Проект: Анализ резюме из HeadHunter

In [1]:
```
# импортируем библиотеки
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px

# настроим вывод графиков
import plotly.io as pio
pio.renderers.default='notebook'
```

Исследование структуры данных

1. Читаем данные

In [2]:
```
hh_data = pd.read_csv('dst-3.0_16_1_hh_database.csv', sep=';')
```

2. Вывод первых строк

In [5]:
```
hh_data.head(5)
```

Out[5]:

	Пол, возраст	ЗП	Ищет работу на должность:	Город, переезд, командировки	Занятость	График	Опыт работы
0	Мужчина , 39 лет , родился 27 ноября 1979	29000 руб.	Системный администратор	Советск (Калининградская область) , не готов к...	частичная занятость, проектная работа, полная ...	гибкий график, полный день, сменный график, ва...	Опыт работы 16 лет 10 месяцев Август 2010 — п...
1	Мужчина , 60 лет , родился 20 марта 1959	40000 руб.	Технический писатель	Королев , не готов к переезду , готов к редким...	частичная занятость, проектная работа, полная ...	гибкий график, полный день, сменный график, уд...	Опыт работы 19 лет 5 месяцев Январь 2000 — по...
2	Женщина , 36 лет , родилась 12 августа 1982	20000 руб.	Оператор	Тверь , не готова к переезду , не готова к ком...	полная занятость	полный день	Опыт работы 10 лет 3 месяца Октябрь 2004 — Де...
3	Мужчина , 38 лет , родился 25 июня 1980	100000 руб.	Веб-разработчик (HTML / CSS / JS / PHP / базы ...	Саратов , не готов к переезду , готов к редким...	частичная занятость, проектная работа, полная ...	гибкий график, удаленная работа	Опыт работы 18 лет 9 месяцев Август 2017 — Ап...
4	Женщина , 26 лет , родилась 3 марта 1993	140000 руб.	Региональный менеджер по продажам	Москва , не готова к переезду , готова к коман...	полная занятость	полный день	Опыт работы 5 лет 7 месяцев Региональный мене...

3. Основная информация о числе непустых значений в столбцах и их типах в таблице.

In [6]:
```
hh_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44744 entries, 0 to 44743
Data columns (total 12 columns):
 #   Column                            Non-Null Count  Dtype 
---  ------                            --------------  ----- 
 0   Пол, возраст                      44744 non-null  object
 1   ЗП                                44744 non-null  object
 2   Ищет работу на должность:         44744 non-null  object
 3   Город, переезд, командировки      44744 non-null  object
 4   Занятость                         44744 non-null  object
 5   График                            44744 non-null  object
 6   Опыт работы                       44576 non-null  object
 7   Последнее/нынешнее место работы   44743 non-null  object
 8   Последняя/нынешняя должность      44742 non-null  object
 9   Образование и ВУЗ                 44744 non-null  object
 10  Обновление резюме                 44744 non-null  object
 11  Авто                              44744 non-null  object
dtypes: object(12)
memory usage: 4.1+ MB
```

4. Сумма непустых значений

In [7]:
```
hh_data.isnull().sum()
```

Out[7]:
```
Пол, возраст                        0
ЗП                                  0
Ищет работу на должность:           0
Город, переезд, командировки        0
Занятость                           0
График                              0
Опыт работы                       168
Последнее/нынешнее место работы     1
Последняя/нынешняя должность        2
Образование и ВУЗ                   0
Обновление резюме                   0
Авто                                0
dtype: int64
```

5. Основная статистическая информация о столбцах.

In [8]:
```
hh_data.describe(include='object')
```

Out[8]:

	Пол, возраст	ЗП	Ищет работу на должность:	Город, переезд, командировки	Занятость	График	Опыт работы	н
count	44744	44744	44744	44744	44744	44744	44576	
unique	16003	690	14929	10063	38	47	44413	
top	Мужчина , 32 года , родился 17 сентября 1986	50000 руб.	Системный администратор	Москва , не готов к переезду , не готов к кома...	полная занятость	полный день	Опыт работы 10 лет 8 месяцев Апрель 2018 — по...	предг
freq	18	4064	3099	1261	30026	22727	3	

6. Размерность таблицы

In [9]:
```
hh_data.shape
```

Out[9]:
```
(44744, 12)
```

Преобразование данных

1. Создадим с помощью функции-преобразования новый признак **"Образование"**, который будет иметь 4 категории: "высшее", "неоконченное высшее", "среднее специальное" и "среднее".

In [10]:
```
def education(x):
    x = x.split()
    if x[1] == 'образование':
        return ' '.join(x[:1])
    return ' '.join(x[:2])

hh_data['Образование'] = hh_data['Образование и ВУЗ'].apply(education)
hh_data['Образование'] = hh_data['Образование'].astype('category')
```

Удалим признак 'Образование и ВУЗ'

In [11]:
```
hh_data = hh_data.drop(['Образование и ВУЗ'], axis=1)
```

Сколько соискателей имеет средний уровень образования (школьное образование)?

In [12]:
```
hh_data['Образование'].value_counts()
```

Out[12]:
```
Высшее                   33863
Среднее специальное       5765
Неоконченное высшее       4557
Среднее                    559
Name: Образование, dtype: int64
```

2. Создадим два столбца **Пол**, **возраст** из столбца **"Пол, возраст"**

In [13]:
```
def gender(x):
    x = x.replace('  ', ' ')
    x = x.split(' , ')
    return str(x[0][0])

def age(x):
    x = x.replace('  ', ' ')
    return int(x.split(' , ')[1].split(' ')[0])

hh_data['Пол'] = hh_data['Пол, возраст'].apply(gender)
hh_data['Возраст'] = hh_data['Пол, возраст'].apply(age)
```

Удалим признак «Пол, возраст» из таблицы.

In [14]:
```
hh_data = hh_data.drop(['Пол, возраст'], axis=1)
```

Сколько процентов женских резюме представлено в наших данных?

In [15]:
```
round(len(hh_data[hh_data['Пол'] == 'Ж']) / len(hh_data) * 100, 2)
```

Out[15]:
```
19.07
```

In [16]:
```
hh_data.describe(include='object')
```

Out[16]:

	ЗП	Ищет работу на должность:	Город, переезд, командировки	Занятость	График	Опыт работы	Послед нынешнее м ра
count	44744	44744	44744	44744	44744	44576	4
unique	690	14929	10063	38	47	44413	3
top	50000 руб.	Системный администратор	Москва , не готов к переезду , не готов к кома...	полная занятость	полный день	Опыт работы 10 лет 8 месяцев Апрель 2018 — по...	Индивидуал предпринимател / частна
freq	4064	3099	1261	30026	22727	3	

Чему равен средний возраст соискателей?

In [17]:
```
round(hh_data['Возраст'].mean(), 1)
```

Out[17]: 32.2

3. Преобразуем признак **"Опыт работы"**. Его текущий формат - это: **<Опыт работы: n лет m месяцев, периоды работы в различных компаниях...>**.

выделим общий опыт работы соискателя в месяцах, новый признак назовем "Опыт работы (месяц)"

In [18]:
```
def get_experience(arg):
    if arg is np.nan or arg == 'Не указано':
        return None
    year_words=['год', 'года', 'лет']
    month_words=['месяц', 'месяца', 'месяцев']
    arg_splitted = arg.split(' ')[:7]
    years = 0
    months = 0
    for index, item in enumerate (arg_splitted):
        if item in year_words:
            years = int(arg_splitted[index-1])
        if item in month_words:
            months = int(arg_splitted[index-1])
    return int(years*12 + months)
hh_data['Опыт работы (месяц)'] = hh_data['Опыт работы'].apply(get_experience)
```

Медианный опыт работы (в месяцах)?

In [19]:
```
hh_data['Опыт работы (месяц)'].median()
```

Out[19]: 100.0

Удалим столбец «Опыт работы» из таблицы.

In [20]:
```
hh_data = hh_data.drop(['Опыт работы'], axis=1)
```

4. Создадим отдельные признаки **"Город"**, **"Готовность к переезду"**, **"Готовность к командировкам"**. При этом важно учесть:

- Признак **"Город"** должен содержать только 4 категории: "Москва", "Санкт-Петербург" и "город-миллионник", остальные обозначим как "другие".
- Признак **"Готовность к переезду"** должен иметь два возможных варианта: True или False.
- Признак **"Готовность к командировкам"** должен иметь два возможных варианта: True или False.

Город:

In [21]:
```
def city_f(x):
    million_cities = ['Новосибирск', 'Екатеринбург','Нижний Новгород','Казань',
```

```
    x = x.replace('  ', ' ').split(' , ')[0]
    if x == 'Москва':
        return 'Москва'
    if x == 'Санкт-Петербург':
        return 'Санкт-Петербург'
    if x in million_cities:
        return 'город-миллионник'
    return 'другие'

hh_data['Город'] = hh_data['Город, переезд, командировки'].apply(city_f)
hh_data['Город'] = hh_data['Город'].astype('category')
```

Готовность к переезду:

In [22]:
```
def removal_f(x):
    x = x.replace('  ', ' ').split(' , ')
    if x[1][0] == 'м':
        x = x[2]
    else:
        x = x[1]
    x = x.split(' ')
    for i in x:
        if i == 'не':
            return False
    return True

hh_data['Готовность к переезду'] = hh_data['Город, переезд, командировки'].apply
```

Готовность к командировкам:

In [23]:
```
def business_trips_f(x):
    if x is np.nan:
        return False
    x = x.replace('  ', ' ').split(' , ')
    if len(x) < 3 or x[1][0] == 'м' and len(x) <= 3:
        return False
    if x[1][0] == 'м':
        x = x[3]
    else:
        x = x[2]
    x = x.split(' ')

    for i in x:
        if i == 'не':
            return False
    return True

hh_data['Готовность к командировкам'] = hh_data['Город, переезд, командировки'].
```

Сколько процентов соискателей живут в Санкт-Петербурге?

In [24]:
```
hh_data[hh_data['Город'] == 'Санкт-Петербург'].shape[0] / hh_data.shape[0] * 100
```

Out[24]:
```
11.033881637761487
```

Сколько процентов соискателей готовы одновременно и к переездам, и к командировкам?

In [25]:
```
hh_data[(hh_data['Готовность к переезду'] == True) & (hh_data['Готовность к кома
```

Out[25]:
```
31.88136956910424
```

Удалим столбец "Город, переезд, командировки"

In [26]:
```
hh_data = hh_data.drop(['Город, переезд, командировки'], axis=1)
```

5. Рассмотрим поближе признаки **"Занятость"** и **"График"**. Сейчас признаки представляют собой набор категорий желаемой занятости (полная занятость, частичная занятость, проектная работа, волонтерство, стажировка) и желаемого графика работы (полный день, сменный график, гибкий график, удаленная работа, вахтовый метод).

Такой вариант признаков имеет множество различных комбинаций, а значит множество уникальных значений, что мешает анализу.

Преобразуем категориальные признаки в One Hot Encoding

In [27]:
```
employments = ['полная занятость', 'частичная занятость',
               'проектная работа', 'волонтерство', 'стажировка']
charts = ['полный день', 'сменный график',
          'гибкий график', 'удаленная работа',
          'вахтовый метод']
for employment, chart in zip(employments, charts):
    hh_data[employment] = hh_data['Занятость'].apply(lambda x: employment in x)
    hh_data[chart] = hh_data['График'].apply(lambda x: chart in x)
```

Сколько людей ищут проектную работу и волонтёрство?

In [28]:
```
hh_data[hh_data['проектная работа'] & hh_data['волонтерство']].shape[0]
```

Out[28]:
```
436
```

Сколько людей хотят работать вахтовым методом и с гибким графиком?

In [29]:
```
hh_data[hh_data['вахтовый метод'] & hh_data['гибкий график']].shape[0]
```

Out[29]:
```
2311
```

Удалим столбцы «Занятость» и «График»

In [30]:
```
hh_data = hh_data.drop(['Занятость','График'], axis=1)
```

6. Соискатель указывает зарплату в различный валютах, преобразуем заработную плату в единую валюту, например, в рублях. Возникает вопрос, а где взять курс валют по отношению к рублю?

Сделаем выгрузку курсов валют, которые встречаются в наших данных за период с 29.12.2017 по 05.12.2019. И сохраним в csv файл.

Создайте новый DataFrame из полученного файла. В полученной таблице будут столбцы:

- "currency" - наименование валюты в ISO кодировке,
- "date" - дата,
- "proportion" - пропорция,
- "close" - цена закрытия (последний зафиксированный курс валюты на указанный день).

В признаке **"Обновление резюме"**, содержится дата и время, когда соискатель выложил текущий вариант своего резюме. По дате и будем сопоставлять курсы валют.

Считаем данные курса из файла

In [31]:
```
exch_data = pd.read_csv('ExchangeRates.csv', sep=',')
```

Переведём признак "Обновление резюме" из таблицы с резюме в формат datetime и создадим столбец с датой. В тот же формат приведем признак "date" из таблицы с валютами.

In [32]:
```
hh_data['date'] = pd.to_datetime(hh_data['Обновление резюме'], dayfirst=True).dt
exch_data['date'] = pd.to_datetime(exch_data['date'], dayfirst=True).dt.date
```

Выделим из столбца "ЗП" сумму желаемой заработной платы и наименование валюты, в которой она исчисляется. Наименование валюты переведем в стандарт ISO.

In [33]:
```
def get_salary_sum(x): # получаем зп
    return float(x.split(' ')[0])


def get_salary_currency(arg): # получаем курс
    currency_dict = {
        'USD': 'USD', 'KZT': 'KZT',
        'грн': 'UAH', 'белруб': 'BYN',
        'EUR': 'EUR', 'KGS': 'KGS',
        'сум': 'UZS', 'AZN': 'AZN',
        'руб': 'RUB'
    }
    curr = arg.split(' ')[1].replace('.', '')
    return currency_dict[curr]

hh_data['ЗП (tmp)'] = hh_data['ЗП'].apply(get_salary_sum)
hh_data['Курс (tmp)'] = hh_data['ЗП'].apply(get_salary_currency)
```

Присоединим к таблице с резюме таблицу с курсами по столбцам с датой и названием валюты. Значение close для рубля заполним единицей (курс рубля самого к себе)

In [34]:
```
merged = hh_data.merge(
    exch_data,
```

```
    left_on=['Курс (tmp)', 'Обновление резюме'],
    right_on=['currency', 'date',],
    how='left'
)
merged['close'] = merged['close'].fillna(1) # заполним пропуски
merged['proportion'] = merged['proportion'].fillna(1)
```

Умножим сумму желаемой заработной платы на присоединенный курс валюты (close) и разделить на пропорцию.

In [35]:
```
hh_data['ЗП (руб)'] = merged['close'] * merged['ЗП (tmp)'] / merged['proportion'
```

Чему равна желаемая медианная заработная плата соискателей?

In [36]:
```
hh_data['ЗП (руб)'].median()/1000
```

Out[36]:
```
60.0
```

Удалим исходный столбец заработной платы "ЗП" и все промежуточные столбцы.

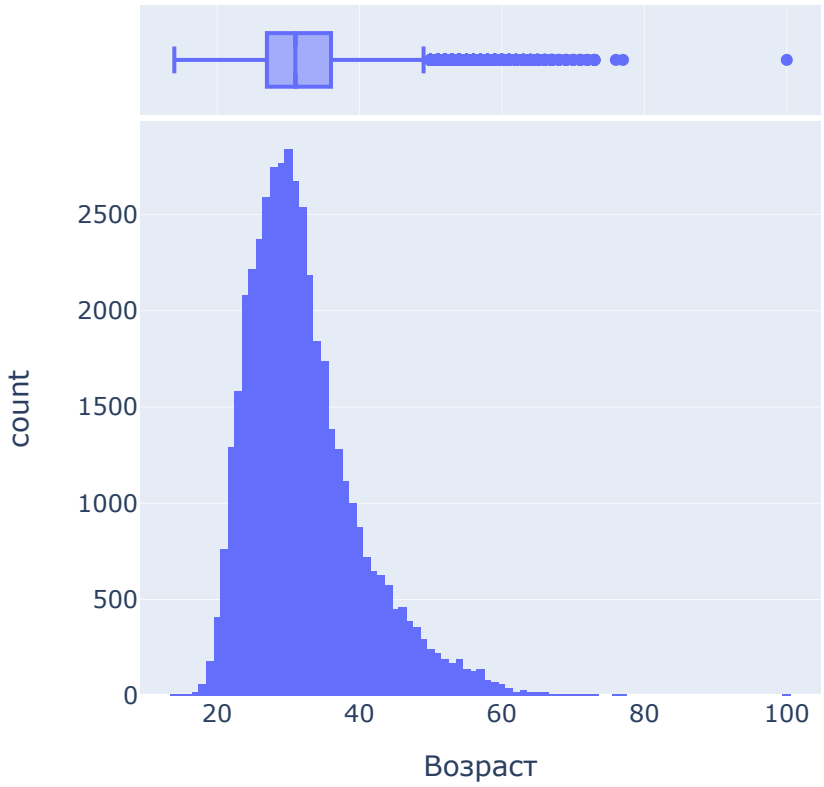
In [37]:
```
hh_data = hh_data.drop(['ЗП', 'ЗП (tmp)', 'Курс (tmp)','date'], axis=1)
```

Исследование зависимостей в данных

In [38]:
```
import plotly.express as px
import plotly.graph_objs as go
```

1. Построим распределение признака **"Возраст"**.

In [48]:
```
fig = px.histogram(
    data_frame=hh_data,
    x='Возраст',
    title='Распределение Возраст соискателей',
    width=500,
    marginal='box',
)
fig.show()
```

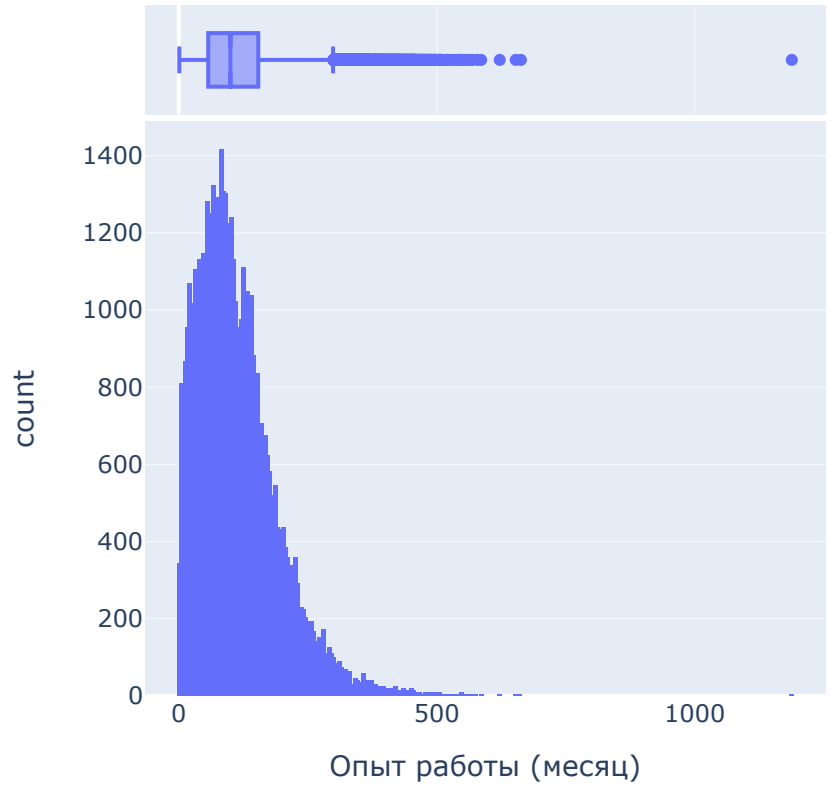


Вывод:

- среднее значение возраста - 30 лет.
- предельные значения - от 14 до 100, большинство соискателей от 21 до 41 года
- имеются аномалии - возраст 100 лет, 77 и 76

2. Построим распределение признака **"Опыт работы (месяц)"**.

In [49]:
```
fig = px.histogram(
    data_frame=hh_data,
    x='Опыт работы (месяц)',
    title='Распределение опыта работы соискателей',
    width=500,
    marginal='box',
)
fig.show()
```

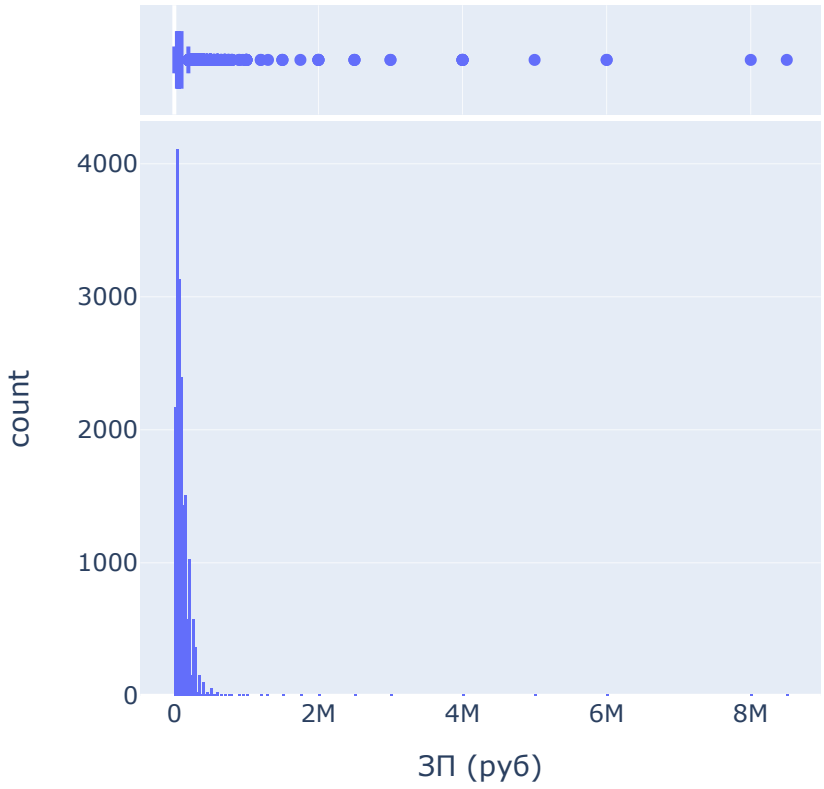


Вывод:

- среднее значение опыта - 100 месяцев(8,33 года).
- предельные значения - от 1 до 1188 месяцев, большинство соискателей имеют опыт от 5 до 160 месяцев
- имеются аномалии - 1188 месяцев(99 лет)

3. Построим распределение признака **"ЗП (руб)"**.

In [50]:
```
fig = px.histogram(
    data_frame=hh_data,
    x='ЗП (руб)',
    title='Распределение желаемой з/п соискателей',
    width=500,
    marginal='box'
)
fig.show()
```



Вывод:

- среднее значение зп - 60т рублей.
- предельные значения - от 1 до 8.5м рублей, большинство соискателей хотят от 36т до 100т рублей
- имеются аномалии - зп в размере 1 рубль и также зп от 1 миллиона, но зависит от должности

4. Построим диаграмму, которая показывает зависимость **медианной** желаемой заработной платы (**"ЗП (руб)"**) от уровня образования (**"Образование"**).

In [51]:
```
temp_data = hh_data[hh_data['ЗП (руб)']<1e6].groupby('Образование', as_index=Fal
fig = px.bar(
    data_frame=temp_data,
    x='Образование',
    y='ЗП (руб)',
    title='Медианная з/п по уровню образования'
)
fig.show()
```

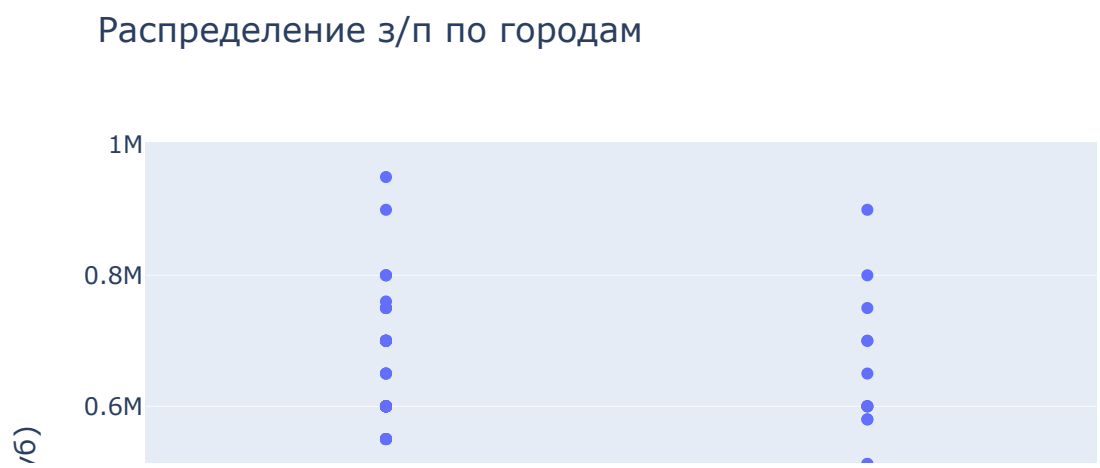
```
C:\Users\rustem\AppData\Local\Temp\ipykernel_13520\2951648698.py:1: FutureWarni
ng:

The default value of numeric_only in DataFrameGroupBy.median is deprecated. In
a future version, numeric_only will default to False. Either specify numeric_on
ly or select only columns which should be valid for the function.
```

Вывод: Можно сделать вывод, что от образования зависит зп, для высшего оно наибольшее, для среднего и среднего специального зп наименьшее

5. Построим диаграмму, которая показывает распределение желаемой заработной платы (**"ЗП (руб)"**) в зависимости от города (**"Город"**).

In [52]:
```
temp_data = hh_data[hh_data['ЗП (руб)']<1e6]
fig = px.box(
    data_frame=temp_data,
    x='Город',
    y='ЗП (руб)',
    title='Распределение з/п по городам'
)
fig.show()
```



Вывод: Можно заметить, что зп зависит от города, в городах-миллионниках зп меньше, по сравнению с Санкт-Петербургом и Москвой. В "Других" зп наибольшая, так как в этой категории все остальные города России.

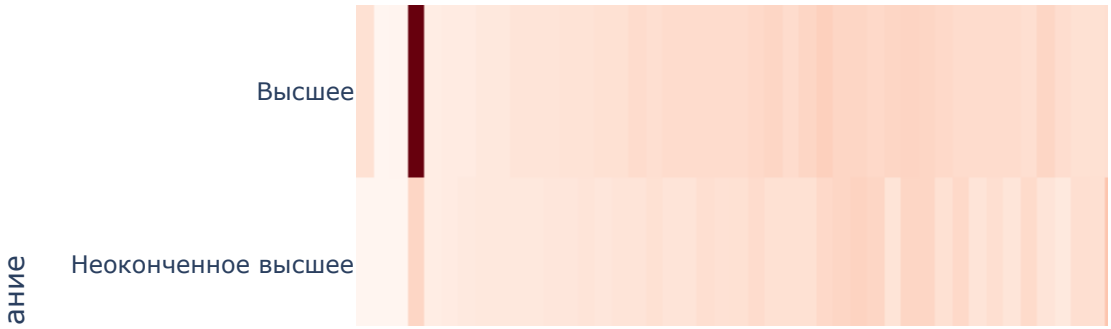
6. Построим график, которая показывает зависимость медианной заработной платы (**"ЗП (руб)"**) от признаков **"Готовность к переезду"** и **"Готовность к командировкам"**.

In [53]:
```
temp_data = hh_data.groupby(
    ['Готовность к командировкам', 'Готовность к переезду'],
    as_index=False
)['ЗП (руб)'].median()
fig = px.bar(
    data_frame=temp_data,
    y='Готовность к переезду',
    x='ЗП (руб)',
    barmode="group",
    color='Готовность к командировкам',
    title='Медианная з/п по готовности к командировкам/переезду'
)
fig.show()
```

Вывод: Уровень зп зависит от возможности командировок и переездов - чем больше вариаций, тем больше зарплата

7. Построим карту иллюстрирующую зависимость **медианной** желаемой заработной платы от возраста (**"Возраст"**) и образования (**"Образование"**).

In [54]:
```
temp_data = hh_data.pivot_table(
    index='Образование',
    columns='Возраст',
    values='ЗП (руб)',
    aggfunc='median',
    fill_value=0
)
fig = px.imshow(
    temp_data,
    aspect='auto',
    color_continuous_scale='reds',
    title='Медианная з/п по образованию и возрасту'
)
fig.show()
```



Вывод: Для высшего образования рост зп происходит быстрее всего и больше, чем в других категориях. Также неоконченное высшее цениться больше, чем среднее образование. Видим аномалию в зп - для высшего образования в возрасте 17 лет средняя зарплата 505т

Для большей наглядности можем убрать все аномалии - выберем зарплату, меньшую, чем 1000000 рублей

In [55]:
```
temp_data = hh_data[hh_data['ЗП (руб)'] < 1000000]
temp_data = temp_data.pivot_table(
    index='Образование',
    columns='Возраст',
    values='ЗП (руб)',
    aggfunc='median',
    fill_value=0
)
fig = px.imshow(
    temp_data,
    aspect='auto',
    color_continuous_scale='reds',
    title='Медианная з/п по образованию и возрасту',
)
fig.show()
```

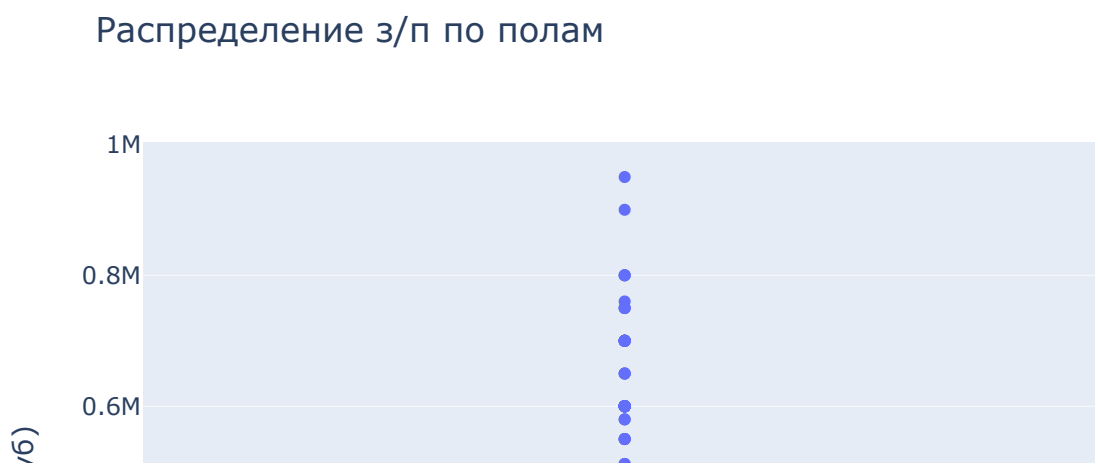
8. Построим график, показывающую зависимость опыта работы (**"Опыт работы (месяц)"**) от возраста (**"Возраст"**). Также построим прямую с координатами (0, 0) и (100, 100).Точки, лежащие на этой прямой и выше неё, — аномалии в наших данных (опыт работы больше либо равен возрасту соискателя).

In [56]:
```
x = [0, 100]
import seaborn as sns
temp_data = hh_data.copy()
temp_data['Опыт работы (год)'] = temp_data['Опыт работы (месяц)']/12
fig = px.scatter(
    temp_data,
    x='Возраст',
    y='Опыт работы (год)',
    title = 'Зависимость опыта работы от возраста')
fig.add_trace(go.Scatter(x=x, y=x))
fig.show();
```

Вывод: Из нашего графика можно сделать вывод, что в БД присутсвтуют 7 аномалий, связанные с оптытом работы

9. Построим диаграмму, показывающий зависимость между признаком (**"Пол"**) и зп(**"ЗП (руб)"**).

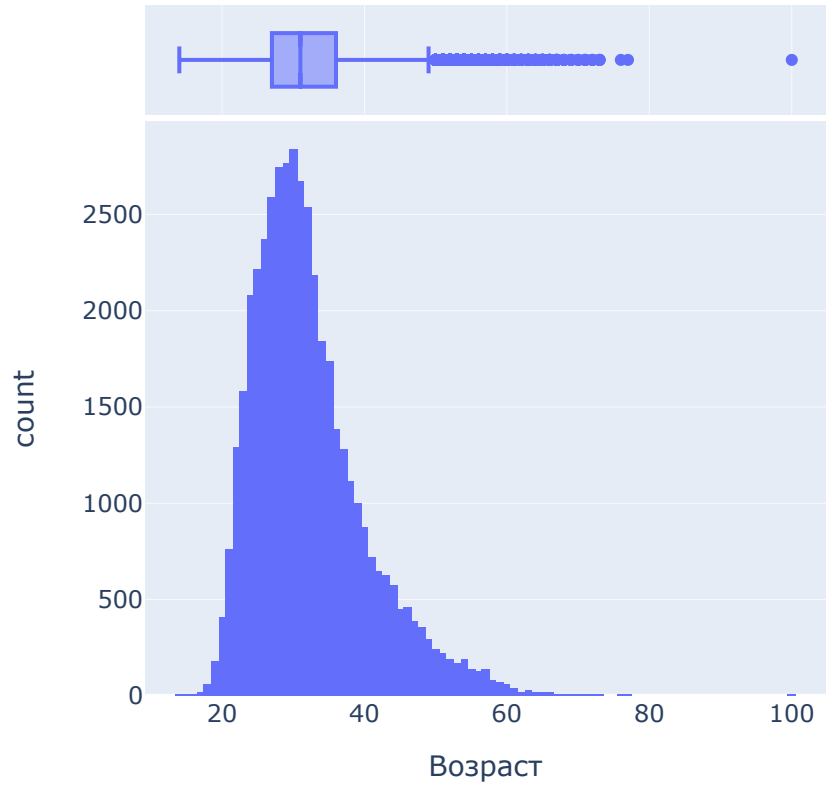
In [57]:
```
temp_data = hh_data[hh_data['ЗП (руб)']<1e6]
fig = px.box(
    data_frame=temp_data,
    x='Пол',
    y='ЗП (руб)',
    title='Распределение з/п по полам'
)
fig.show()
```



Вывод: Можно заметить, что мужчины в среднем зарабатывают больше, чем женщины, также зарплата у мужчин имеют больший размах

10. Построим диаграмму, которая показывает, какое кол-во людей ищут стажировку, в зависимоти от возраста человека

In [58]:
```
temp_data = hh_data[hh_data['стажировка'] == True]
fig = px.histogram(
    data_frame=hh_data,
    x='Возраст',
    title='Распределение стажировок по возрастам',
    width=500,
    marginal='box'
)
fig.show()
```



Вывод: С повышением возраста, кол-во людей, которые ищут стажировку уменьшаются

Очистка данных

1. Найдием полные дубликаты в таблице с резюме и удалим их.

In [59]:
```
duplicates = hh_data[hh_data.duplicated(subset=hh_data.columns)]
hh_data = hh_data.drop_duplicates()
duplicates.shape[0]
```

Out[59]: 158

2. Выведем информацию о числе пропусков в столбцах.

In [60]:
```
hh_data.isnull().sum()
```

Out[60]:
```
Ищет работу на должность:            0
Последнее/нынешнее место работы      1
Последняя/нынешняя должность         2
Обновление резюме                    0
Авто                                 0
Образование                          0
Пол                                  0
Возраст                              0
Опыт работы (месяц)                168
Город                                0
Готовность к переезду                0
Готовность к командировкам           0
полная занятость                     0
полный день                          0
частичная занятость                  0
сменный график                       0
проектная работа                     0
гибкий график                        0
волонтерство                         0
удаленная работа                     0
стажировка                           0
вахтовый метод                       0
ЗП (руб)                             0
dtype: int64
```

3. Есть пропуски в 3ех столбцах: **"Опыт работы (месяц)"**, **"Последнее/ нынешнее место работы"**, **"Последняя/нынешняя должность"**. Поступим следующим образом: удалиv строки, где есть пропуск в столбцах с местом работы и должностью. Пропуски в столбце с опытом работы заполним медианным значением.

In [61]:
```
hh_data = hh_data.dropna(subset=['Последнее/нынешнее место работы', 'Последняя/н
hh_data['Опыт работы (месяц)'] = hh_data['Опыт работы (месяц)'].fillna(hh_data['
```

Теперь удалим выбросы и аномалии

4. Удалим резюме, в которых указана заработная плата либо выше 1 млн. рублей, либо ниже 1 тыс. рублей.

In [62]:
```
delete_data = hh_data[(hh_data['ЗП (руб)'] > 1e6) | (hh_data['ЗП (руб)'] < 1e3)]
hh_data = hh_data.drop(delete_data.index)
print(f'Удалено {delete_data.shape[0]} записей')
```

```
Удалено 435 записей
```

5. Удалим резюме, в которых опыт работы в годах превышает возраст соискателя.

In [63]:
```
delete_data = hh_data[hh_data['Опыт работы (месяц)']/12 >= hh_data['Возраст']]
hh_data = hh_data.drop(delete_data.index)
print(f'Удалено {delete_data.shape[0]} записей')
```

```
Удалено 7 записей
```

6. В результате анализа мы обнаружили потенциальные выбросы в признаке **"Возраст"**. Это оказались резюме людей чересчур преклонного возраста для поиска работы. Попробуем построить распределение признака в **логарифмическом масштабе**. Добавим к графику линии, отображающие среднее и границы интервала метода трех сигм.

Найдём выбросы с помощью метода z-отклонения и удалим их из данных, используя логарифмический масштаб.

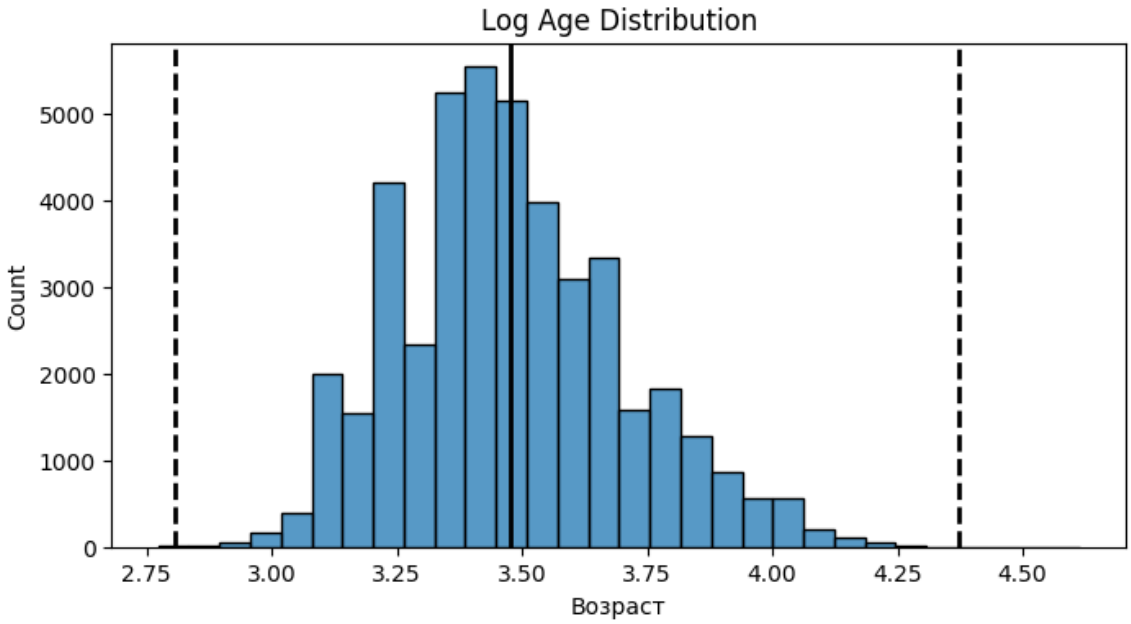
Выведим таблицу с полученными выбросами и оценим их.

In [64]:
```
fig, ax = plt.subplots(1, 1, figsize=(8, 4))
log_age = np.log(hh_data['Возраст'] + 1)
histplot = sns.histplot(log_age, bins=30, ax=ax)
histplot.axvline(log_age.mean(), color='k', lw=2)
histplot.axvline(log_age.mean()+ 4 *log_age.std(), color='k', ls='--', lw=2)
histplot.axvline(log_age.mean()- 3 *log_age.std(), color='k', ls='--', lw=2)
histplot.set_title('Log Age Distribution');

def outliers_z_score_mod(data, feature, left=3, right=4, log_scale=False):
    if log_scale:
        x = np.log(data[feature]+1)
    else:
        x = data[feature]
    mu = x.mean()
    sigma = x.std()
    lower_bound = mu - left * sigma
    upper_bound = mu + right * sigma
    delete_data = data[(x < lower_bound) | (x > upper_bound)]
    cleaned = data[(x >= lower_bound) & (x <= upper_bound)]
    return delete_data, cleaned
delete_data, hh_data = outliers_z_score_mod(hh_data, 'Возраст', left=3,  right=4
print(f'Удалено {delete_data.shape[0]} записей')
delete_data['Возраст']
```

```
Удалено 3 записей
```

Out[64]:
```
31137     15
32950     15
33654    100
Name: Возраст, dtype: int64
```



Вывод: График асимметричен в левую сторону, под категорию выбросов попадают люди с возрастом 15, 15 и 100 лет