In [2]:

```
import pandas as pd
import numpy as np
```

In [4]:

```
pd.set_option("display.max_columns",1000)
```

1. school - student's school (binary: "GP" - Gabriel Pereira or "MS" - Mousinho da Silveira)
2. sex - student's sex (binary: "F" - female or "M" - male)
3. age - student's age (numeric: from 15 to 22)
4. address_type - student's home address type (binary: "Urban" or "Rural")
5. family_size - family size (binary: "Less or equal to 3" or "Greater than 3")
6. parent_status - parent's cohabitation status (binary: "Living together" or "Apart")
7. mother_education - mother's education (ordinal: "none", "primary education (4th grade)", "5th to 9th grade", "secondary education" or "higher education")
8. father_education - father's education (ordinal: "none", "primary education (4th grade)", "5th to 9th grade", "secondary education" or "higher education")
9. mother_job - mother's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")
10. father_job - father's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")
11. reason - reason to choose this school (nominal: close to "home", school "reputation", "course" preference or "other")
12. guardian - student's guardian (nominal: "mother", "father" or "other")
13. travel_time - home to school travel time (ordinal: "<15 min.", "15 to 30 min.", "30 min. to 1 hour", or 4 - ">1 hour")
14. study_time - weekly study time (ordinal: 1 - "<2 hours", "2 to 5 hours", "5 to 10 hours", or ">10 hours")
15. class_failures - number of past class failures (numeric: n if 1<=n<3, else 4)
16. school_support - extra educational support (binary: yes or no)
17. family_support - family educational support (binary: yes or no)
18. extra_paid_classes - extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)
19. activities - extra-curricular activities (binary: yes or no)
20. nursery - attended nursery school (binary: yes or no)
21. higher_ed - wants to take higher education (binary: yes or no)
22. internet - Internet access at home (binary: yes or no)
23. romantic_relationship - with a romantic relationship (binary: yes or no)
24. family_relationship - quality of family relationships (numeric: from 1 - very bad to 5 - excellent)
25. free_time - free time after school (numeric: from 1 - very low to 5 - very high)
26. social - going out with friends (numeric: from 1 - very low to 5 - very high)
27. weekday_alcohol - workday alcohol consumption (numeric: from 1 - very low to 5 - very high)
28. weekend_alcohol - weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)
29. health - current health status (numeric: from 1 - very bad to 5 - very good)
30. absences - number of school absences (numeric: from 0 to 93)

These grades are related with the course subject, Math or Portuguese:

1. grade_1 - first period grade (numeric: from 0 to 20)
2. grade_2 - second period grade (numeric: from 0 to 20)
3. final_grade - final grade (numeric: from 0 to 20, output target)

In [223]:

```
df = pd.read_csv('student_math_clean.csv')
df.head()
```

Out[223]:

	student_id	school	sex	age	address_type	family_size	parent_status	mother_education	father_education	mother_job	fa
0	1	GP	F	18	Urban	Greater than 3	Apart	higher education	higher education	at_home	
1	2	GP	F	17	Urban	Greater than 3	Living together	primary education (4th grade)	primary education (4th grade)	at_home	
2	3	GP	F	15	Urban	Less than or equal to 3	Living together	primary education (4th grade)	primary education (4th grade)	at_home	
3	4	GP	F	15	Urban	Greater than 3	Living together	higher education	5th to 9th grade	health	
4	5	GP	F	16	Urban	Greater than 3	Living together	secondary education	secondary education	other	

In [ ]:

```
df.describe
```

dataset analysis

In [26]:

```
def dataset_analysis(data):
    print(f"columns in Datasets : {data.columns}\n")
    print(f"datatypes :: {data.dtypes}\n")
    print(f"null values {data.isna().sum()}")
    print(f"over look \n{data.describe()}")
```

In [27]:

```
dataset_analysis(df)
```

```
columns in Datasets : Index(['student_id', 'school', 'sex', 'age', 'address_type', 'family_size',
       'parent_status', 'mother_education', 'father_education', 'mother_job',
       'father_job', 'school_choice_reason', 'guardian', 'travel_time',
       'study_time', 'class_failures', 'school_support', 'family_support',
       'extra_paid_classes', 'activities', 'nursery_school', 'higher_ed',
       'internet_access', 'romantic_relationship', 'family_relationship',
       'free_time', 'social', 'weekday_alcohol', 'weekend_alcohol', 'health',
       'absences', 'grade_1', 'grade_2', 'final_grade'],
      dtype='object')

datatypes :: student_id                int64
school                    object
sex                       object
age                        int64
address_type              object
family_size               object
parent_status             object
mother_education          object
father_education          object
```

univariant

univariant for object type

In [224]:

```
ob_columns = df.select_dtypes(include=['object']).columns
print(ob_columns)
```

```
Index(['school', 'sex', 'address_type', 'family_size', 'parent_status',
       'mother_education', 'father_education', 'mother_job', 'father_job',
       'school_choice_reason', 'guardian', 'travel_time', 'study_time',
       'school_support', 'family_support', 'extra_paid_classes', 'activities',
       'nursery_school', 'higher_ed', 'internet_access',
       'romantic_relationship'],
      dtype='object')
```

In [225]:

```
student_info = ['school','sex','address_type','school_choice_reason']
famliy_info = ['family_size','parent_status','mother_education','father_education','mother_job',
              'father_job','guardian','family_support']
student_personal_info = ['travel_time','study_time','school_support','extra_paid_classes',
                        'activities','nursery_school','higher_ed','internet_access','romantic_relatio
```

In [226]:

```
import matplotlib.pyplot as plt
%matplotlib inline
```

lable_encoder for all applicable data

In [227]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [228]:

```
def lable_encoder(listof_col_data,dataFrame=df,encoder = LabelEncoder()):
    for col in listof_col_data:
        dataFrame[col] = encoder.fit_transform(dataFrame[col])
```

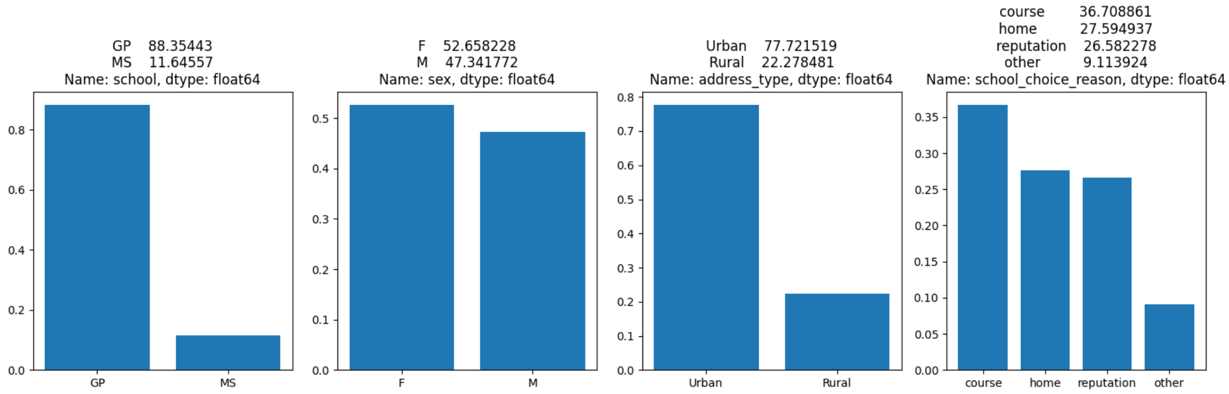
In [229]:

```
def univarinat_obj_anlysis(listOfData):
    plt.figure(figsize=(10,7))
    fig,ax = plt.subplots(1,len(listOfData),figsize=(15,5))
    for i,col in enumerate(listOfData):
        plt.subplot(1,len(listOfData),i+1)
        counts = df[col].value_counts(normalize=True)
        ax[i].bar(counts.index,counts)
        ax[i].set_title(f"{counts*100}")
    plt.tight_layout()
    plt.show()
```

In [230]:

```
univarinat_obj_anlysis(student_info)
```

```
<Figure size 1000x700 with 0 Axes>
```



Analysis:

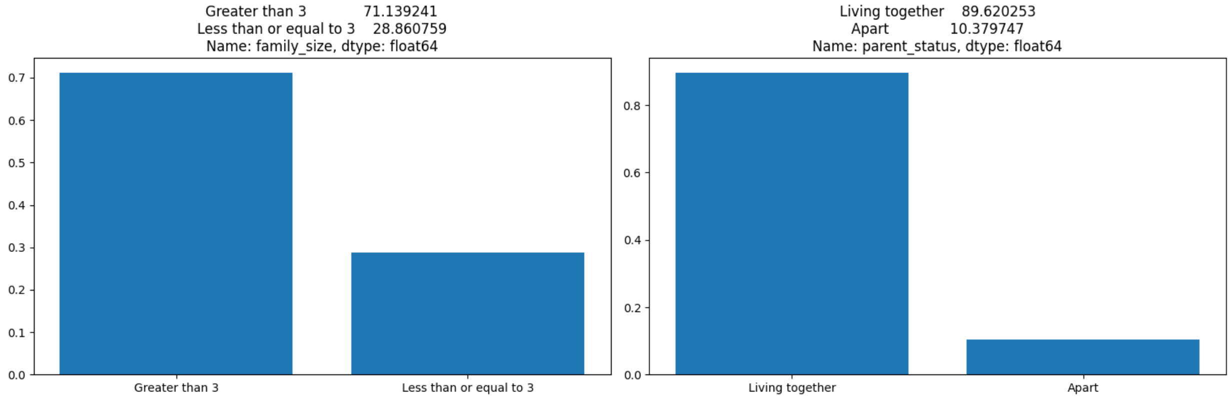
- lable encoding can be done for all there of data

In [231]:

```
lable_encoder(student_info)
```

```
In [232]: univarinat_obj_anlysis(famliy_info[:2])
```

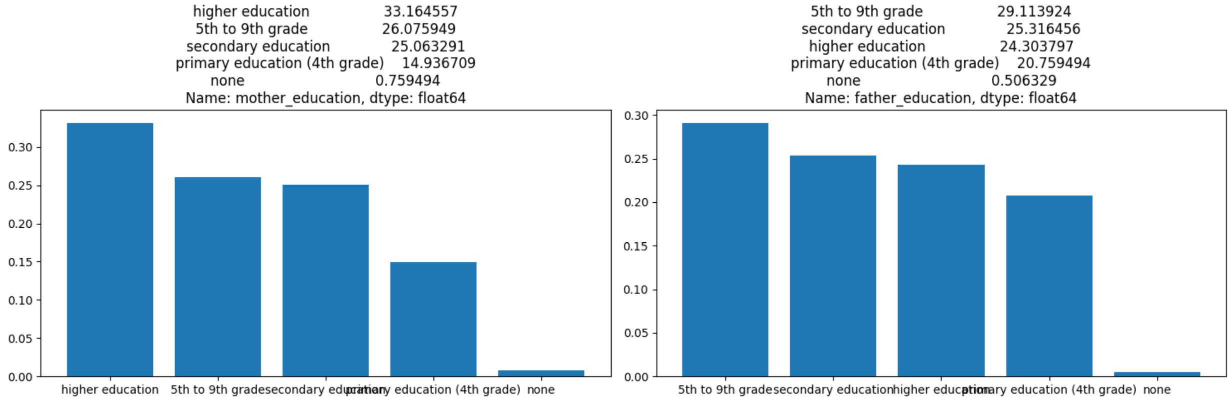
```
<Figure size 1000x700 with 0 Axes>
```



```
In [233]: lable_encoder(famliy_info[:2])
```

```
In [234]: univarinat_obj_anlysis(famliy_info[2:4])
```

```
<Figure size 1000x700 with 0 Axes>
```

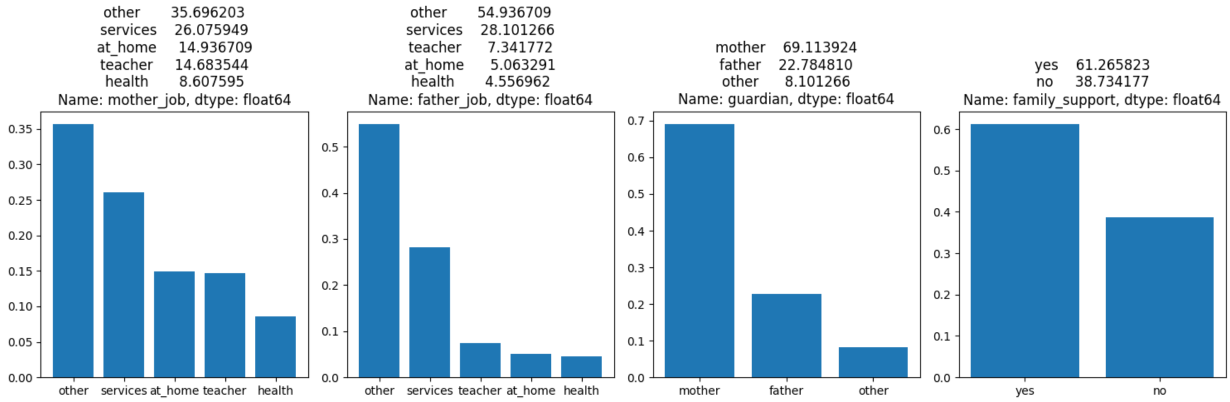


analysis:

- there are some none values in father and mother education, and it's is very less
- after dealing with null value we can lable encode them

```
In [235]: univarinat_obj_anlysis(famliy_info[4:8])
```

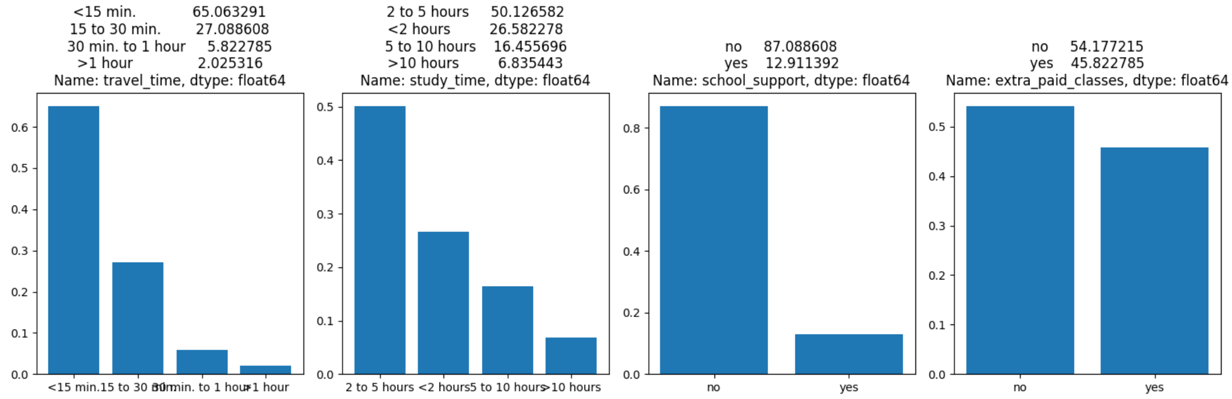
```
<Figure size 1000x700 with 0 Axes>
```



In [236]:
```
lable_encoder(famliy_info[4:8])
```

In [237]:
```
student_personal_info

univarinat_obj_anlysis(student_personal_info[:len(student_personal_info)//2])
```

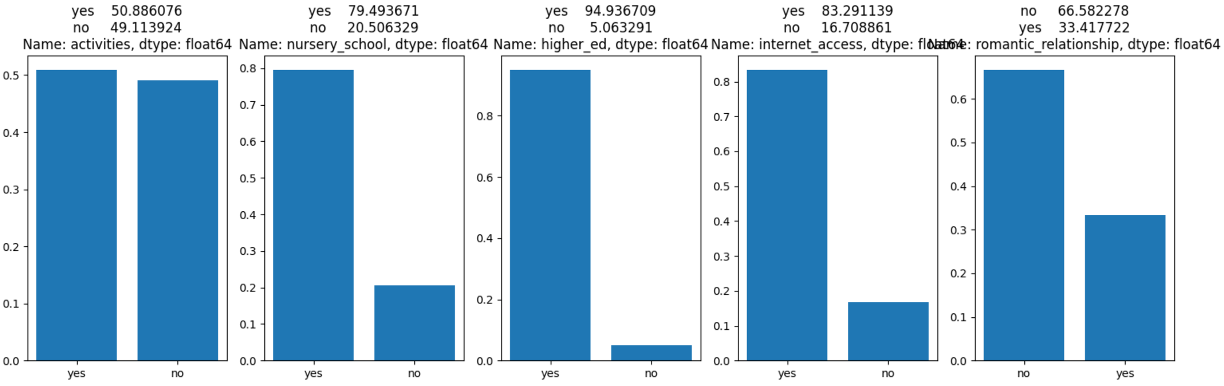
```
<Figure size 1000x700 with 0 Axes>
```



In [238]:
```
lable_encoder(student_personal_info[:len(student_personal_info)//2])
```

In [239]:
```
univarinat_obj_anlysis(student_personal_info[len(student_personal_info)//2:len(student_personal_info)
```

```
<Figure size 1000x700 with 0 Axes>
```



In [240]:
```
lable_encoder(student_personal_info[len(student_personal_info)//2:len(student_personal_info)])
```

analysis

- we can label encode all of them

univariant numeric anaysis

In [241]:
```
num_values = df.select_dtypes(include=['int64']).columns
num_values
```

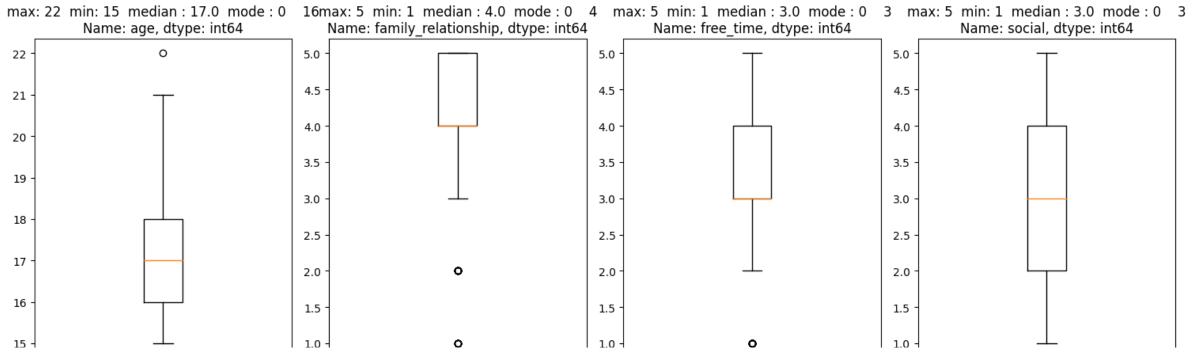
Out[241]:
```
Index(['student_id', 'age', 'class_failures', 'family_relationship',
       'free_time', 'social', 'weekday_alcohol', 'weekend_alcohol', 'health',
       'absences', 'grade_1', 'grade_2', 'final_grade'],
      dtype='object')
```

In [242]:
```python
def univarinat_num_anlysis(listOfData):

    plt.figure(figsize=(10,7))
    fig,ax = plt.subplots(1,len(listOfData),figsize=(15,5))

    for i,col in enumerate(listOfData):
        print(f"null values for {col} ::  {df[col].isna().sum()}")
        plt.subplot(1,len(listOfData),i+1)
        ax[i].boxplot(df[col])
        ax[i].set_title(f"max: {df[col].max()}  min: {df[col].min()}  median : {df[col].median()}  mo
    plt.tight_layout()
    plt.show()
```

In [243]:
```python
student_info = ['age','family_relationship','free_time','social','weekday_alcohol','weekend_alcohol',
grade_info = ['class_failures','grade_1','grade_2','final_grade']
```

In [244]:
```python
univarinat_num_anlysis(student_info[:len(student_info)//2])
```

```
null values for age ::  0
null values for family_relationship ::  0
null values for free_time ::  0
null values for social ::  0

<Figure size 1000x700 with 0 Axes>
```



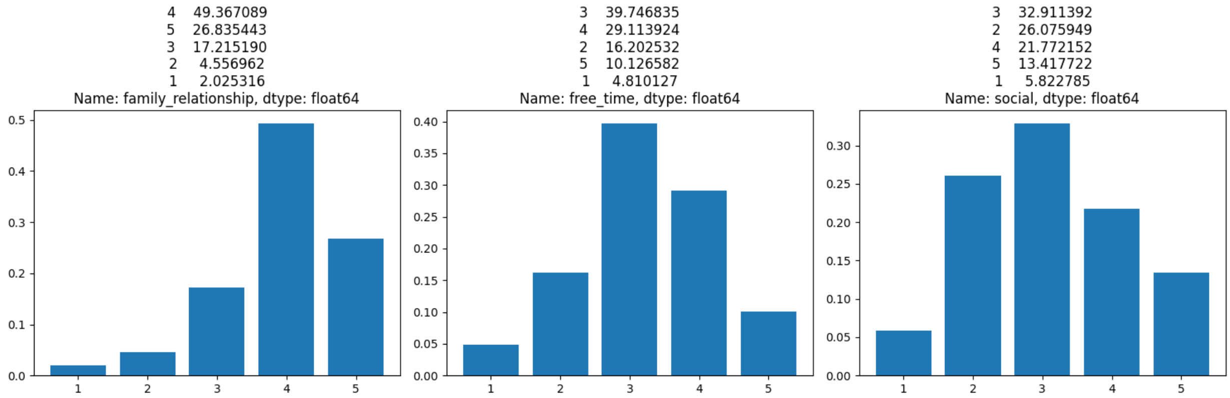
anlysis

- there are some outlier's in the family relation ship and free time
- free time , social , family relationship can be treted as categorical variable

In [245]:
```python
info = ['family_relationship','free_time','social','weekday_alcohol','weekend_alcohol','health']
```

In [246]:
```python
univarinat_obj_anlysis(info[:len(info)//2])
```

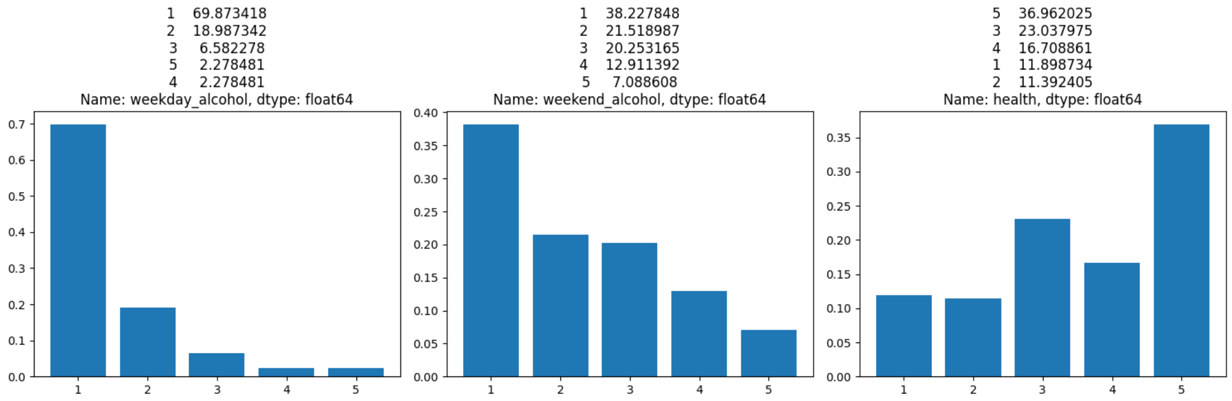
```
<Figure size 1000x700 with 0 Axes>
```



In [247]:

```
univarinat_obj_anlysis(info[len(info)//2:])
```

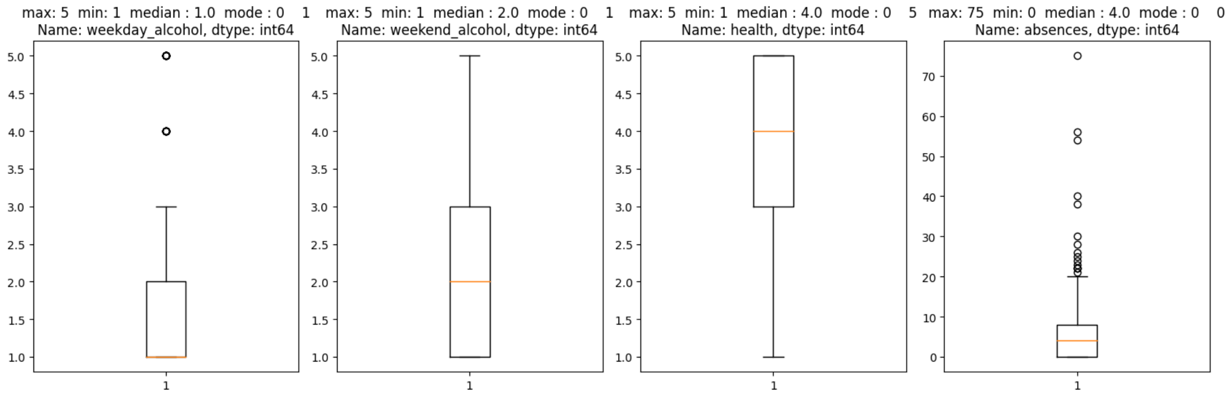
```
<Figure size 1000x700 with 0 Axes>
```



In [248]:

```
univarinat_num_anlysis(student_info[len(student_info)//2:len(student_info)])
```

```
null values for weekday_alcohol ::  0
null values for weekend_alcohol ::  0
null values for health ::  0
null values for absences ::  0

<Figure size 1000x700 with 0 Axes>
```

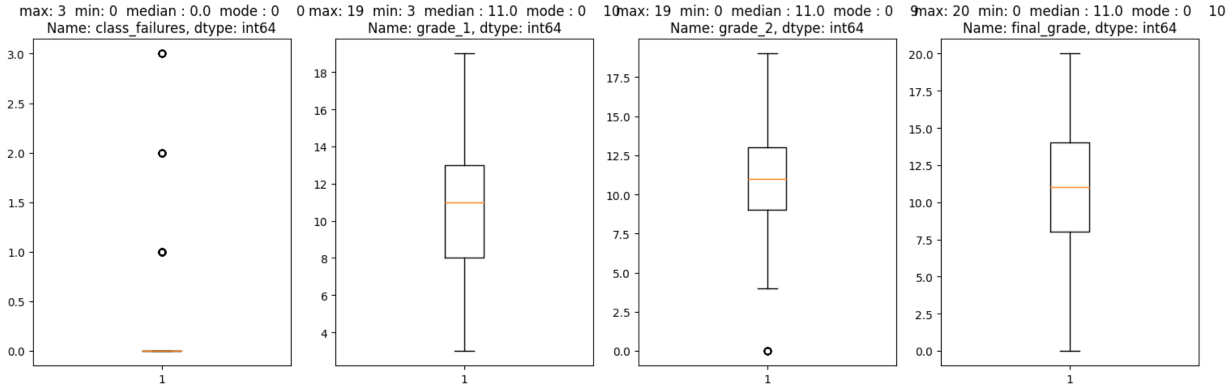


- absense is having lots of outliers

In [249]:

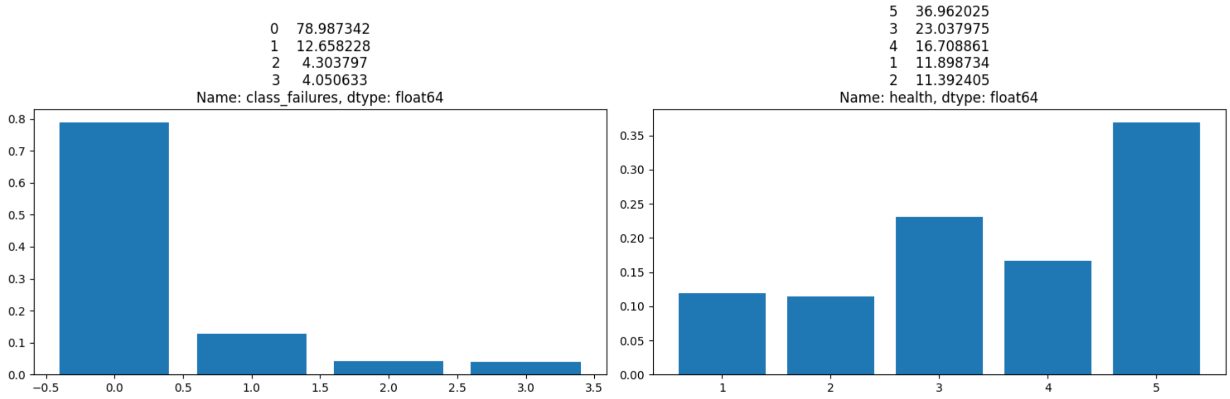
```
univarinat_num_anlysis(grade_info)
```

```
null values for class_failures ::  0
null values for grade_1 ::  0
null values for grade_2 ::  0
null values for final_grade ::  0

<Figure size 1000x700 with 0 Axes>
```



In [250]:
```
univarinat_obj_anlysis(['class_failures','health'])
```

```
<Figure size 1000x700 with 0 Axes>
```



In [252]:
```
df.loc[df['father_education'] == 'none']
```

Out[252]:

	student_id	school	sex	age	address_type	family_size	parent_status	mother_education	father_education	mother_job
76	77	0	1	15	1	0	1	higher education	none	4
171	172	0	1	16	1	0	1	primary education (4th grade)	none	2

In [253]:
```
df.loc[df['mother_education'] == 'none']
```

Out[253]:

	student_id	school	sex	age	address_type	family_size	parent_status	mother_education	father_education	mother_job
127	128	0	0	19	1	0	1	none	primary education (4th grade)	0
249	250	0	1	16	1	0	1	none	5th to 9th grade	2
324	325	0	0	17	1	1	1	none	5th to 9th grade	0

In [254]:
```
df.loc[df['father_education'] == 'none', 'father_education'] = df['father_education'].mode()[0]
df.loc[df['mother_education'] == 'none', 'mother_education'] = df['mother_education'].mode()[0]
```

In [256]:
```
df['father_education'].value_counts(), df['mother_education'].value_counts()
```

Out[256]:
```
(5th to 9th grade                117
 secondary education             100
 higher education                 96
 primary education (4th grade)    82
 Name: father_education, dtype: int64,
 higher education                134
 5th to 9th grade                103
 secondary education              99
 primary education (4th grade)    59
 Name: mother_education, dtype: int64)
```

In [257]:
```
lable_encoder(['father_education','mother_education'])
```

Bivariate Analysis

In [258]:

```
df.head(2)
```

Out[258]:

	student_id	school	sex	age	address_type	family_size	parent_status	mother_education	father_education	mother_job	fa
0	1	0	0	18	1	0	0	1	1	0	
1	2	0	0	17	1	0	1	2	2	0	

In [263]:

```
df.drop(columns=['student_id'],inplace=True)
```

In [264]:

```
df.dtypes
```

Out[264]:

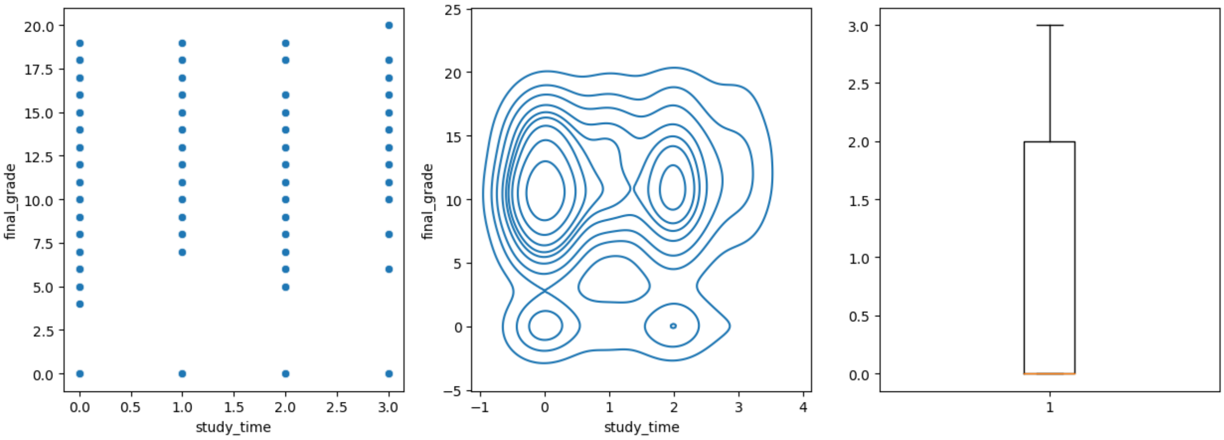
```
school                   int32
sex                      int32
age                      int64
address_type             int32
family_size              int32
parent_status            int32
mother_education         int32
father_education         int32
mother_job               int32
father_job               int32
school_choice_reason     int32
guardian                 int32
travel_time              int32
study_time               int32
class_failures           int64
school_support           int32
family_support           int32
extra_paid_classes       int32
activities               int32
nursery_school           int32
higher_ed                int32
internet_access          int32
romantic_relationship    int32
family_relationship      int64
free_time                int64
social                   int64
weekday_alcohol          int64
weekend_alcohol          int64
health                   int64
absences                 int64
grade_1                  int64
grade_2                  int64
final_grade              int64
dtype: object
```

Correlation between Study Time and Final Grade:

In [320]:
```
def bivariate_analysis(data1, data2):
    plt.figure(figsize=(15, 5))

    # Scatter Plot
    plt.subplot(1, 3, 1)
    sns.scatterplot(data=df, x=data1, y=data2)

    # Kernel Density Plot
    plt.subplot(1, 3, 2)
    sns.kdeplot(data=df, x=data1, y=data2)

    # Box Plot for data1
    plt.subplot(1, 3, 3)
    plt.boxplot(x=df[data1])


    # T-test
    stat, p_value = stats.ttest_ind(df[data1], df[data2])
    print(f"p value: {p_value}\n")

    # correlation

#     print(f"corr::  {df.corr()[data1][data2]}")
    print(f"corr: {df.corr().abs()[data1][data2]}")
    # Null hypothesis testing
    if p_value > 0.05:
        print("Null hypothesis accepted")
    else:
        print("Null hypothesis rejected")
```

In [318]:
```
bivariate_analysis('study_time','final_grade')
```

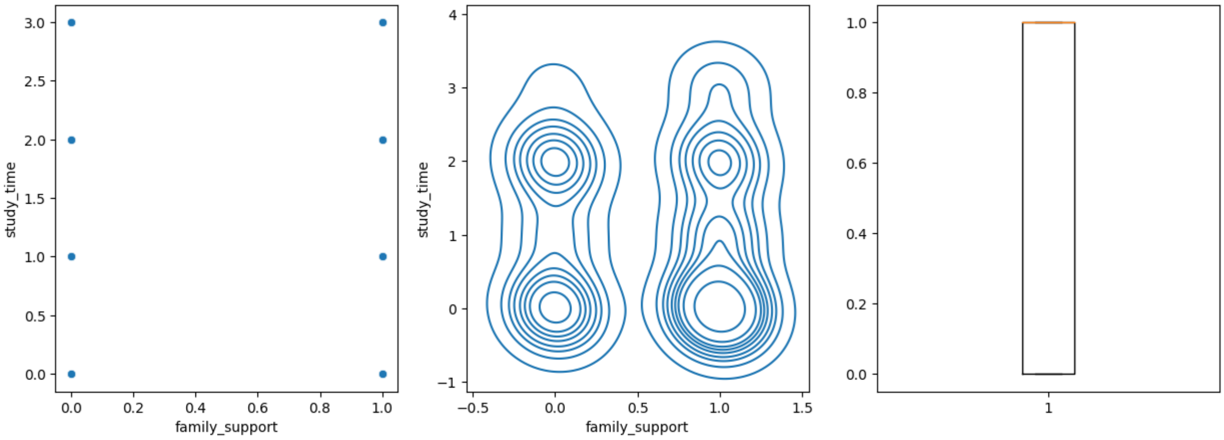
```
p value: 1.4208745105274756e-193

corr: 0.030078216123238653
Null hypothesis rejected
```



Impact of Family Support on Study Time:

```
In [321]: bivariate_analysis('family_support','study_time')
```

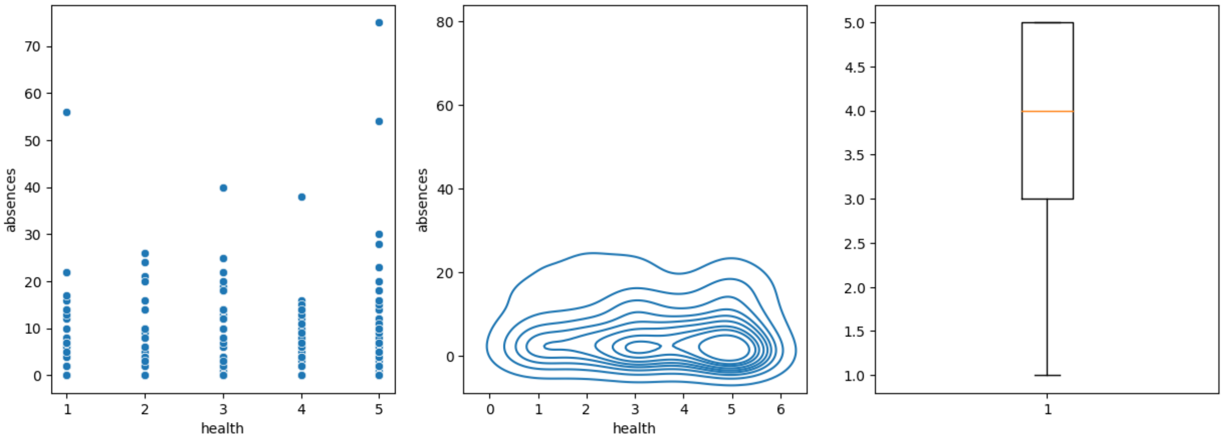
```
p value: 4.5439506599690837e-07

corr: 0.06708869386260417
Null hypothesis rejected
```



Comparison of Health and Absences:

```
In [322]: bivariate_analysis('health','absences')
```

```
p value: 1.750681468912281e-07

corr: 0.02993671093168928
Null hypothesis rejected
```

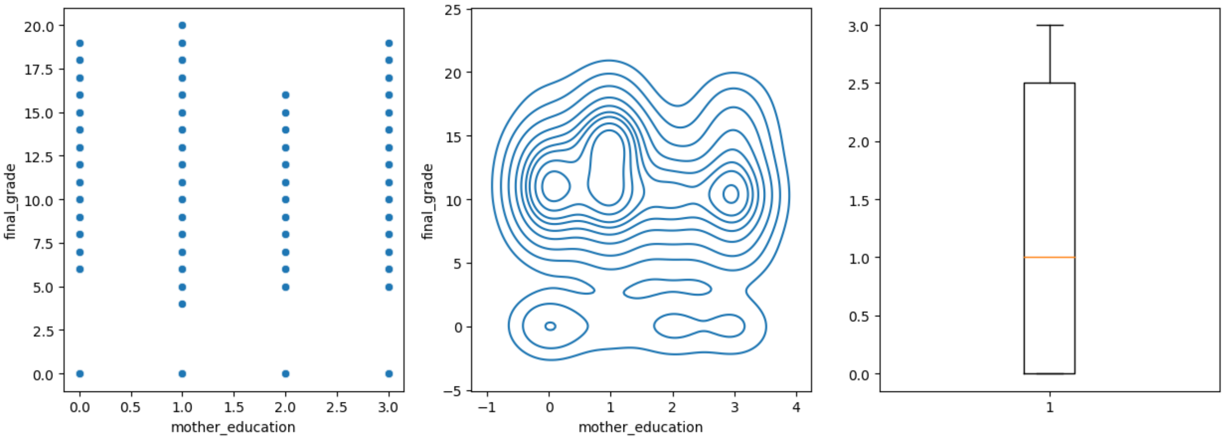


Influence of Mother's and Father's Education on Final Grade:

In [323]:

```
bivariate_analysis('mother_education','final_grade')
```

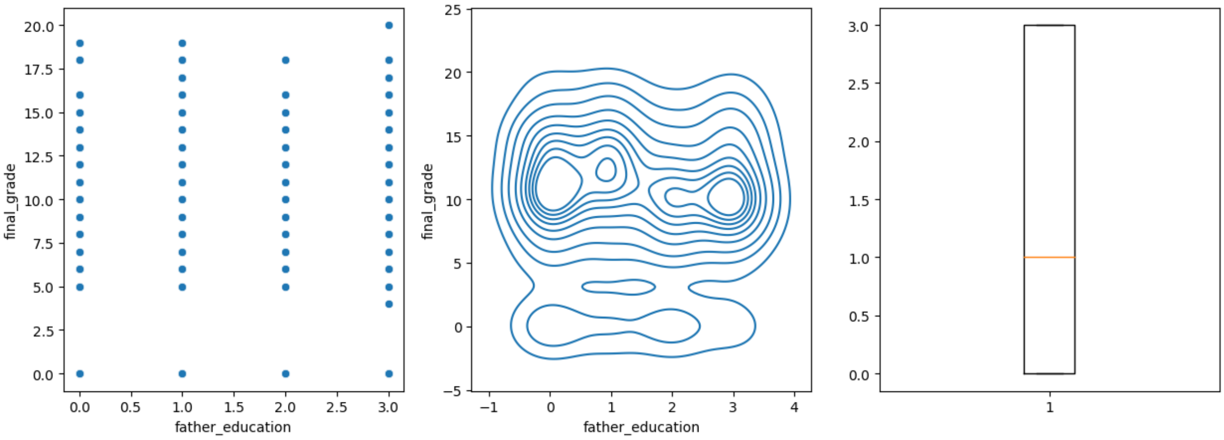
```
p value: 1.7845045099219184e-180

corr: 0.026580151808584664
Null hypothesis rejected
```



In [324]:

```
bivariate_analysis('father_education','final_grade')
```

```
p value: 2.186926991726661e-179

corr: 0.019824165456563153
Null hypothesis rejected
```

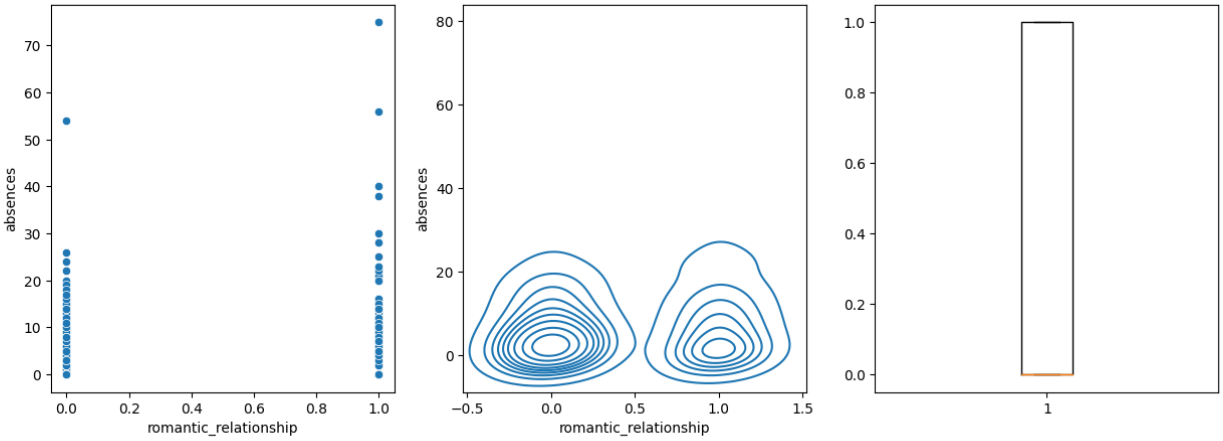


Impact of Romantic Relationship on Absences:

In [325]:

```python
bivariate_analysis('romantic_relationship','absences')
```

```
p value: 1.1274953085161249e-36

corr: 0.15338449094534373
Null hypothesis rejected
```

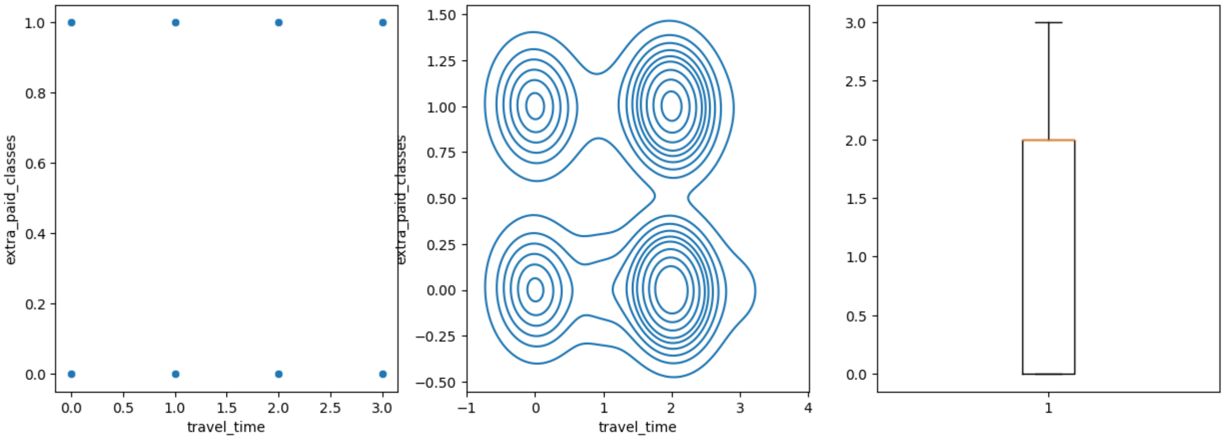


Association between Travel Time and Extra Paid Classes:

In [326]:

```python
bivariate_analysis('travel_time','extra_paid_classes')
```

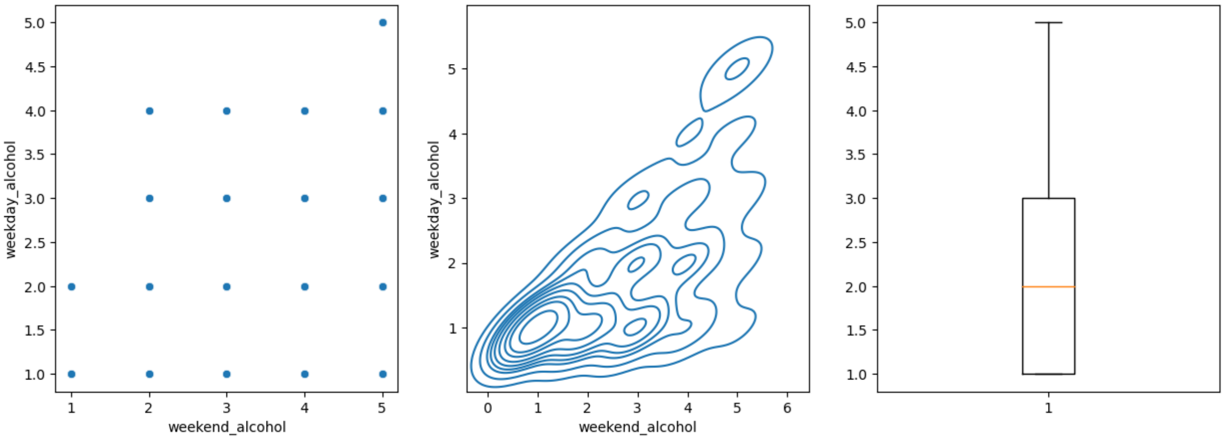
```
p value: 2.6043350536989204e-63

corr: 0.03950461725269721
Null hypothesis rejected
```



Comparison of Weekday and Weekend Alcohol Consumption:

In [327]:
```
bivariate_analysis('weekend_alcohol','weekday_alcohol')
```

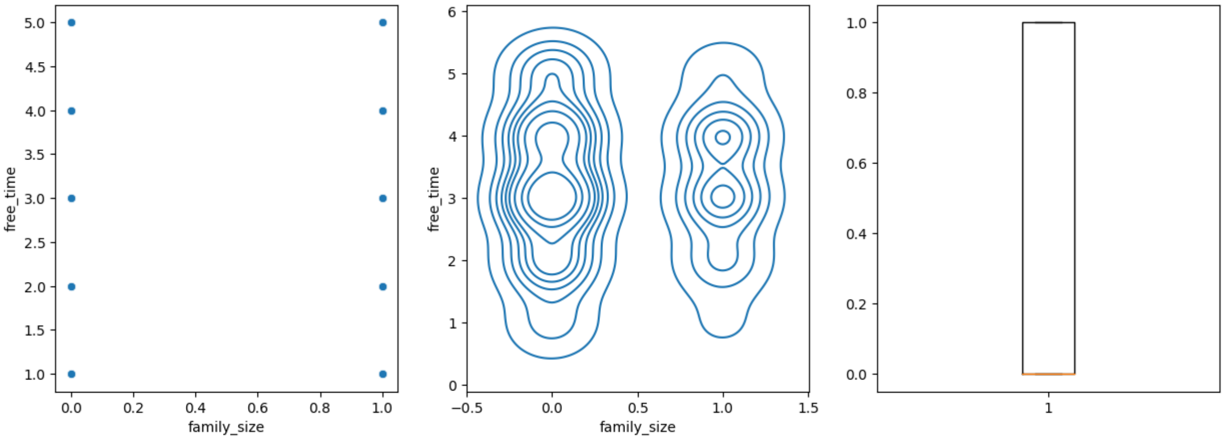
```
p value: 2.3476442547020824e-23

corr: 0.6475442300180093
Null hypothesis rejected
```



Effect of Family Size on Free Time:

In [328]:
```
bivariate_analysis('family_size','free_time')
```

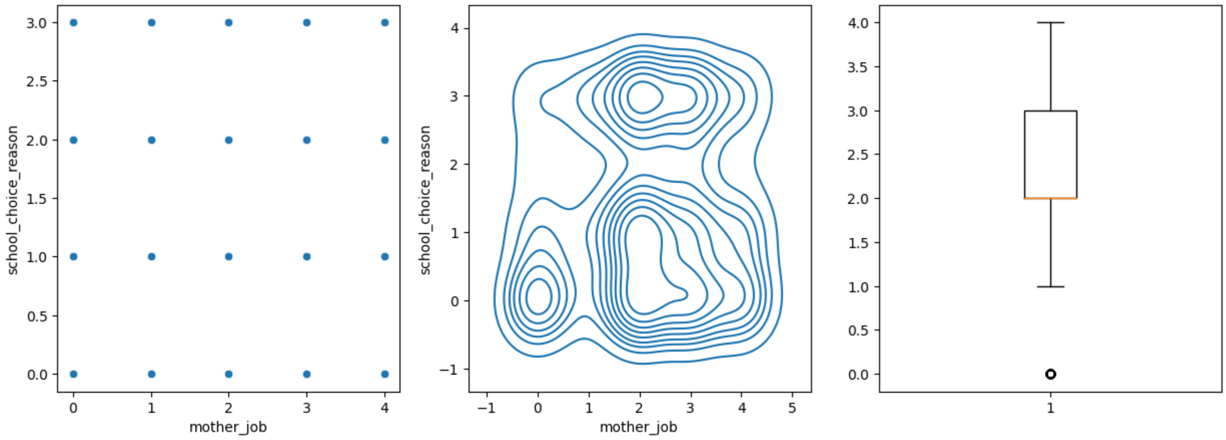
```
p value: 5.740484325393483e-264

corr: 0.017695255277930748
Null hypothesis rejected
```



Relationship between Parent's Job and School Choice Reason:

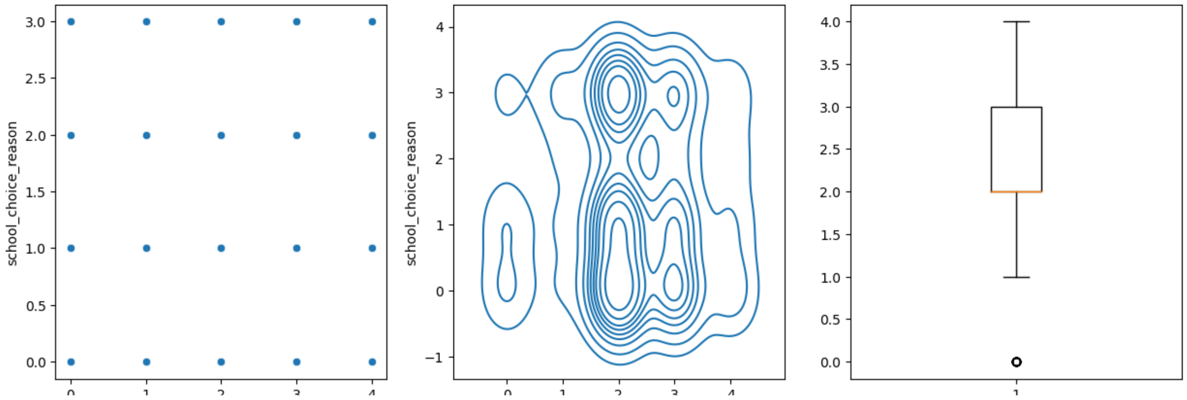
```
In [329]: bivariate_analysis('mother_job','school_choice_reason')
```

```
p value: 2.052894920607027e-24

corr: 0.022022327743222322
Null hypothesis rejected
```



```
In [330]: bivariate_analysis('father_job','school_choice_reason')
```

```
p value: 1.4064542534929025e-38

corr: 0.027688397220506663
Null hypothesis rejected
```



Further Steps:

Consider exploring additional relationships between variables.

Pay attention to outliers and consider whether they should be addressed in further analysis.

Check for multicollinearity among predictor variables.