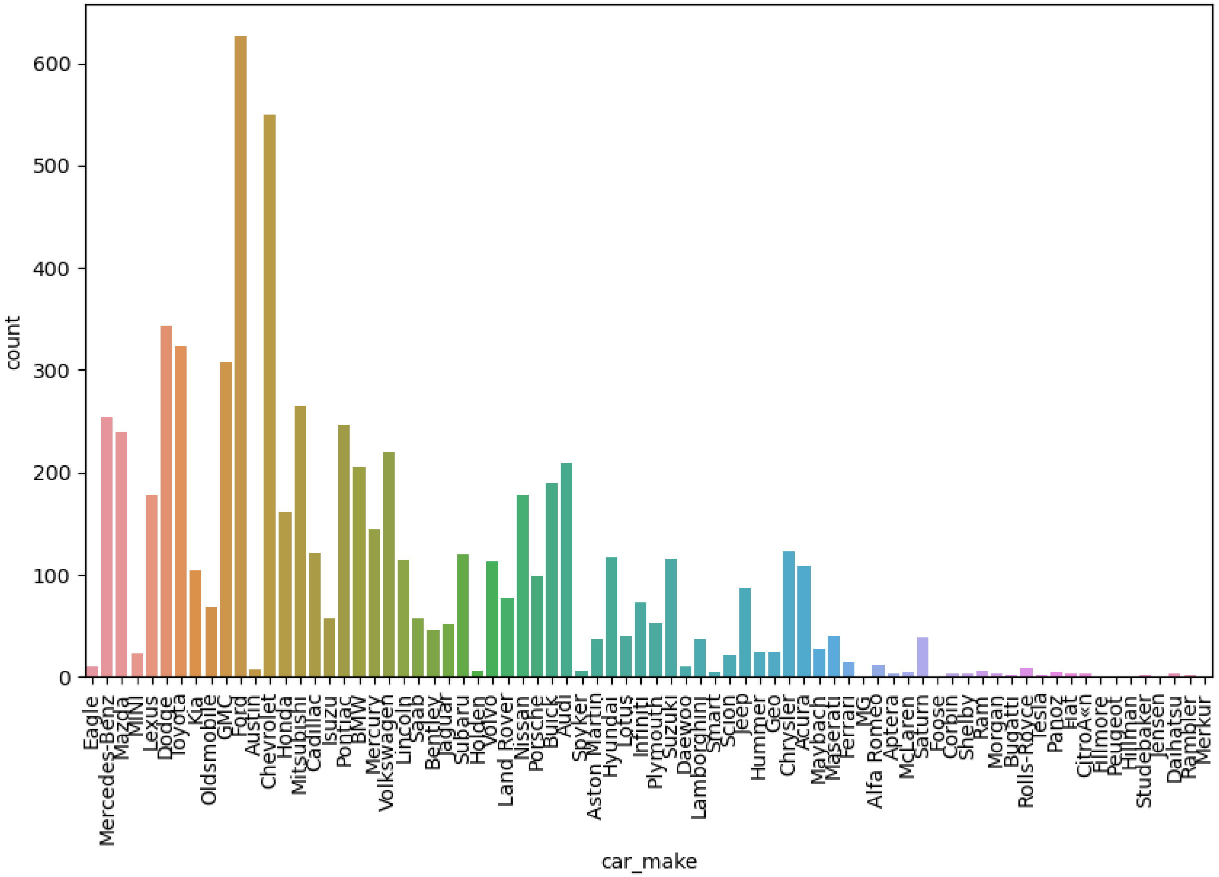

Distribution of car_use among Policyholders with Low Claim Frequency and High Income

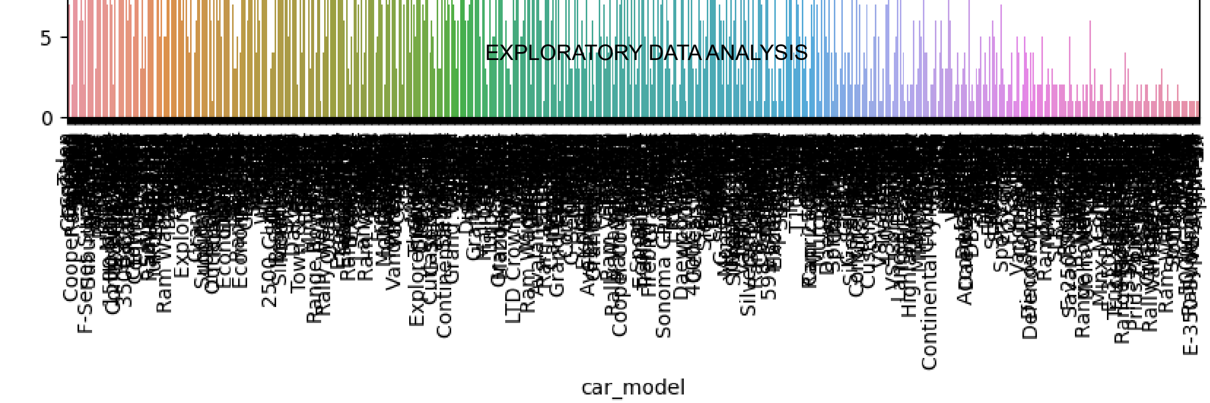



Distribution of parent among Policyholders with Low Claim Frequency and High Income

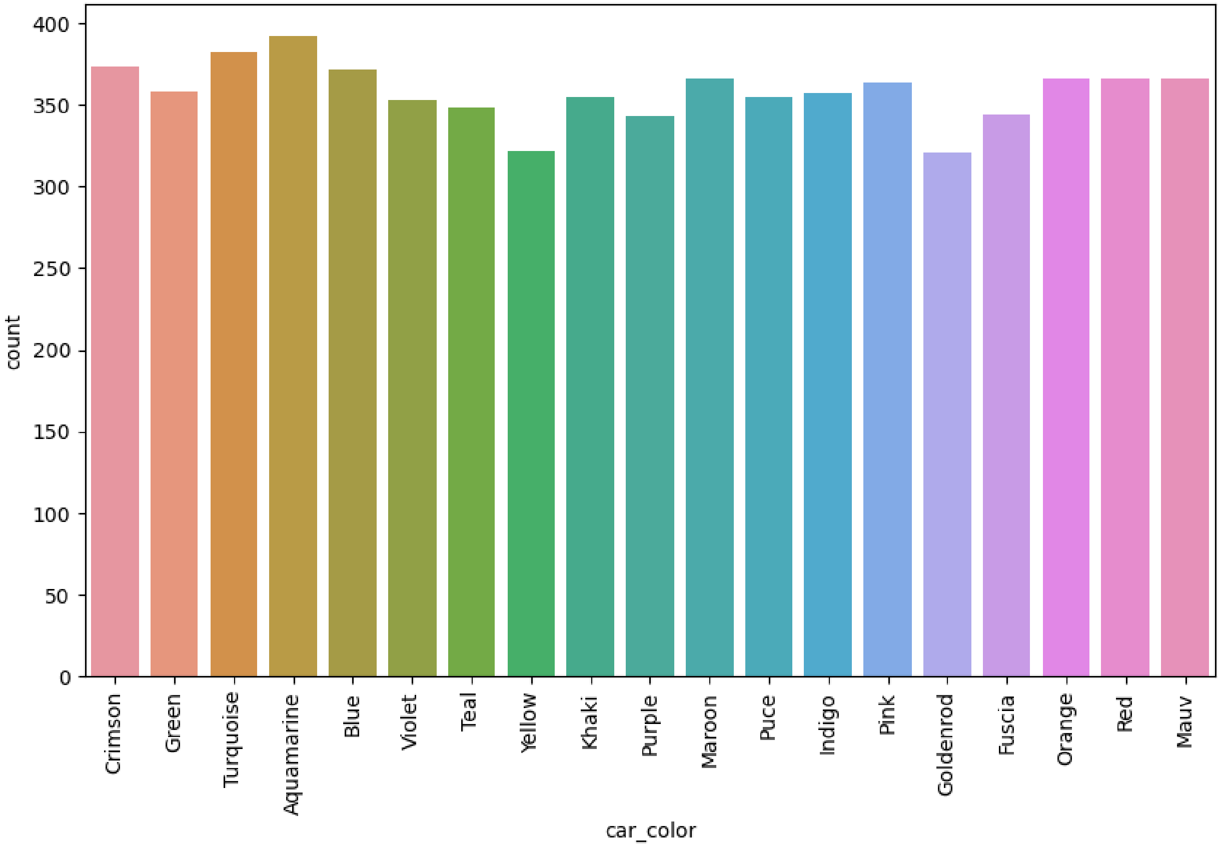


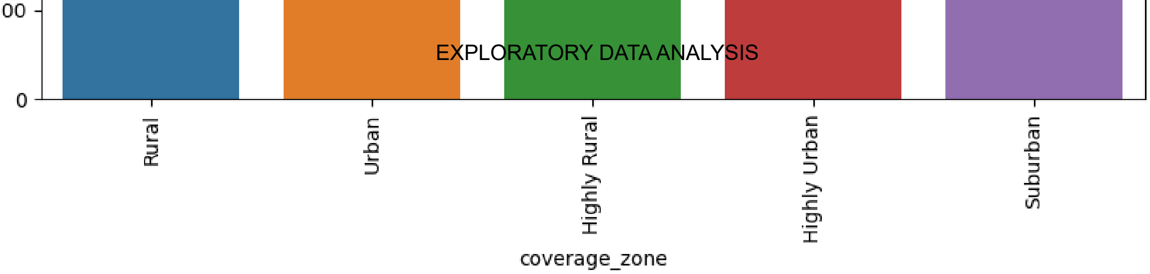
Distribution of car_make among Policyholders with Low Claim Frequency and High Income

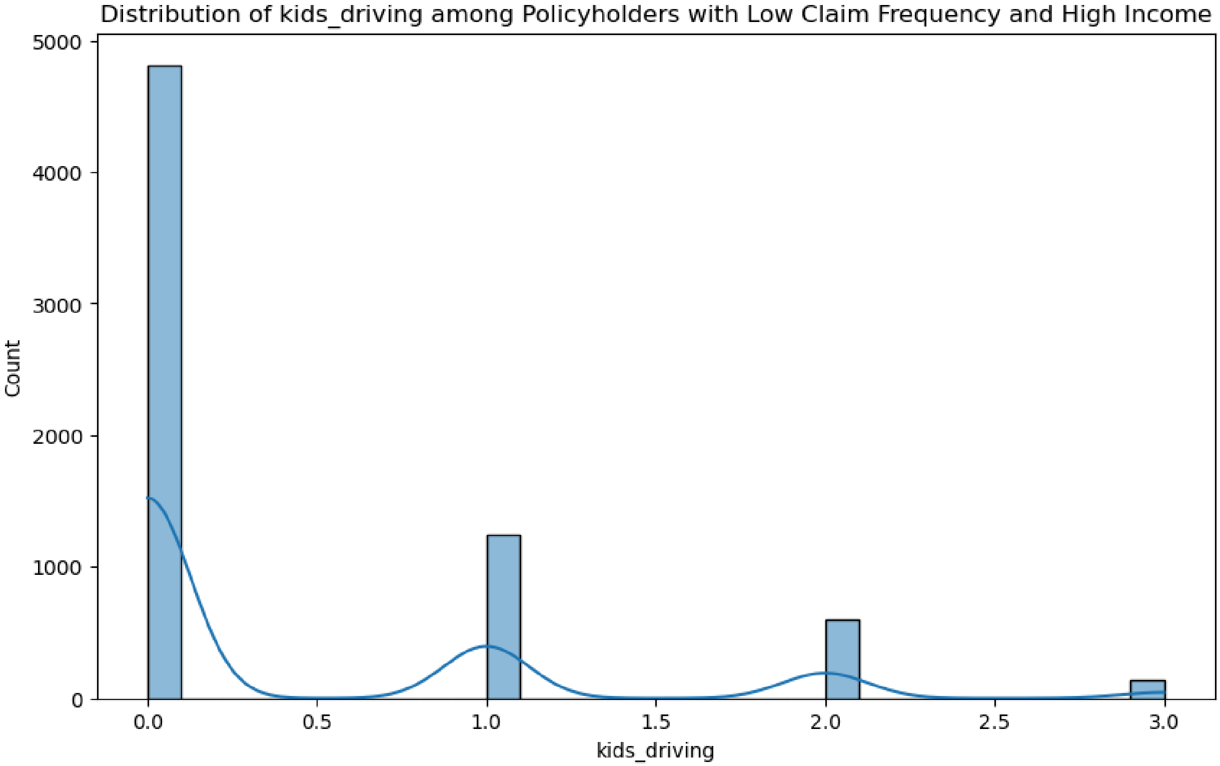


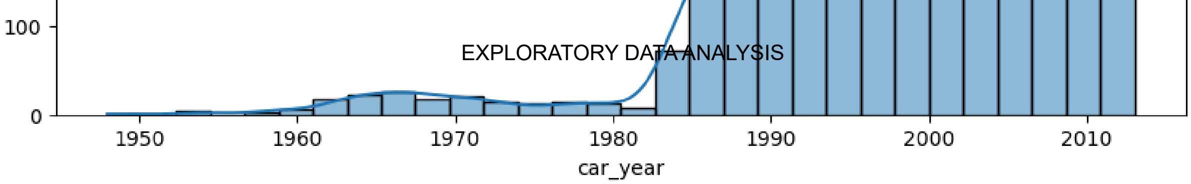


Distribution of car_color among Policyholders with Low Claim Frequency and High Income




200
0
Rural
Urban
Highly Rural
Highly Urban
Suburban
coverage_zone


Distribution of kids_driving among Policyholders with Low Claim Frequency and High Income
Count
5000
4000
3000
2000
1000
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
kids_driving



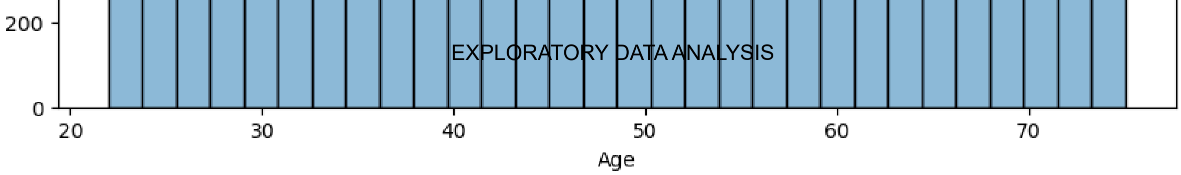
The percentage distribution were the same with the descriptive statistics except for coverage zone where highly rural has a higher percentage.

In [ ]:

4. How does the distribution of policyholders vary across different demographic factors (age, gender, marital status)?

In [18]:

```
# Convert birthdate to age
df['birthdate'] = pd.to_datetime(df['birthdate'])
current_year = pd.to_datetime('today').year
df['age'] = current_year - df['birthdate'].dt.year



# Plot age distribution
plt.figure(figsize=(10, 6))
sns.histplot(df['age'], bins=30, kde=True)
plt.title('Distribution of Policyholders by Age')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()


# Summary statistics for age
age_stats = df['age'].describe()
print('Age Statistics:')
print(age_stats)
```



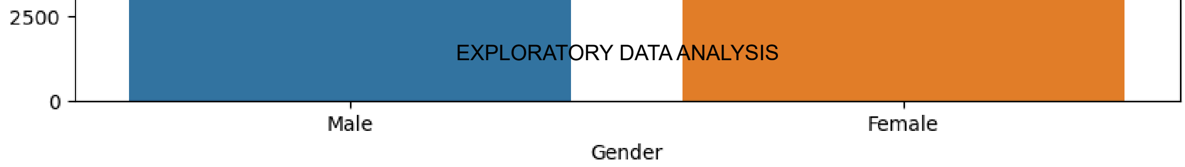
```
Age Statistics:
count    37542.000000
mean        48.153721
std         15.295082
min         22.000000
25%         35.000000
50%         48.000000
75%         61.000000
max         75.000000
Name: age, dtype: float64
```

Each decade in the range of 20 < age > 80 is properly distributed

In [ ]:

In [19]:

```
# Plot gender distribution
plt.figure(figsize=(10, 6))
sns.countplot(x=df['gender'])
plt.title('Distribution of Policyholders by Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()


# Summary statistics for gender
gender_stats = df['gender'].value_counts(normalize=True) * 100
print('Gender Distribution (Percentage):')
print(gender_stats)
```



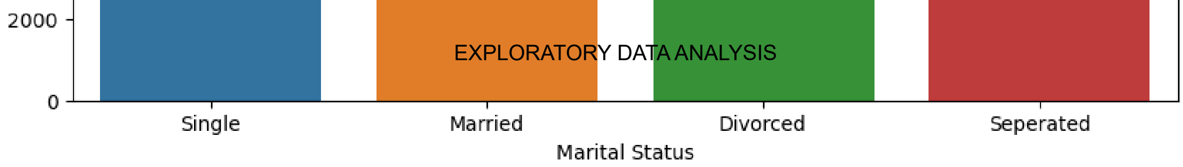
```
Gender Distribution (Percentage):
gender
Female    50.093229
Male      49.906771
Name: proportion, dtype: float64
```

Both genders are almost equal in distribution

In [ ]:

In [20]:

```
# Plot marital status distribution
plt.figure(figsize=(10, 6))
sns.countplot(x=df['marital_status'])
plt.title('Distribution of Policyholders by Marital Status')
plt.xlabel('Marital Status')
plt.ylabel('Count')
plt.show()


# Summary statistics for marital status
marital_status_stats = df['marital_status'].value_counts(normalize=True) * 100
print('Marital Status Distribution (Percentage):')
print(marital_status_stats)
```



```
Marital Status Distribution (Percentage):
marital_status
Single       41.353684
Married      33.482500
Divorced     16.933035
Seperated     8.230782
Name: proportion, dtype: float64
```

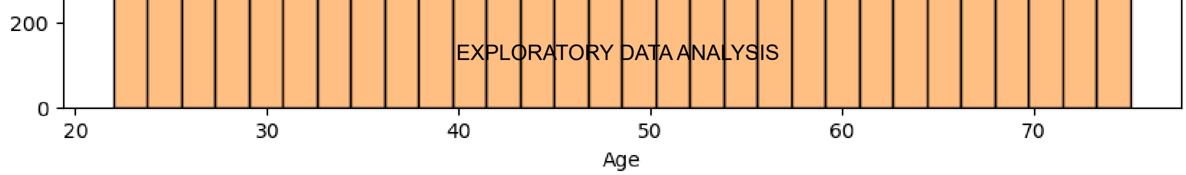
There is a significantly higher percentage of single and married marital status

In [ ]:

```

```

In [14]:

```python
# Plot age distribution by gender
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='age', hue='gender', bins=30, kde=True, multiple="stack")
plt.title('Distribution of Policyholders by Age and Gender')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```

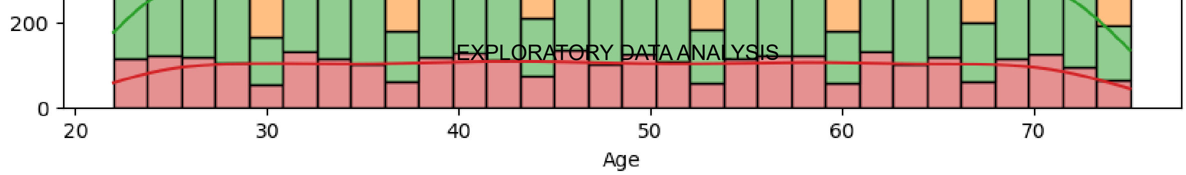


The genders are evenly distributed across all ages

In [ ]:

In [15]:

```
# Plot age distribution by marital status
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='age', hue='marital_status', bins=30, kde=True, multiple="stac
plt.title('Distribution of Policyholders by Age and Marital Status')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



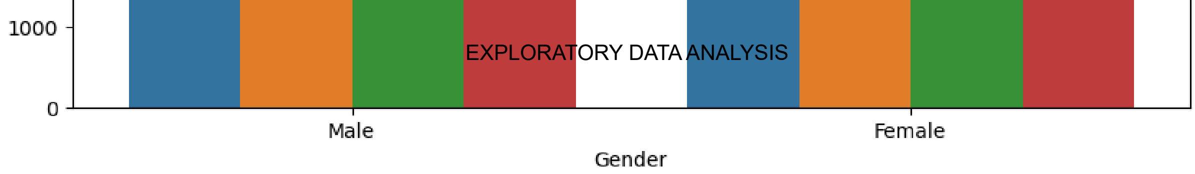
The trend on the marital status is equally distrubuted across all ages

In [ ]:

```
```

In [16]:

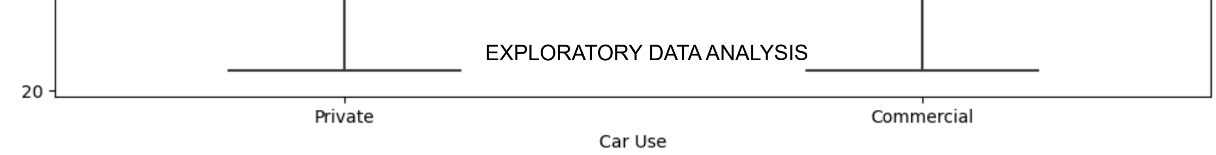
```
# Plot gender distribution by marital status
plt.figure(figsize=(10, 6))
sns.countplot(x='gender', hue='marital_status', data=df)
plt.title('Distribution of Policyholders by Gender and Marital Status')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```

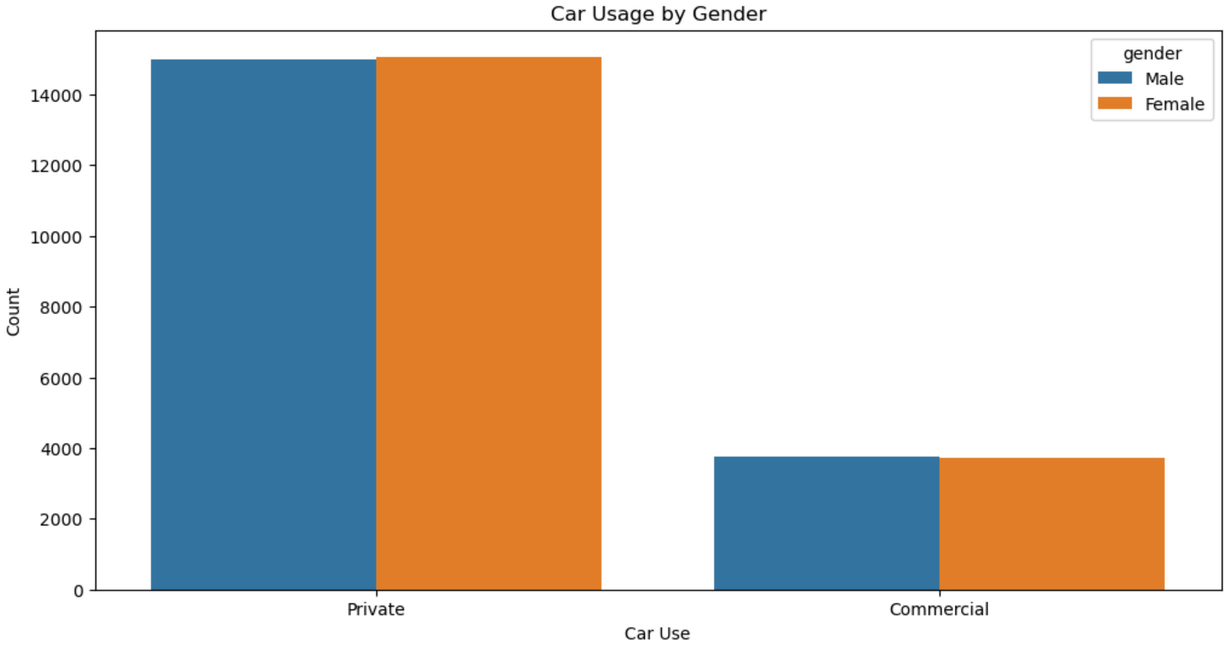


The trend across the marital status is evenly distributed across all genders

5. Are there any noticeable trends in car usage and ownership among different demographic groups?

```
#Car usage by demographic group



# Plot car usage by age group
plt.figure(figsize=(12, 6))
sns.boxplot(x='car_use', y='age', data=df)
plt.title('Car Usage by Age Group')
plt.xlabel('Car Use')
plt.ylabel('Age')
plt.show()

# Plot car usage by gender
plt.figure(figsize=(12, 6))
sns.countplot(x='car_use', hue='gender', data=df)
plt.title('Car Usage by Gender')
plt.xlabel('Car Use')
plt.ylabel('Count')
plt.show()

# Plot car usage by marital status
plt.figure(figsize=(12, 6))
sns.countplot(x='car_use', hue='marital_status', data=df)
plt.title('Car Usage by Marital Status')
plt.xlabel('Car Use')
```

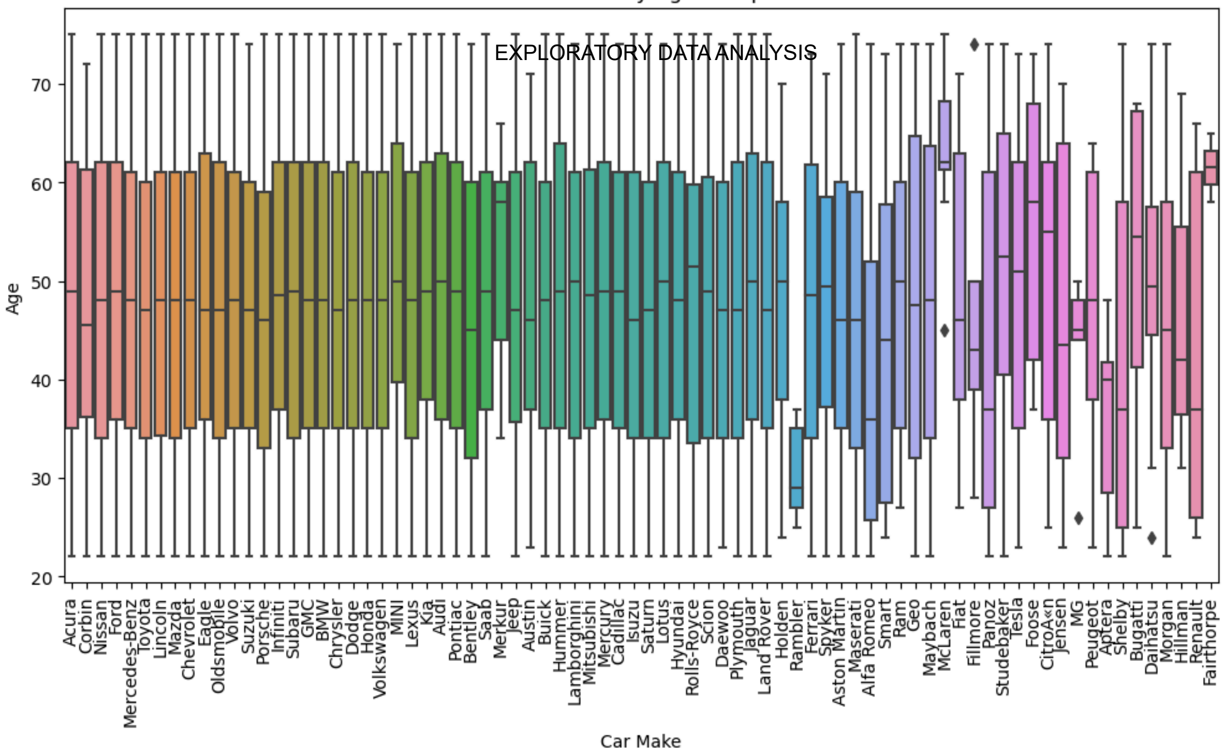

20
Private
Commercial
Car Use

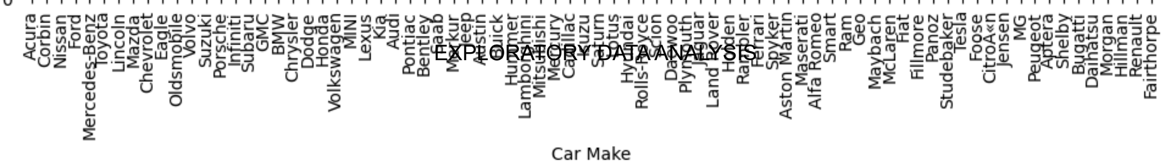

Car Usage by Gender
gender
🟦 Male
🟧 Female
Count
0
2000
4000
6000
8000
10000
12000
14000
Private
Commercial
Car Use

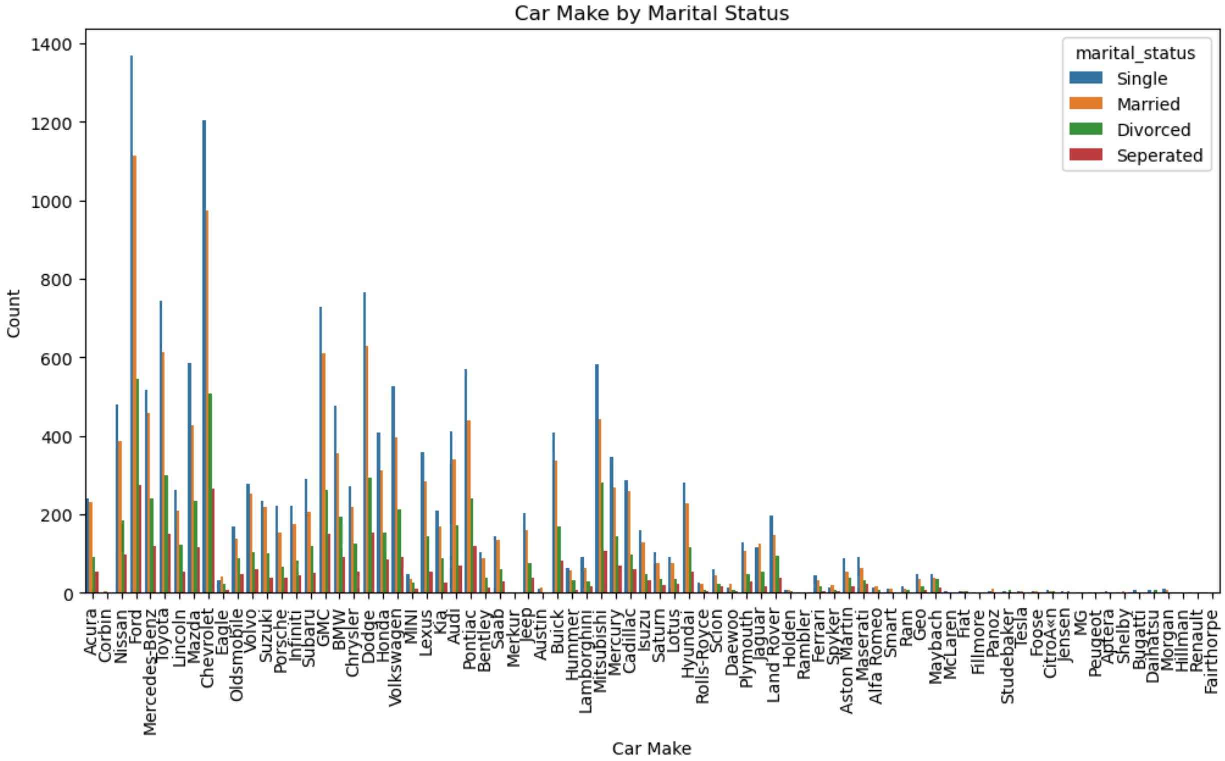
Car usage by demographic group are evenly distributed

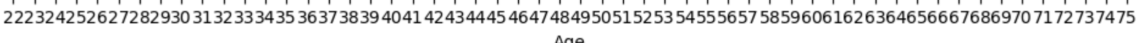
In [23]:

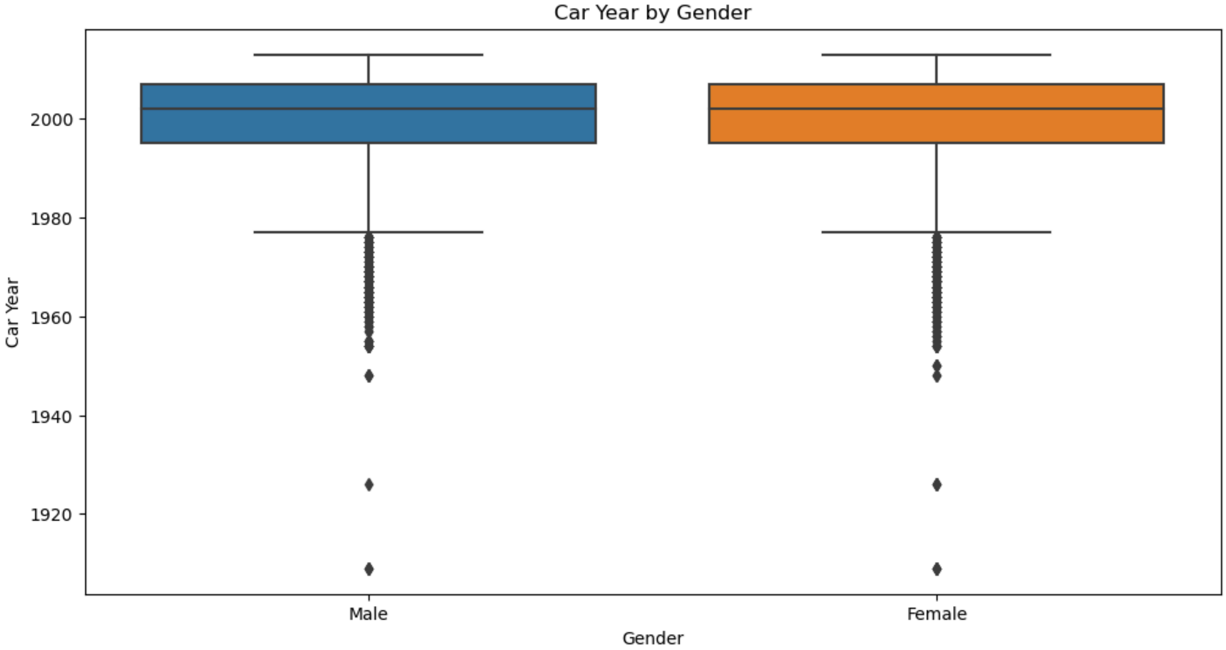
```
#Car ownership by demographic group


# Plot car make by age group
plt.figure(figsize=(12, 6))
sns.boxplot(x='car_make', y='age', data=df)
plt.title('Car Make by Age Group')
plt.xlabel('Car Make')
plt.ylabel('Age')
plt.xticks(rotation=90)
plt.show()

# Plot car make by gender
plt.figure(figsize=(12, 6))
sns.countplot(x='car_make', hue='gender', data=df)
plt.title('Car Make by Gender')
plt.xlabel('Car Make')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()

# Plot car make by marital status
plt.figure(figsize=(12, 6))
sns.countplot(x='car_make', hue='marital_status', data=df)
plt.title('Car Make by Marital Status')
plt.xlabel('Car Make')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()

# Plot car year by age group
plt.figure(figsize=(12, 6))
sns.boxplot(x='age', y='car_year', data=df)
plt.title('Car Year by Age Group')
plt.xlabel('Age')
plt.ylabel('Car Year')
```


Age
70
60
50
40
30
20
Car Make
Acura
Corbin
Nissan
Ford
Mercedes-Benz
Toyota
Lincoln
Mazda
Chevrolet
Eagle
Oldsmobile
Volvo
Suzuki
Porsche
Infiniti
Subaru
GMC
BMW
Chrysler
Dodge
Honda
Volkswagen
MINI
Lexus
Kia
Audi
Pontiac
Bentley
Saab
Merkur
Jeep
Austin
Buick
Hummer
Lamborghini
Mitsubishi
Mercury
Cadillac
Isuzu
Saturn
Lotus
Hyundai
Rolls-Royce
Scion
Daewoo
Plymouth
Jaguar
Land Rover
Holden
Rambler
Ferrari
Spyker
Aston Martin
Maserati
Alfa Romeo
Smart
Ram
Geo
Maybach
McLaren
Fiat
Fillmore
Panoz
Studebaker
Tesla
Foose
CitroA«n
Jensen
MG
Peugeot
Aptera
Shelby
Bugatti
Daihatsu
Morgan
Hillman
Renault
Fairthorpe


0
Acura
Corbin
Nissan
Ford
Mercedes-Benz
Toyota
Lincoln
Mazda
Chevrolet
Eagle
Oldsmobile
Volvo
Suzuki
Porsche
Infiniti
Subaru
GMC
BMW
Chrysler
Dodge
Honda
Volkswagen
MINI
Lexus
Kia
Audi
Pontiac
Bentley
Saab
Merkur
Jeep
Austin
Buick
Hummer
Lamborghini
Mitsubishi
Mercury
Cadillac
Isuzu
Saturn
Lotus
Hyundai
Rolls-Royce
Scion
Daewoo
Plymouth
Jaguar
Land Rover
Holden
Rambler
Ferrari
Spyker
Aston Martin
Maserati
Alfa Romeo
Smart
Ram
Geo
Maybach
McLaren
Fiat
Fillmore
Panoz
Studebaker
Tesla
Foose
CitroA«n
Jensen
MG
Peugeot
Aptera
Shelby
Bugatti
Daihatsu
Morgan
Hillman
Renault
Fairthorpe
Car Make


Car Make by Marital Status
Count
1400
1200
1000
800
600
400
200
0
marital_status
Single
Married
Divorced
Seperated
Acura
Corbin
Nissan
Ford
Mercedes-Benz
Toyota
Lincoln
Mazda
Chevrolet
Eagle
Oldsmobile
Volvo
Suzuki
Porsche
Infiniti
Subaru
GMC
BMW
Chrysler
Dodge
Honda
Volkswagen
MINI
Lexus
Kia
Audi
Pontiac
Bentley
Saab
Merkur
Jeep
Austin
Buick
Hummer
Lamborghini
Mitsubishi
Mercury
Cadillac
Isuzu
Saturn
Lotus
Hyundai
Rolls-Royce
Scion
Daewoo
Plymouth
Jaguar
Land Rover
Holden
Rambler
Ferrari
Spyker
Aston Martin
Maserati
Alfa Romeo
Smart
Ram
Geo
Maybach
McLaren
Fiat
Fillmore
Panoz
Studebaker
Tesla
Foose
CitroA«n
Jensen
MG
Peugeot
Aptera
Shelby
Bugatti
Daihatsu
Morgan
Hillman
Renault
Fairthorpe
Car Make


222324252627282930 31 3233343536373839404142434445464748495051525354555657585960616263646566676869 70 7172737475
Age


Car Year by Gender
2000
1980
1960
1940
1920
Car Year
Male
Female
Gender

Majority of the car make are evenly distributed across the age group

There are very little changes in the choice of car make across the different genders

The trend in the marital status is also reflected in the choice of a car make

majority of the age group prefer cars made in the 2000s

There is an equal distribution in the car year across the genders

The car year below the 1940s were common among the marital status with the exception of the separated

6. How do claim frequencies and amounts vary across different coverage zones?

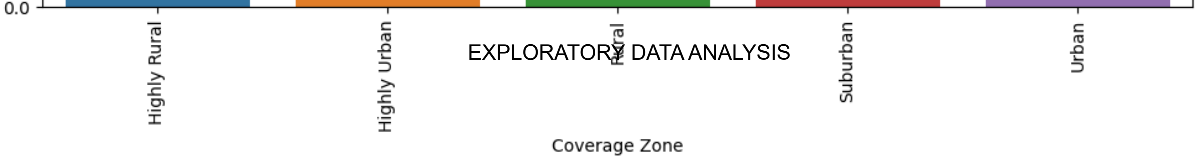
In [25]:

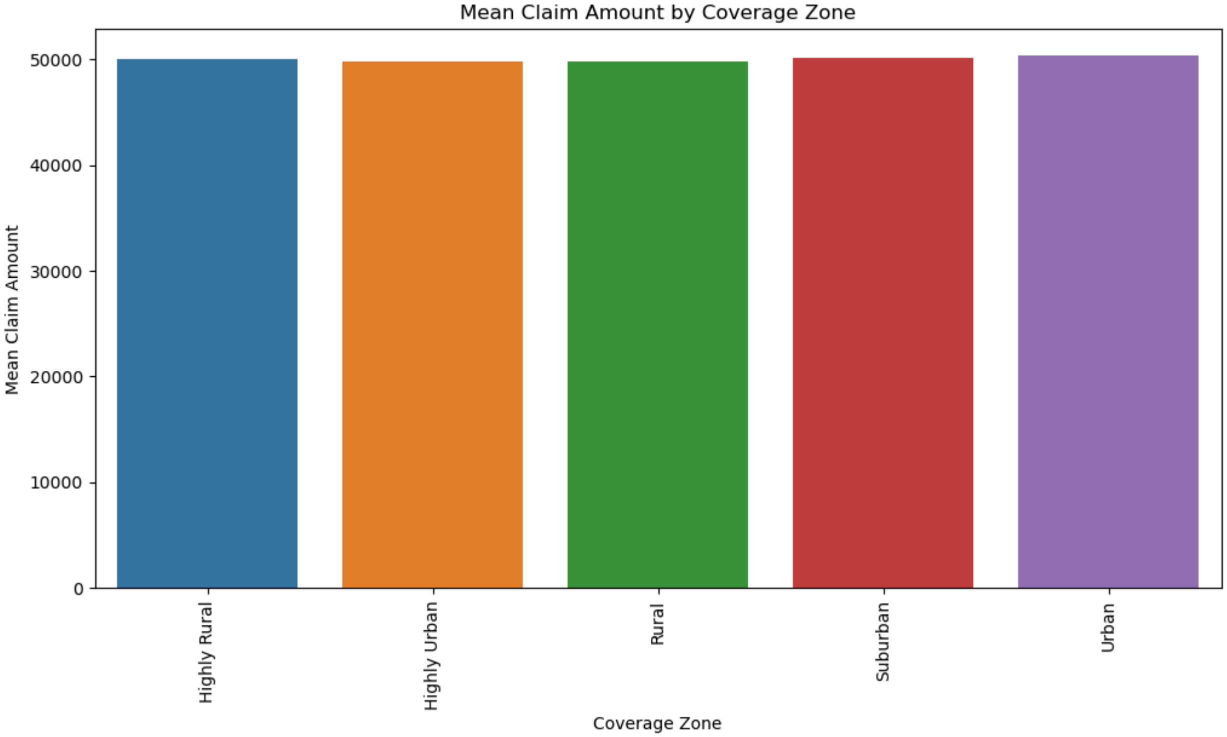
```
#Summary statistics by coverage zone

# Group by coverage zone and calculate summary statistics
coverage_zone_stats = df.groupby('coverage_zone').agg({
    'claim_freq': ['mean', 'median', 'std', 'min', 'max'],
    'claim_amt': ['mean', 'median', 'std', 'min', 'max']
}).reset_index()

# Rename columns for clarity
coverage_zone_stats.columns = ['coverage_zone', 'mean_claim_freq', 'median_claim_freq'
                               'max_claim_freq', 'mean_claim_amt', 'median_claim_amt',
                               'max_claim_amt']

print(coverage_zone_stats)
```

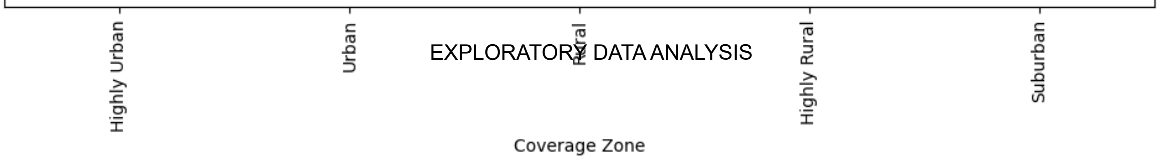
```
3   28669.617026           22.00        99994.03
4   28642.245078            0.04        99975.60
```

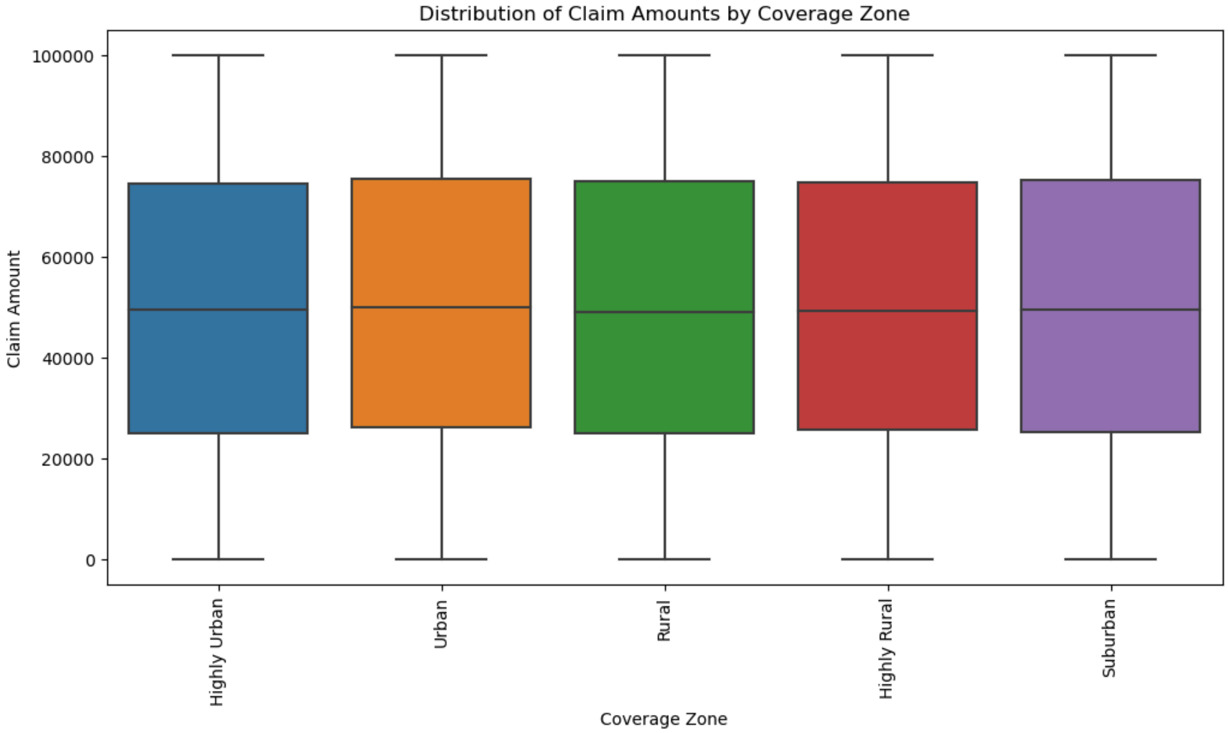
In [26]:

```
#Visualization of the distribution


# Plot mean claim frequency by coverage zone
plt.figure(figsize=(12, 6))
sns.barplot(x='coverage_zone', y='mean_claim_freq', data=coverage_zone_stats)
plt.title('Mean Claim Frequency by Coverage Zone')
plt.xlabel('Coverage Zone')
plt.ylabel('Mean Claim Frequency')
plt.xticks(rotation=90)
plt.show()

# Plot mean claim amount by coverage zone
plt.figure(figsize=(12, 6))
sns.barplot(x='coverage_zone', y='mean_claim_amt', data=coverage_zone_stats)
plt.title('Mean Claim Amount by Coverage Zone')
plt.xlabel('Coverage Zone')
plt.ylabel('Mean Claim Amount')
plt.xticks(rotation=90)
plt.show()

# Plot distribution of claim frequencies by coverage zone
plt.figure(figsize=(12, 6))
sns.boxplot(x='coverage_zone', y='claim_freq', data=df)
plt.title('Distribution of Claim Frequencies by Coverage Zone')
plt.xlabel('Coverage Zone')
plt.ylabel('Claim Frequency')
plt.xticks(rotation=90)
plt.show()

# Plot distribution of claim amounts by coverage zone
plt.figure(figsize=(12, 6))
sns.boxplot(x='coverage_zone', y='claim_amt', data=df)
plt.title('Distribution of Claim Amounts by Coverage Zone')
plt.xlabel('Coverage Zone')
plt.ylabel('Claim Amount')
plt.xticks(rotation=90)
plt.show()
```


0.0
Highly Rural
Highly Urban
Rural
Suburban
Urban
Coverage Zone


Mean Claim Amount by Coverage Zone
Mean Claim Amount
50000
40000
30000
20000
10000
0
Highly Rural
Highly Urban
Rural
Suburban
Urban
Coverage Zone





The stat for the claim frequency and claim amount across all coverage zones are all within the range of +- 0.03 for claim frequency and +-1000 for claim amount

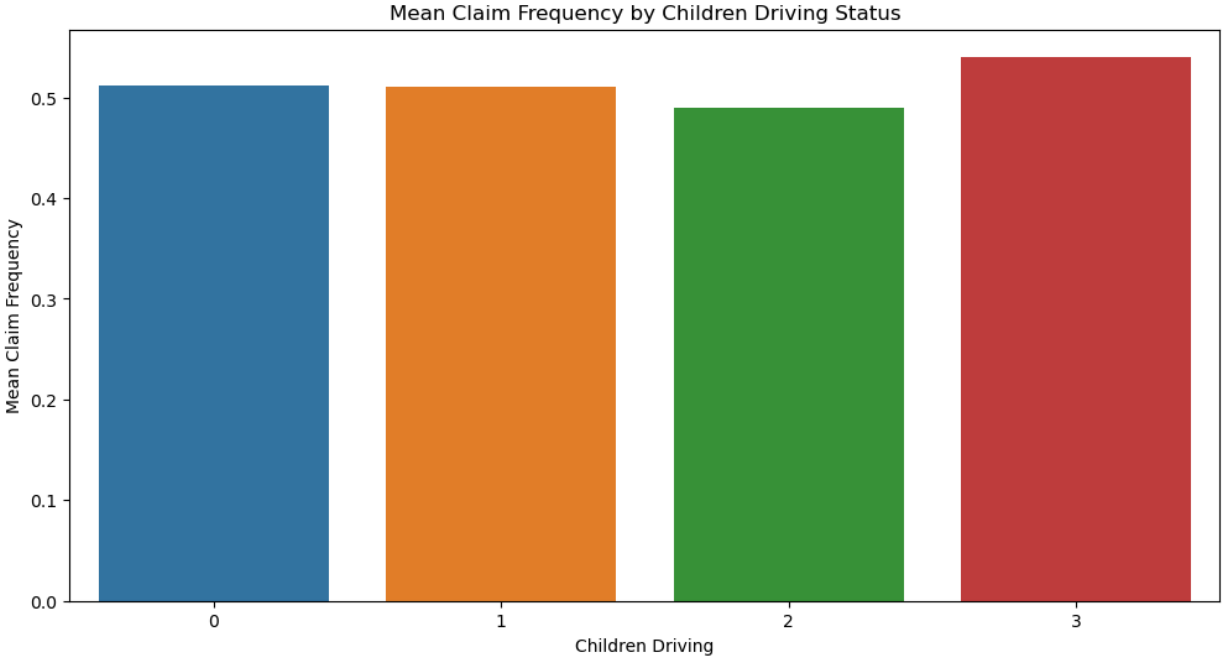
7. Are there any trends or patterns in the behavior of policyholders who have children driving?

In [27]:

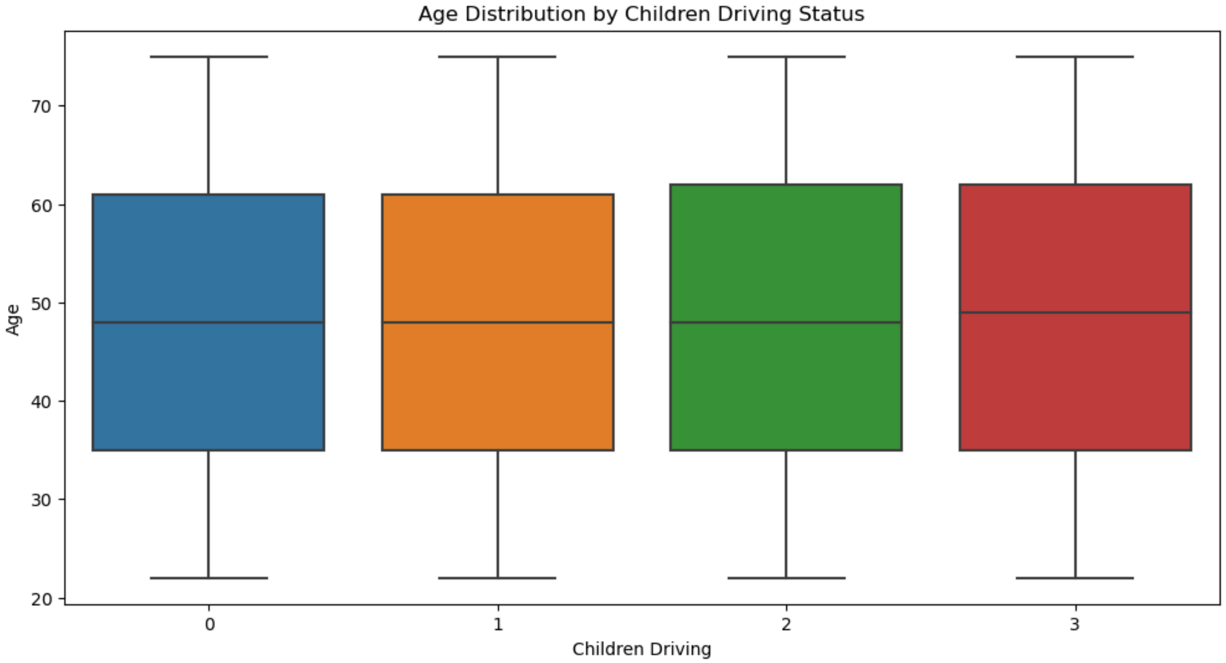
```
# Group by children driving status and calculate summary statistics
children_driving_stats = df.groupby('kids_driving').agg({
    'claim_freq': ['mean', 'median', 'std', 'min', 'max'],
    'claim_amt': ['mean', 'median', 'std', 'min', 'max'],
2               0               4    49804.218819         49383.875
3               0               4    52329.871988         52360.600

   std_claim_amt  min_claim_amt  max_claim_amt   mean_age  median_age  \
0   28802.963773          19.70       99997.70  48.078359        48.0
1   28290.290812           0.04       99993.69  48.274588        48.0
2   28777.752103          78.61       99975.59  48.454933        48.0
3   28621.188930         534.73       99991.40  48.411012        49.0

     std_age  min_age  max_age
0  15.302392       22       75
1  15.259609       22       75
2  15.269678       22       75
3  15.463566       22       75
```

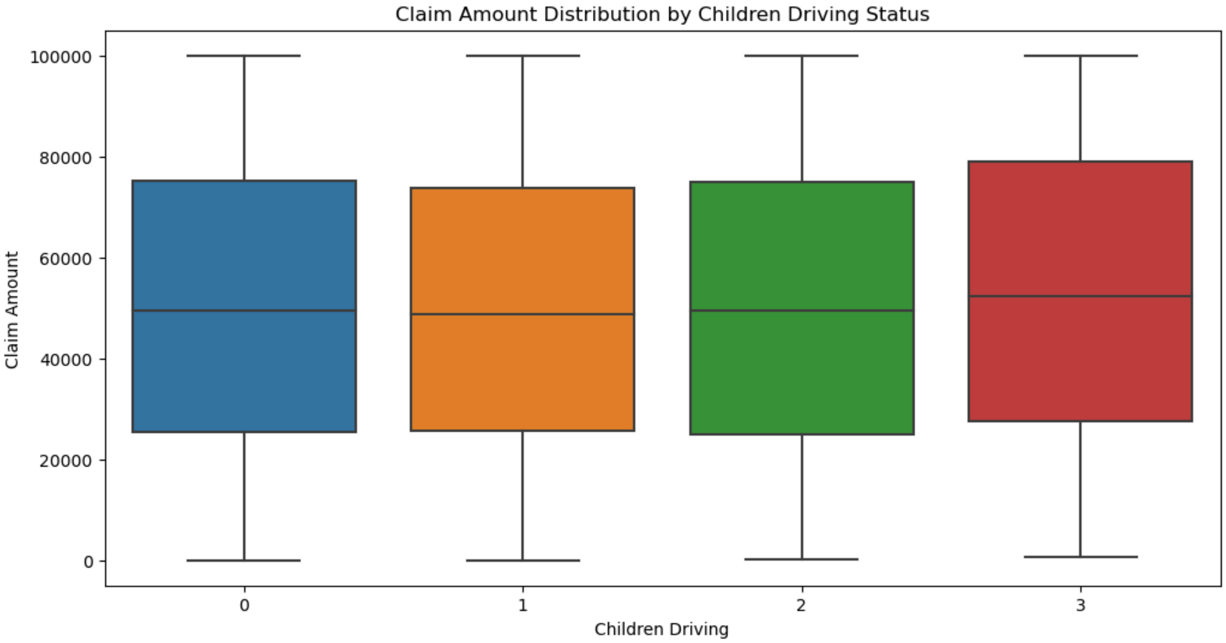
In [28]:

```
#Visualization of the distribution

# Plot mean claim frequency by children driving status
plt.figure(figsize=(12, 6))
sns.barplot(x='kids_driving', y='mean_claim_freq', data=children_driving_stats)
plt.title('Mean Claim Frequency by Children Driving Status')
plt.xlabel('Children Driving')
plt.ylabel('Mean Claim Frequency')
plt.xticks(rotation=0)
plt.show()

# Plot mean claim amount by children driving status
plt.figure(figsize=(12, 6))
sns.barplot(x='kids_driving', y='mean_claim_amt', data=children_driving_stats)
plt.title('Mean Claim Amount by Children Driving Status')
plt.xlabel('Children Driving')
plt.ylabel('Mean Claim Amount')
plt.xticks(rotation=0)
plt.show()

# Plot age distribution by children driving status
plt.figure(figsize=(12, 6))
sns.boxplot(x='kids_driving', y='age', data=df)
plt.title('Age Distribution by Children Driving Status')
plt.xlabel('Children Driving')
plt.ylabel('Age')
plt.xticks(rotation=0)
plt.show()
```



0
1
2
3
Children Driving


Age Distribution by Children Driving Status
Age
70
60
50
40
30
20
0
1
2
3
Children Driving



Those with 3 kids have a higher claim frequency amd claim amount distribution

8. How does the presence of children driving affect the frequency and number of claims?

In [33]:

```
# T-test for claim frequency

from scipy.stats import ttest_ind

# Separate data into groups
group_with_kids = df[df['kids_driving'] > 0]['claim_freq']
group_without_kids = df[df['kids_driving'] == 0]['claim_freq']

# Perform t-test
t_stat, p_value = ttest_ind(group_with_kids, group_without_kids)
print(f'T-test for Claim Frequency: t-statistic = {t_stat}, p-value = {p_value}')
```

In [32]:

```
#T-test for claim amount

# Separate data into groups
group_with_kids_amt = df[df['kids_driving'] > 0]['claim_amt']
group_without_kids_amt = df[df['kids_driving'] == 0]['claim_amt']

# Perform t-test
t_stat_amt, p_value_amt = ttest_ind(group_with_kids_amt, group_without_kids_amt)
print(f'T-test for Claim Amount: t-statistic = {t_stat_amt}, p-value = {p_value_amt}')

#Critical value
n1 = (df['kids_driving'] > 0).sum()
n2 = (df['kids_driving'] == 0).sum()
alpha = 0.05
dof = n1 + n2 - 2
critical_value = stats.t.ppf(1 - alpha / 2, dof)
print(f'The critical value for the t-test at alpha = {alpha} and dof = {dof} is: {crit
```

```
T-test for Claim Amount: t-statistic = -0.8362472799449246, p-value = 0.4030211278733
7296
The critical value for the t-test at alpha = 0.05 and dof = 37540 is: 1.9600271797030
413
```

The t-test for claim amount shows that the presence of kids does not affect the claim amount

In [ ]: