Проект: Анализ вакансий на hh.ru

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objs as go
```

Исследование структуры данных

1). Прочитайте данные с помощью библиотеки Pandas. Совет: перед чтением обратите внимание на разделитель внутри файла.

In [ ]:
```
hh_data = pd.read_csv('dst-3.0_16_1_hh_database.csv', sep = ';')
display(hh_data.shape[0])
display(hh_data.shape[1])
```

```
44744
12
```

2). Выведите несколько первых (последних) строк таблицы, чтобы убедиться в том, что ваши данные не повреждены. Ознакомьтесь с признаками и их структурой.

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

	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). Выведите основную информацию о числе непустых значений в столбцах и их типах в таблице.

4). Обратите внимание на информацию о числе непустых значений.

In [ ]:
```
display(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
```

```
None
```

5). Выведите основную статистическую информацию о столбцах.

In [ ]:
```
display(hh_data.describe())
```

	Пол, возраст	ЗП	Ищет работу на должность:	Город, переезд, командировки	Занятость	График	Опыт работы	Посл
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...	пре,
freq	18	4064	3099	1261	30026	22727	3	

Преобразование данных

1). Начнем с простого - с признака **"Образование и ВУЗ"**. Его текущий формат это: **<Уровень образования год выпуска ВУЗ специальность...>**. Например:

- Высшее образование 2016 Московский авиационный институт (национальный исследовательский университет)...
- Неоконченное высшее образование 2000 Балтийская государственная академия рыбопромыслового флота… Нас будет интересовать только уровень образования.

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

Выполните преобразование, ответьте на контрольные вопросы и удалите признак "Образование и ВУЗ".

Совет: обратите внимание на структуру текста в столбце **"Образование и ВУЗ"**. Гарантируется, что текущий уровень образования соискателя всегда находится в первых 2ух слов и начинается с заглавной буквы. Воспользуйтесь этим.

Совет: проверяйте полученные категории, например, с помощью метода unique()

In [ ]:
```
def education_func(arg):
    """Определяет категорию "Образование"

    Args:
        arg (_type_): Запись об образовании

    Returns:
        _type_: образование
    """
    arg = ' '.join(arg.split(' ')[:3]) #первые два слова из строчки
    if 'Высшее' in arg:
        return 'высшее'
    elif 'Неоконченное высшее' in arg:
        return 'неоконченное высшее'
    elif 'Среднее специальное' in arg:
        return 'среднее специальное'
    elif 'Среднее образование' in arg:
        return 'среднее'
    else: return 'другое'

hh_data['Образование'] = hh_data['Образование и ВУЗ'].apply(education_fun
hh_data['Образование'] = hh_data['Образование'].astype('category') #преоб

hh_data = hh_data.drop('Образование и ВУЗ', axis = 1) #удаление столбца

display(hh_data['Образование'].value_counts()) #количество повторений по
#display(hh_data['Образование'].unique()) #уникальные значения
```

```
высшее                  33863
среднее специальное      5765
неоконченное высшее      4557
среднее                   559
Name: Образование, dtype: int64
```

2). Теперь нас интересует столбец **"Пол, возраст"**. Сейчас он представлен в формате **<Пол , возраст , дата рождения >**. Например:

- Мужчина , 39 лет , родился 27 ноября 1979
- Женщина , 21 год , родилась 13 января 2000 Как вы понимаете, нам необходимо выделить каждый параметр в отдельный столбец.

Создайте два новых признака **"Пол"** и **"Возраст"**. При этом важно учесть:

- Признак пола должен иметь 2 уникальных строковых значения: 'М' - мужчина, 'Ж' - женщина.
- Признак возраста должен быть представлен целыми числами.

Выполните преобразование, ответьте на контрольные вопросы и удалите признак **"Пол, возраст"** из таблицы.

Совет: обратите внимание на структуру текста в столбце, в части на то, как разделены параметры пола, возраста и даты рождения между собой - символом ' , '. Гарантируется, что структура одинакова для всех строк в таблице. Вы можете воспользоваться этим.

In [ ]:
```
def sex_func(arg):
    """Определяет пол

    Args:
        arg (_type_): Запись о гендерной принадлежности

    Returns:
        _type_: пол
    """
    arg = arg.split(',')[0] #первое слово из строки
    if 'Мужчина' in arg:
        return 'М'
    elif 'Женщина' in arg:
        return 'Ж'
    else: return 'другое'

#разбиение строки по элементам, выбор элемента, указывающего возраст, пре
hh_data['Возраст'] = hh_data['Пол, возраст'].apply(lambda arg: arg.split(

hh_data['Пол'] = hh_data['Пол, возраст'].apply(sex_func) #добавление стол
hh_data['Пол'] = hh_data['Пол'].astype('category') #преобразование типа д

hh_data = hh_data.drop('Пол, возраст', axis = 1) #удаление столбца

display(round(hh_data['Пол'].value_counts(normalize=True)*100, 2)) #соотн
display('Средний возраст с округлением до сотых: ', round(hh_data['Возрас
```

```
М    80.93
Ж    19.07
Name: Пол, dtype: float64
'Средний возраст с округлением до сотых: '
32.2
```

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

Из столбца нам необходимо выделить общий опыт работы соискателя в месяцах, новый признак назовем "Опыт работы (месяц)"

Для начала обсудим условия решения задачи:

- Во-первых, в данном признаке есть пропуски. Условимся, что если мы встречаем пропуск, оставляем его как есть (функция-преобразование возвращает NaN)
- Во-вторых, в данном признаке есть скрытые пропуски. Для некоторых соискателей в столбце стоит значения "Не указано". Их тоже обозначим как NaN (функция-преобразование возвращает NaN)
- В-третьих, нас не интересует информация, которая описывается после указания опыта работы (периоды работы в различных компаниях)
- В-четвертых, у нас есть проблема: опыт работы может быть представлен только в годах или только месяцах. Например, можно встретить следующие варианты:
  - Опыт работы 3 года 2 месяца…
  - Опыт работы 4 года…
  - Опыт работы 11 месяцев…
  - Учитывайте эту особенность в вашем коде

Учитывайте эту особенность в вашем коде

В результате преобразования у вас должен получиться столбец, содержащий информацию о том, сколько месяцев проработал соискатель. Выполните преобразование, ответьте на контрольные вопросы и удалите столбец **"Опыт работы"** из таблицы.

In [ ]:
```
def experience_func(arg):
    """Определяет опыт работы в месяцах

    Args:
        arg (_type_): запись об опыте работы

    Returns:
        _type_: количество месяцев
    """
    years =['год', 'года', 'лет'] #список слов для определения количества
    months=['месяц', 'месяца', 'месяцев'] #список слов для определения ко
    y=0 #счетчик лет
    m=0 #счетчик месяцев

    if arg is np.nan:
        return None #если нет данных, вернется пустое значение

    experience=arg.split(' ')[:7] #разделение по пробелу первых слов из з

    for i, j in enumerate (experience): #перебор слов по порядку, значени

        if j in years:
            y = int(experience[i-1]) #если значение в списке слов, обозна
        if j in months:
            m = int(experience[i-1]) #если значение в списке слов, обозна
    exp_period = int(y*12 + m) #подсчет общего количества месяцев
    return exp_period

hh_data['Опыт работы (месяц)'] = hh_data['Опыт работы'].apply(experience_
hh_data = hh_data.drop('Опыт работы', axis=1) #удаление столбца
display('Медиана для признака Опыт работы: ', round(hh_data['Опыт работы
```

```
'Медиана для признака Опыт работы: '
100
```

4). Хорошо идем! Следующий на очереди признак "Город, переезд, командировки". Информация в нем представлена в следующем виде: **<Город , (метро) , готовность к переезду (города для переезда) , готовность к командировкам>**. В скобках указаны необязательные параметры строки. Например, можно встретить следующие варианты:

- Москва , не готов к переезду , готов к командировкам
- Москва , м. Беломорская , не готов к переезду, не готов к командировкам
- Воронеж , готов к переезду (Сочи, Москва, Санкт-Петербург) , готов к командировкам

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" (их список ниже), остальные обозначьте как "другие".

  Список городов-миллионников:

  ```
  million_cities = ['Новосибирск', 'Екатеринбург','Нижний Новгород','Казань', 'Челябинск','Омск', 'Самара', 'Ростов-на-Дону', 'Уфа', 'Красноярск', 'Пермь', 'Воронеж','Волгоград']
  ```

  Инфорация о метро, рядом с которым проживает соискатель нас не интересует.

- Признак **"Готовность к переезду"** должен иметь два возможных варианта: True или False. Обратите внимание, что возможны несколько вариантов описания готовности к переезду в признаке "Город, переезд, командировки". Например:

  - … , готов к переезду , …
  - … , не готова к переезду , …
  - … , готова к переезду (Москва, Санкт-Петербург, Ростов-на-Дону)
  - … , хочу переехать (США) , …

  Нас интересует только сам факт возможности или желания переезда.

- Признак **"Готовность к командировкам"** должен иметь два возможных варианта: True или False. Обратите внимание, что возможны несколько вариантов описания готовности к командировкам в признаке "Город, переезд, командировки". Например:

  - … , готов к командировкам , …
  - … , готова к редким командировкам , …
  - … , не готов к командировкам , …

  Нас интересует только сам факт готовности к командировке.

  Еще один важный факт при выгрузки данных у некоторых соискателей

Еще один важный факт: при выгрузки данных у некоторых соискателей "потерялась" информация о готовности к командировкам. Давайте по умолчанию будем считать, что такие соискатели не готовы к командировкам.

Выполните преобразования и удалите столбец **"Город, переезд, командировки"** из таблицы.

Совет: обратите внимание на то, что структура текста может меняться в зависимости от указания ближайшего метро. Учите это, если будете использовать порядок слов в своей программе.

In [ ]:
```
def city_func(arg):
    """Определяет город

    Args:
        arg (_type_): Запись о городе проживания, готовности к переездам

    Returns:
        _type_: город
    """
    ##список городов-миллионников
    million_cities = ['Новосибирск', 'Екатеринбург','Нижний Новгород','Ка
    #if arg is np.nan:
    #    return None #возвращает пустое значение, если нет данных
    city = arg.split(' , ')[0] #первый элемент из строки информации (горо
    if (city == 'Москва') or (city == 'Санкт-Петербург'):
        return city
    elif city in million_cities:
        return 'город миллионник' #если название города встречается в спи
    else: return 'другие'


def moving_func(arg):
    """Определяет готовность к переезду в формате Да/Нет

    Args:
        arg (_type_): Запись о городе проживания, готовности к переездам

    Returns:
        _type_: True/False
    """
    #если в строке информации встречается одно из указанных значений, то
    if ('не готов к переезду' in arg) or ('не готова к переезду' in arg):
        return False
    elif 'хочу' in arg:
        return True
    else: return True

def trip_func(arg):
    """Определяет готовность к командировкам в формате Да/Нет

    Args:
        arg (_type_): Запись о городе проживания, готовности к переездам

    Returns:
        _type_: True/False
    """
    if ('командировка' in arg):
        #если в строке информации встречается одно из указанных значений,
```

```
        if ('не готов к командировкам' in arg) or ('не готова к командиро
            return False
        else: return True
    else: return False

##создание столбцов с новыми признакаи с применением функций
hh_data['Город'] = hh_data['Город, переезд, командировки'].apply(city_fun
hh_data['Готовность к переезду'] = hh_data['Город, переезд, командировки'
hh_data['Готовность к командировкам'] = hh_data['Город, переезд, командир
hh_data = hh_data.drop('Город, переезд, командировки', axis=1) #удаление

display('Процент соискателей, проживающих в Санкт-Петербурге: ', round(hh
display('Процент соискателей, готовых к командировкам и переездам: ', rou
```

- полная занятость, частичная занятость
- частичная занятость, проектная работа, волонтерство
- полный день, удаленная работа
- вахтовый метод, гибкий график, удаленная работа, полная занятость

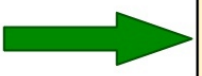
Такой вариант признаков имеет множество различных комбинаций, а значит множество уникальных значений, что мешает анализу. Нужно это исправить!

Давайте создадим признаки-мигалки для каждой категории: если категория присутствует в списке желаемых соискателем, то в столбце на месте строки рассматриваемого соискателя ставится True, иначе - False.

Такой метод преобразования категориальных признаков называется One Hot Encoding и его схема представлена на рисунке ниже:

Схема One Hot Encoding преобразования

Занятость
полная занятость, частичная занятость, стажировка
полная занятость, проектная работа
стажировка, проектная работа, волонтерство



полная занятость	частичная занятость	проектная работа	стажировка	волонтерство
True	True	False	True	False
True	False	True	False	False
False	False	True	True	True

Выполните данное преобразование для признаков "Занятость" и "График", ответьте на контрольные вопросы, после чего удалите их из таблицы

In [ ]:
```
employments = ['полная занятость', 'частичная занятость', 'проектная рабо
schedules = ['полный день', 'сменный график', 'гибкий график', 'удаленная

for employment, schedule in zip(employments, schedules): #идем по каждому
    #добавляем столбец по каждому названию по обоим спискам
    #для элементов столбцов "Занятость" и "График" применяем функцию
    #если в строке проверяемого столбца есть название элемента итерируемо
    hh_data[employment] = hh_data['Занятость'].apply(lambda x: employment
    hh_data[schedule] = hh_data['График'].apply(lambda x: schedule in x)
#удаляем столбцы
hh_data = hh_data.drop('Занятость', axis=1)
hh_data = hh_data.drop('График', axis=1)

#количество людей, ищущих проектную раоту и волонтерство одновременно
display('Количество людей, ищущих проектную раоту и волонтерство одноврем

#количество людей, ищущих раоту с гибким графиком и вахтовым методом одно
display('Количество людей, ищущих раоту с гибким графиком и вахтовым мето
```

```
'Количество людей, ищущих проектную раоту и волонтерство одновременно: '
436
'Количество людей, ищущих раоту с гибким графиком и вахтовым методом одно
временно: '
2311
```

6). (2 балла) Наконец, мы добрались до самого главного и самого важного - признака заработной платы **"ЗП"**. В чем наша беда? В том, что помимо желаемой заработной платы соискатель указывает валюту, в которой он бы хотел ее получать, например:

- 30000 руб.
- 50000 грн.
- 550 USD

Нам бы хотелось видеть заработную плату в единой валюте, например, в рублях. Возникает вопрос, а где взять курс валют по отношению к рублю?

На самом деле язык Python имеет в арсенале огромное количество возможностей получения данной информации, от обращения к API Центробанка, до использования специальных библиотек, например pycbrf. Однако, это не тема нашего проекта.

Поэтому мы пойдем в лоб: обратимся к специальным интернет-ресурсам для получения данных о курсе в виде текстовых файлов. Например, MDF.RU, данный ресурс позволяет удобно экспортировать данные о курсах различных валют и акций за указанные периоды в виде csv файлов. Мы уже сделали выгрузку курсов валют, которые встречаются в наших данных за период с 29.12.2017 по 05.12.2019. Скачать ее вы можете **на платформе**

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

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

Перед вами таблица соответствия наименований иностранных валют в наших данных и их общепринятых сокращений, которые представлены в нашем файле с курсами валют. Пропорция - это число, за сколько единиц валюты указан курс в таблице с курсами. Например, для казахстанского тенге курс на 20.08.2019 составляет 17.197 руб. за 100 тенге, тогда итоговый курс равен - 17.197 / 100 = 0.17197 руб за 1 тенге. Воспользуйтесь этой информацией в ваших преобразованиях.

Наименование валюты в данных	*Наименование валюты в ISO кодировке*	*Пропорция*	*Расшифровка*
грн	UAH	10	гривна
USD	USD	1	доллар
EUR	EUR	1	евро

белруб	BYN	1	белорусский рубль
KGS	KGS	10	киргизский сом
сум	UZS	10000	узбекский сум
AZN	AZN	1	азербайджанский манат
KZT	KZT	100	казахстанский тенге

Осталось только понять, откуда брать дату, по которой определяется курс? А вот же она - в признаке **"Обновление резюме"**, в нем содержится дата и время, когда соискатель выложил текущий вариант своего резюме. Нас интересует только дата, по ней бы и будем сопоставлять курсы валют.

Теперь у нас есть вся необходимая информация для того, чтобы создать признак "ЗП (руб)" - заработная плата в рублях.

После ответа на контрольные вопросы удалите исходный столбец заработной платы "ЗП" и все промежуточные столбцы, если вы их создавали.

Итак, давайте обсудим возможный алгоритм преобразования:

1. Перевести признак "Обновление резюме" из таблицы с резюме в формат datetime и достать из него дату. В тот же формат привести признак "date" из таблицы с валютами.
2. Выделить из столбца "ЗП" сумму желаемой заработной платы и наименование валюты, в которой она исчисляется. Наименование валюты перевести в стандарт ISO согласно с таблицей выше.
3. Присоединить к таблице с резюме таблицу с курсами по столбцам с датой и названием валюты (подумайте, какой тип объединения надо выбрать, чтобы в таблице с резюме сохранились данные о заработной плате, изначально представленной в рублях). Значение close для рубля заполнить единицей 1 (курс рубля самого к себе)
4. Умножить сумму желаемой заработной платы на присоединенный курс валюты (close) и разделить на пропорцию (обратите внимание на пропуски после объединения в этих столбцах), результат занести в новый столбец "ЗП (руб)".

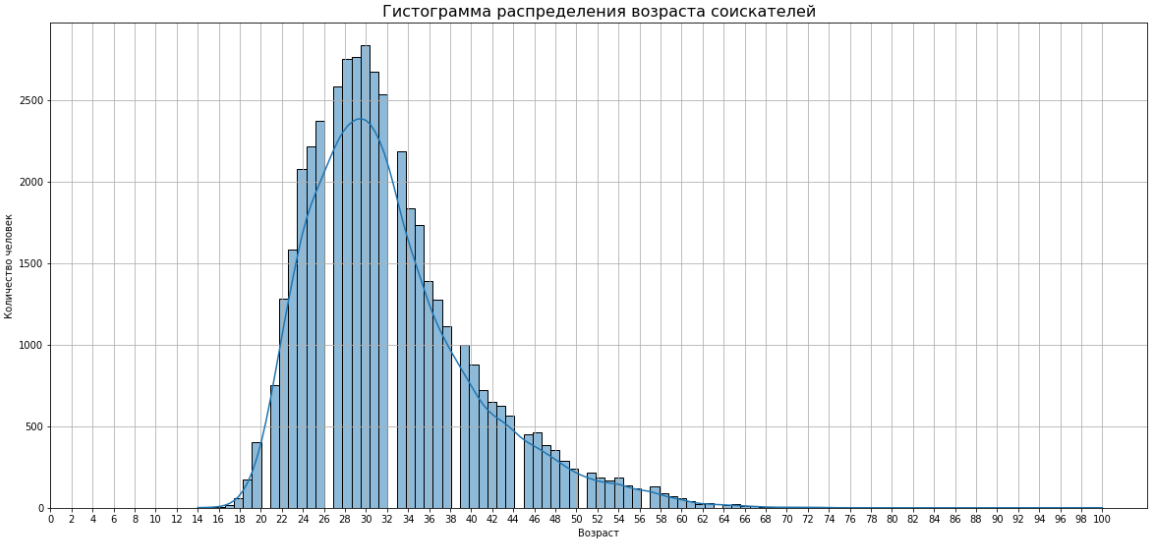
In [ ]:
```
exrate_data = pd.read_csv('ExchangeRates.csv', sep = ',') #чтение файла с
#print(exrate_data.head())

#функция для выделения первого аргумента из строки
def salary_func(arg):
    """Определяет размер заработной платы

    Args:
        arg (_type_): Запись о заработной плате

    Returns:
        _type_: сумма
    """
    salary = float(arg.split(' ')[0])
    return salary

#hh_data['cur_del'] = hh_data['ЗП'].apply(lambda x: x.split(' ')[1].repla
#display(hh_data['cur_del'].unique())
def currency_func(arg):
    """Определяе валюту для указанной заработной платы

    Args:
        arg (_type_): Запись о заработной плате

    Returns:
        _type_: условное обозначение валюты
    """
    #словарь возможных значений валют
    d_currency = {'руб': 'RUB', 'белруб': 'BYN', 'KZT': 'KZT', 'EUR': 'EU
    currency = arg.split(' ')[1].replace('.', '') #выделение первого аргу
    return d_currency[currency] #возвращает значение словаря по ключу(арг

hh_data['salary_del'] = hh_data['ЗП'].apply(salary_func) #создание столбц
hh_data['currency_del'] = hh_data['ЗП'].apply(currency_func) #создание ст
#display(hh_data.head())

exrate_data['date'] = pd.to_datetime(exrate_data['date'], dayfirst=True).
hh_data['Обновление резюме'] = pd.to_datetime(hh_data['Обновление резюме'

#объединение двух баз данных (подстановка значений из таблицы с курсами в
rate_date_cur = hh_data.merge(exrate_data,
                        left_on=['currency_del', 'Обновление резюме'], #пр
                        right_on=['currency', 'date'], #признаки, по котор
                        how='left') #все значения из основной таблицы

#заполнение пустых значений "1" (для рублей)
rate_date_cur['close'] = rate_date_cur['close'].fillna(1)
rate_date_cur['proportion'] = rate_date_cur['proportion'].fillna(1)
#display(rate_date_cur.head())

#добавлени столбца с желаемой ЗП на основании расчетов по данным из вспом
hh_data['ЗП (руб)'] = rate_date_cur['close'] * rate_date_cur['salary_del'

#удаление столбцов
hh_data = hh_data.drop(['ЗП', 'salary_del', 'currency_del'], axis=1)

display('Медианное значение желаемой ЗП в тыс. рублей: ', round(hh_data['
```

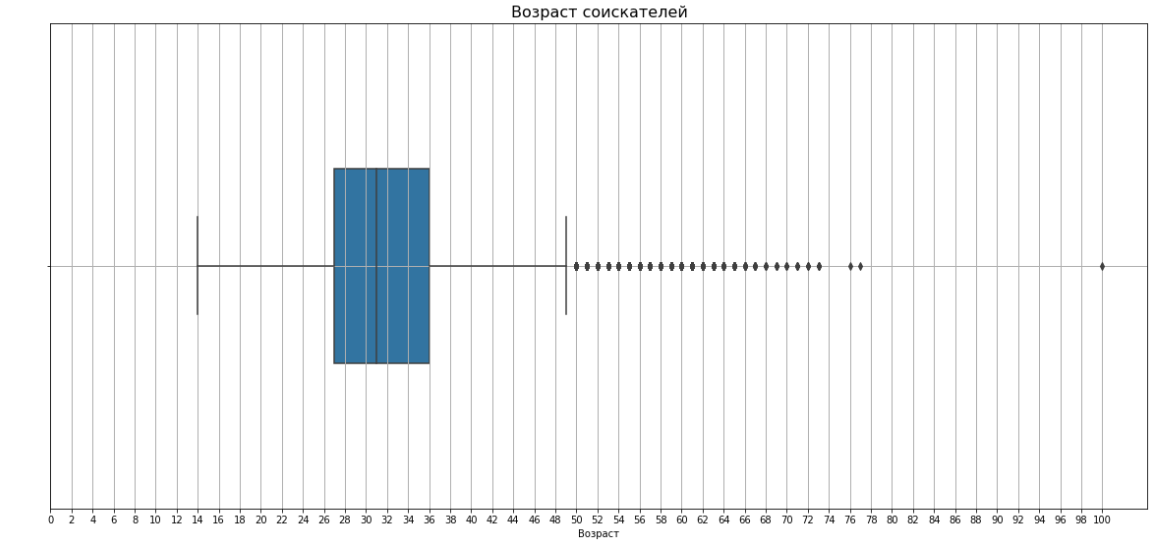
```
'Медианное значение желаемой ЗП в тыс. рублей: '
59
```

Исследование зависимостей в данных

1). Постройте распределение признака **"Возраст"**. Опишите распределение, отвечая на следующие вопросы: чему равна мода распределения, каковы предельные значения признака, в каком примерном интервале находится возраст большинства соискателей? Есть ли аномалии для признака возраста, какие значения вы бы причислили к их числу? *Совет: постройте гистограмму и коробчатую диаграмму рядом.*

In [ ]:
```
fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(20, 20))

sns.histplot(
    data=hh_data,
    x='Возраст',
    bins=100,
    kde=True,
    ax=axes[0]
);
axes[0].set_ylabel('Количество человек')
axes[0].set_title('Гистограмма распределения возраста соискателей', fonts
step_ticks = list(range(0, 101, 2))
axes[0].set_xticks(step_ticks)
axes[0].grid()

sns.boxplot(
    data=hh_data,
    x='Возраст',
    orient='h',

    width=0.4
);
axes[1].set_title('Возраст соискателей', fontsize=16)
axes[1].set_xticks(step_ticks)
axes[1].grid(linewidth=1)
```

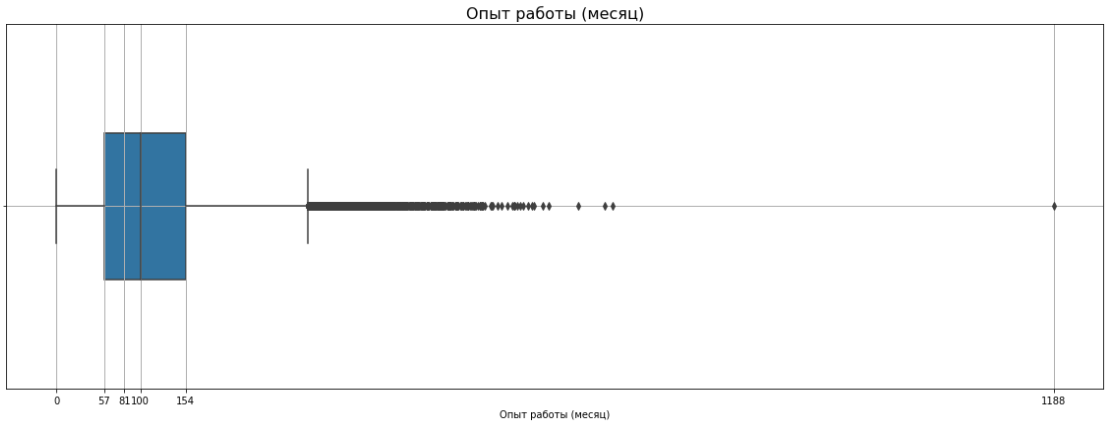




Модальное значение возраста соискателей равно 30 лет. В основном возраст соискателей лежит в диапозоне от 14 до 49 лет, большинство из них возраста от 27 до 36 лет. Выбросам считаются значения от 50 лет, но аномальным можно считать возраст выше пенсионного, например 100 лет.

2). Постройте распределение признака **"Опыт работы (месяц)"**. Опишите данное распределение, отвечая на следующие вопросы: чему равна мода распределения, каковы предельные значения признака, в каком примерном интервале находится опыт работы большинства соискателей? Есть ли аномалии для признака опыта работы, какие значения вы бы причислили к их числу? *Совет: постройте гистограмму и коробчатую диаграмму рядом.*

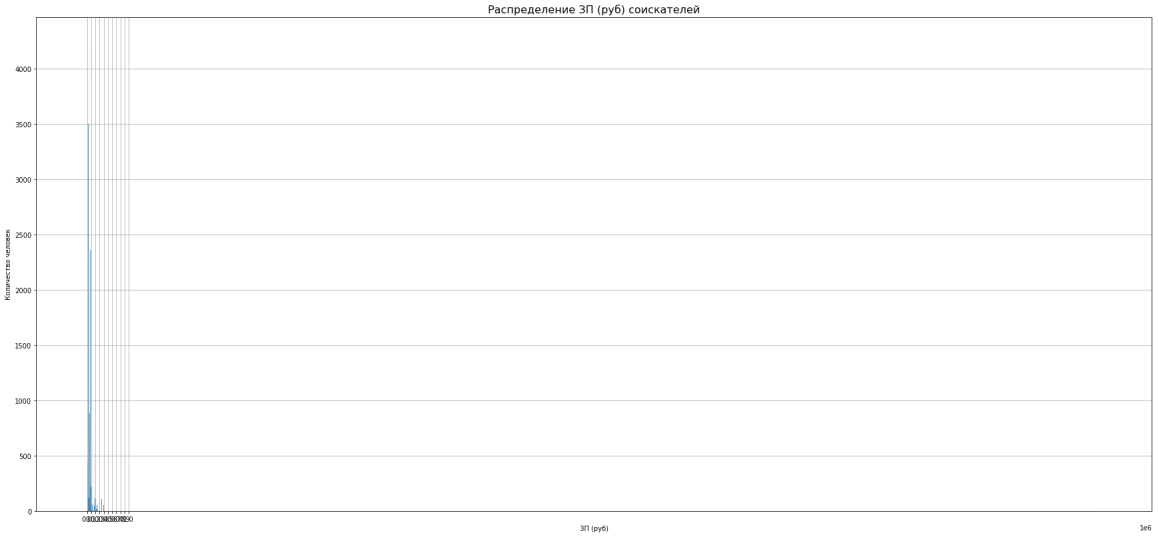
In [ ]:
```
fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(20, 15))

sns.histplot(
    data=hh_data,
    x='Опыт работы (месяц)',
    ax=axes[0]
);

axes[0].set_ylabel('Количество человек')
axes[0].set_title('Гистограмма распределения опыта работы соискателей', f
step_ticks = list(range(0, 1201, 36))
axes[0].set_xticks(step_ticks)
axes[0].grid()


sns.boxplot(
    data=hh_data,
    x='Опыт работы (месяц)',
    orient='h',
    width=0.4
);
axes[1].set_title('Опыт работы (месяц)', fontsize=16)

ticks_value = [int(hh_data['Опыт работы (месяц)'].max()), int(hh_data['Оп
               int(hh_data['Опыт работы (месяц)'].mode()), int(hh_data['О
               int(hh_data['Опыт работы (месяц)'].quantile(0.25)), int(hh
               ]
axes[1].set_xticks(sorted(ticks_value))
axes[1].grid(linewidth=1)
```

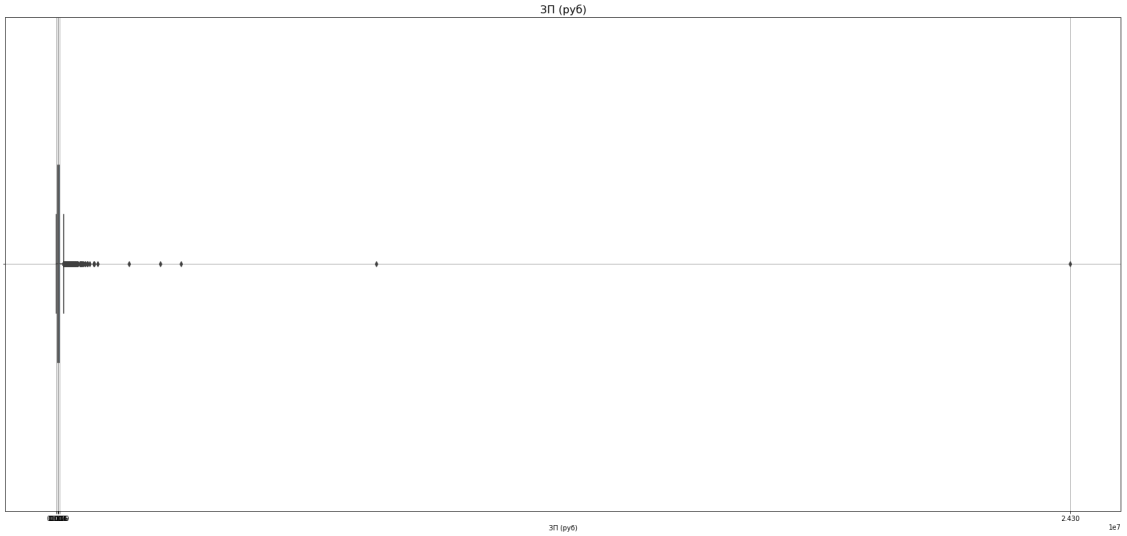


Модальное значение распределения равно 81 месяцу. Опыт соискателей по данным указан от 0 до 1188 месяцев. Опыт работы большинства соискателей примерно в интервале от 57 до 154 месяцев. Аномальным для признака опыта работы значением является 1188 месяцев (99 лет), так как это противоречит здравому смыслу, особенно если учесть, что максимальный возраст был указан 100 лет.

3). Постройте распределение признака **"ЗП (руб)"**. Опишите данное распределение, отвечая на следующие вопросы: каковы предельные значения признака, в каком примерном интервале находится заработная плата большинства соискателей? Есть ли аномалии для признака ЗП? Обратите внимание на гигантские размеры желаемой заработной платы. *Совет: постройте гистограмму и коробчатую диаграмму рядом.*

In [ ]:
```
fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(30, 30))

sns.histplot(
    data=hh_data,
    x='ЗП (руб)',
    #bins=100,
    ax=axes[0]
);

axes[0].set_ylabel('Количество человек')
axes[0].set_title('Распределение ЗП (руб) соискателей', fontsize=16)
step_ticks = list(range(0, 1000001, 100000))
axes[0].set_xticks(step_ticks)
axes[0].grid()

sns.boxplot(
    data=hh_data,
    x='ЗП (руб)',
    orient='h',
    width=0.4
);
axes[1].set_title('ЗП (руб)', fontsize=16)
ticks_value = [int(hh_data['ЗП (руб)'].max()), int(hh_data['ЗП (руб)'].mi
               int(hh_data['ЗП (руб)'].mode()), int(hh_data['ЗП (руб)'].m
               int(hh_data['ЗП (руб)'].quantile(0.25)), int(hh_data['ЗП (
               ]
axes[1].set_xticks(sorted(ticks_value))
axes[1].grid(linewidth=1)

#display(f'Предельные значения признака лежат в диапазоне от {ticks_value
```



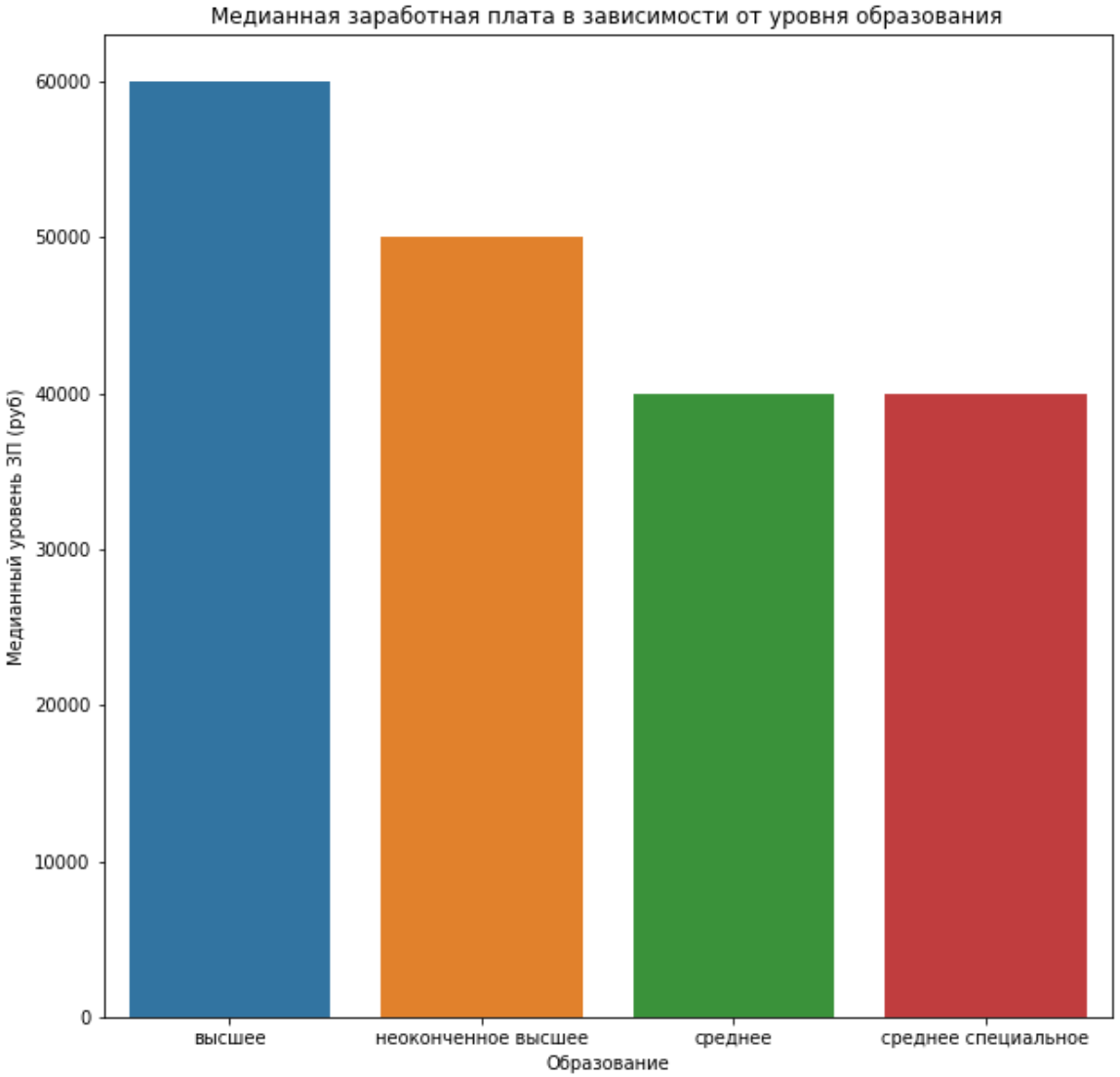


Предельные значения желаемой заработной платы лежат в диапазоне от 24 304 876 до 1 руб. Заработная плата большинства соискателей примерно находится в интервале от 37 082 до 95 000 руб. Аномальными значениями можно считать минимальные (1 руб) и свыше 1 млн руб.

4). Постройте диаграмму, которая показывает зависимость **медианной** желаемой заработной платы (**"ЗП (руб)"**) от уровня образования (**"Образование"**). Используйте для диаграммы данные о резюме, где желаемая заработная плата меньше 1 млн рублей. *Сделайте выводы по представленной диаграмме: для каких уровней образования наблюдаются наибольшие и наименьшие уровни желаемой заработной платы? Как вы считаете, важен ли признак уровня образования при прогнозировании заработной платы?*

In [ ]:
```
education_salary = hh_data[hh_data['ЗП (руб)'] < 1000000]
education_salary = education_salary.groupby(education_salary['Образование
#display(education_salary)

fig = plt.figure(figsize=(10, 10))
barplot = sns.barplot(
    data=education_salary,
    x='Образование',
    y='ЗП (руб)'
)
barplot.set_title('Медианная заработная плата в зависимости от уровня обр
barplot.set_xlabel('Образование');
barplot.set_ylabel('Медианный уровень ЗП (руб)')
```

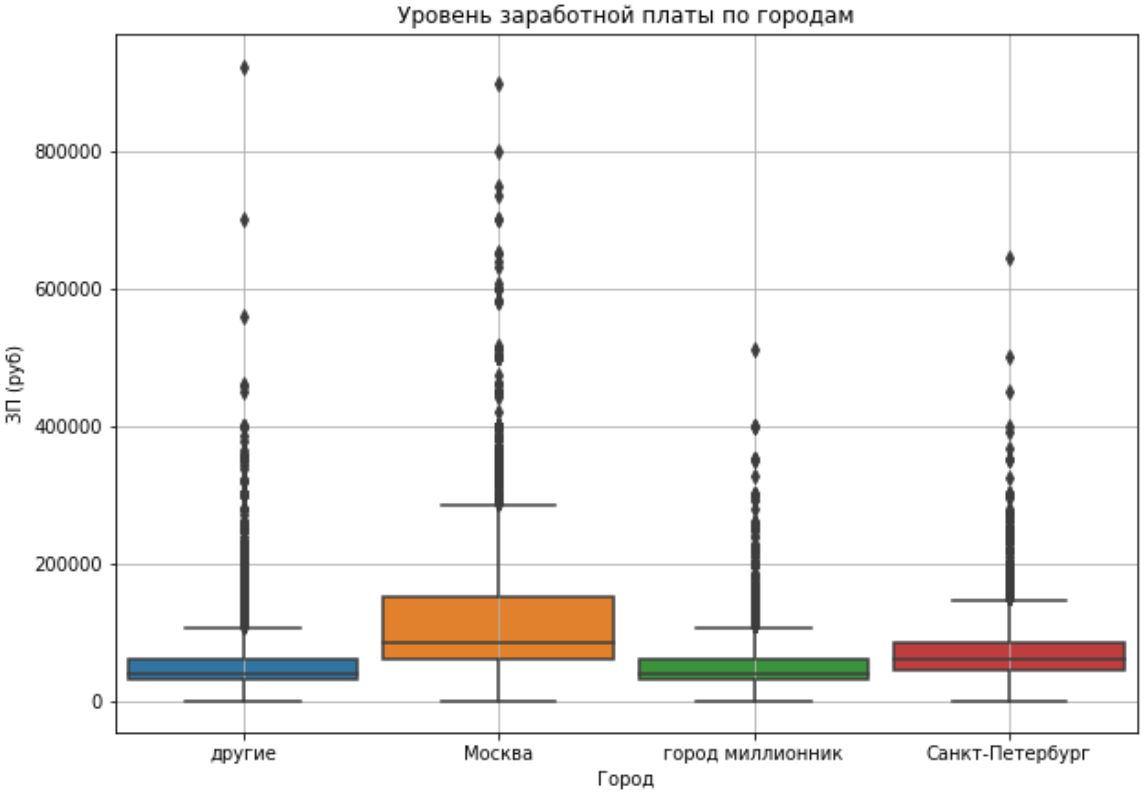
Out[ ]:
```
Text(0, 0.5, 'Медианный уровень ЗП (руб)')
```



Наибольший уровень желаемой заработной платы наблюдается для категории "высшее образование", для категорий "среднее" и "среднее специальное" - нименьший. На данном графике наблюдается прямая зависимость заработной платы от уровня образования. Соответственно, при прогнозировании заработной платы признак уровня образования следует учитывать.

5). Постройте диаграмму, которая показывает распределение желаемой заработной платы (**"ЗП (руб)"**) в зависимости от города (**"Город"**). Используйте для диаграммы данные о резюме, где желая заработная плата меньше 1 млн рублей. *Сделайте выводы по полученной диаграмме: как соотносятся медианные уровни желаемой заработной платы и их размах в городах? Как вы считаете, важен ли признак города при прогнозировании заработной платы?*

In [ ]:
```
city_salary = hh_data[hh_data['ЗП (руб)'] < 1000000]

fig = plt.figure(figsize=(10, 7))
boxplot = sns.boxplot(
    data=city_salary,
    y='ЗП (руб)',
    x='Город',
    orient='v',
    width=0.9
)
boxplot.set_title('Уровень заработной платы по городам');
boxplot.set_xlabel('Город');
boxplot.set_ylabel('ЗП (руб)');
boxplot.grid()
```



Самый высокий медианный уровень желаемой заработной платы и её размах в Москве, чуть ниже в Санкт-Петербурге, в остальных категориях городов медианный уровень примерно одинаковый. На основании этого можно сделать вывод, что признак города влияет на уровень заработной платы (является ли город столицей).

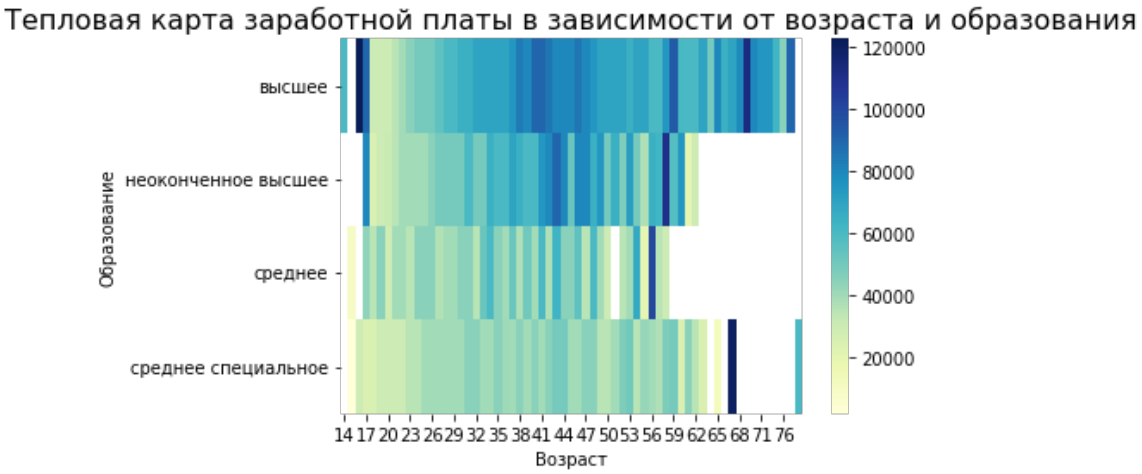
6). Постройте **многоуровневую столбчатую диаграмму**, которая показывает зависимость медианной заработной платы (**"ЗП (руб)"**) от признаков **"Готовность к переезду"** и **"Готовность к командировкам"**. Проанализируйте график, сравнив уровень заработной платы в категориях.

In [ ]:
```
#salary_moving_trip = hh_data.copy()
#salary_moving_trip['ЗП (руб)'] = round(salary_moving_trip['ЗП (руб)'].as

salary_moving_trip = hh_data.groupby(by=['Готовность к переезду', 'Готовн
#display(salary_moving_trip)

#display(hh_data['Готовность к переезду'].value_counts())
#display(hh_data['Готовность к командировкам'].value_counts())

fig = px.bar(
    data_frame = salary_moving_trip,
    x = 'ЗП (руб)',
    y = 'Готовность к командировкам',
    barmode = 'group',
    color = 'Готовность к переезду',
    orientation='h',
    title='Медианная ЗП (руб) по готовности к командировкам и к переездам

)
fig.show()
```

Самый высокий уровень заработной платы наблюдается среди соискателей, готовых к командировкам, в особенности у тех, кто еще готов и к переезду. Вероятнее всего, это связано со спецификой должностей, на которые соискатели претендуют.

7). Постройте сводную таблицу, иллюстрирующую зависимость **медианной** желаемой заработной платы от возраста (**"Возраст"**) и образования (**"Образование"**). На полученной сводной таблице постройте **тепловую карту**. Проанализируйте тепловую карту, сравнив показатели внутри групп.

In [ ]:
```
salary_age_education = hh_data.groupby(by=['Возраст', 'Образование'], as_
#print(salary_age_education)
hm_table = salary_age_education.pivot_table(
    values='ЗП (руб)',
    columns='Возраст',
    index='Образование',
)

#display(hm_table)

heatmap = sns.heatmap(data=hm_table, cmap='YlGnBu')
heatmap.set_title('Тепловая карта заработной платы в зависимости от возра
```



Соискатели с высшим образованием получают более высокую зарплату, чем соискатели такого же возраста с более низкой степенью образования. Для соискателей с высшим образованием характерно, что самую высокую ЗП получают люди с 35 до 50 лет, далее размер ЗП идет на уменьшение. Для остальных категорий признака "Образование" после 25 лет уровень ЗП остается примерно одинаковым.

8). Постройте **диаграмму рассеяния**, показывающую зависимость опыта работы (**"Опыт работы (месяц)"**) от возраста (**"Возраст"**). Опыт работы переведите из месяцев в года, чтобы признаки были в едином масштабе. Постройте на графике дополнительно прямую, проходящую через точки (0, 0) и (100, 100). Данная прямая соответствует значениям, когда опыт работы равен возрасту человека. Точки, лежащие на этой прямой и выше нее - аномалии в наших данных (опыт работы больше либо равен возрасту соискателя)

In [ ]:
```
age_experience = hh_data.copy()
age_experience['Опыт работы'] = age_experience['Опыт работы (месяц)']/12
line_step = list(range(0, 101, 10))


fig = go.Figure()

fig.add_trace(go.Scatter(x = age_experience['Возраст'], y = age_experienc
fig.add_trace(go.Scatter(x = [0, 100], y = [0, 100]))
fig.update_layout(title="Зависимость опыта работы от возраста",
                  xaxis_title="Возраст",
                  yaxis_title="Опыт работы")
fig.show()
```

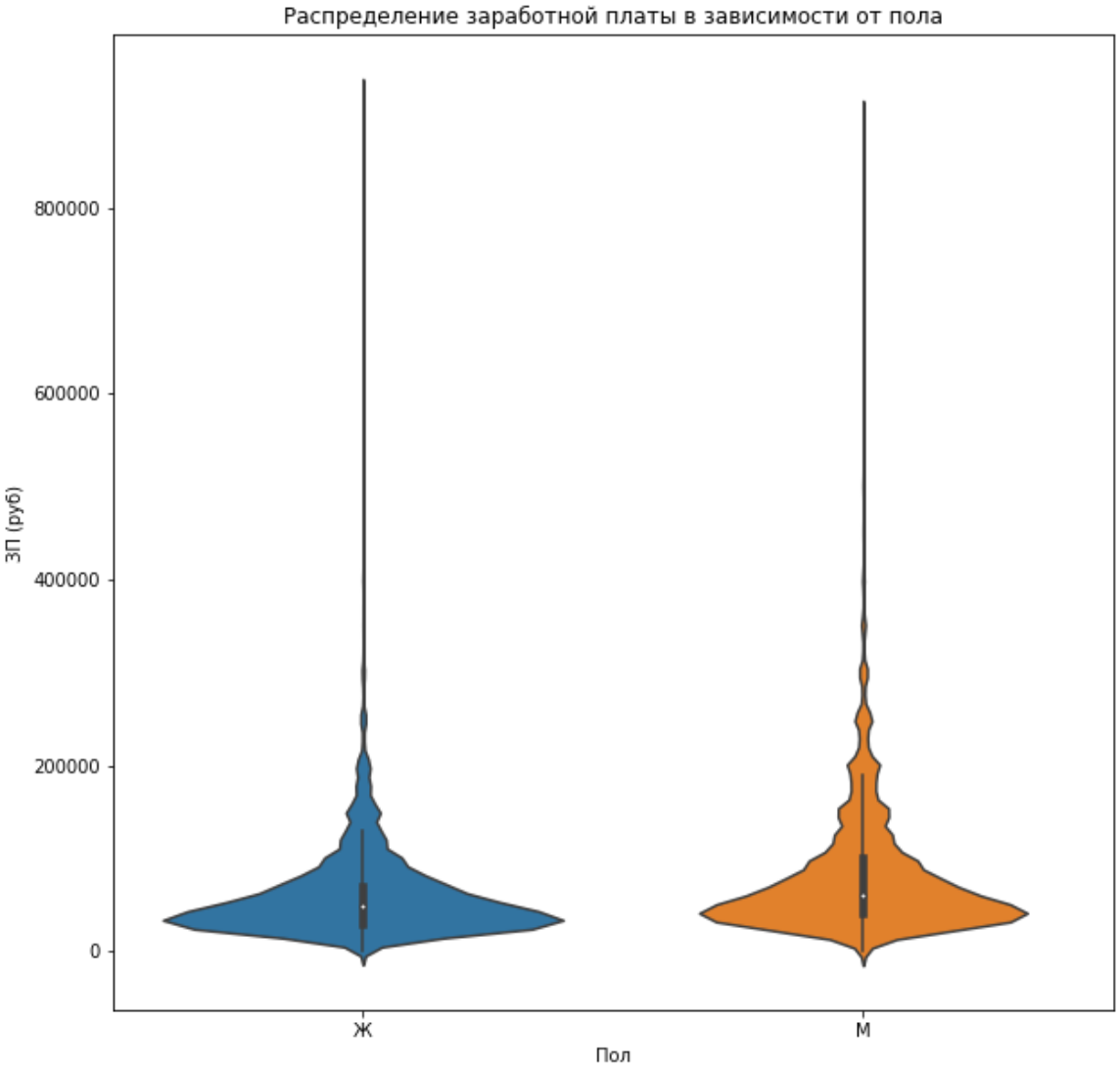
Данный график хорошо иллюстрирует выбросы. Выбросами можно считать точки, приближенные к прямой или лежащие выше неё, так как опыт работы может быть как минимум на 14 лет меньше возраста. Объясняется это тем, что в соответствии с российским законодательством трудоустройство возможно с 14 лет.

Дополнительные баллы

Для получения 2 дополнительных баллов по разведывательному анализу постройте еще два любых содержательных графика или диаграммы, которые помогут проиллюстрировать влияние признаков/взаимосвязь между признаками/распределения признаков. Приведите выводы по ним. Желательно, чтобы в анализе участвовали признаки, которые мы создавали ранее в разделе "Преобразование данных".

In [ ]:
```
hh_df_salary = hh_data[hh_data['ЗП (руб)'] < 1000000]

fig_1 = plt.figure(figsize=(10, 10))
violinplot = sns.violinplot(data = hh_df_salary,
                            x = "Пол",
                            y = "ЗП (руб)"
                            )
violinplot.set_title('Распределение заработной платы в зависимости от пол
fig_1.show()

fig_2 = px.scatter_3d(
    data_frame=hh_df_salary, #DataFrame
    x = 'Образование', #ось абсцисс
    y = 'ЗП (руб)', #ось ординат
    z = 'Опыт работы (месяц)', #ось аппликат
    color='Пол', #расцветка в зависимости от страны
    width=1000,
    height=1000,
    title='ЗП (руб) мужчин и женщин в зависимости от опыта работы и образ
)
fig_2.show()
```

```
/home/zlata/anaconda3/envs/sf/lib/python3.7/site-packages/ipykernel_launc
her.py:9: UserWarning:

Matplotlib is currently using module://matplotlib_inline.backend_inline,
which is a non-GUI backend, so cannot show the figure.
```



Если проанализировать уровень ЗП для мужчин и для женщин, то можно увидеть, что в целом отличий особых нет, за исключением того, что на высокооплачиваемые должности берут в основном мужчин. Об этом можно судить по более широкой форме графика выше отметки 100 тыс руб.

Для более подробного анализа построим трехмерный график, учитывающий еще опыт работы и образование. Этот график показывает, что при прочих равных условиях у мужчин зарплата выше. Самая высокая заработная плата в категории "Высшее образование". Соискатели с высшим образованием, судя по шкале опыта работы, работают дольше.

Очистка данных

1). Начнем с дубликатов в наших данных. Найдите **полные дубликаты** в таблице с резюме и удалите их.

In [ ]:
```
dupl_columns = list(hh_data.columns)
#dupl_columns.remove('id')

mask = hh_data.duplicated(subset=dupl_columns)
hh_duplicates = hh_data[mask]
display(f'Число найденных дубликатов: {hh_duplicates.shape[0]}')

# Создадим новую таблицу, которая будет версией исходной таблицы, очищенн
hh_df = hh_data.drop_duplicates(subset=dupl_columns)
print(f'Результирующее число записей: {hh_df.shape[0]}')
```

```
'Число найденных дубликатов: 161'
Результирующее число записей: 44583
```

2). Займемся пропусками. Выведите информацию **о числе пропусков** в столбцах.

In [ ]:
```
 #Функция для замены пропусков, обозначеных нулём
import numpy as np
def _nan_(arg):
  """Заменяем нули на специальное обозначение пропусков

  Args:
      arg (_type_): числовые значения
  Returns:
      _type_: np.NaN
  """
  if arg==0:
    return np.nan
  else: return arg

hh_data = hh_df.copy()
hh_data['Опыт работы (месяц)']=hh_df['Опыт работы (месяц)'].apply(_nan_)
cols_null = hh_df.isnull().sum()
cols_with_null = cols_null[cols_null>0].sort_values(ascending=False)
display(cols_with_null)
```

```
Опыт работы (месяц)                166
Последняя/нынешняя должность         2
Последнее/нынешнее место работы      1
dtype: int64
```

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

In [ ]:
```
#создаем копию исходной таблицы
drop_data = hh_data.copy()

#удаляем записи, в которых есть хотя бы 1 пропуск
drop_data = drop_data.dropna(how='any', subset=['Последняя/нынешняя должн

#создаем копию исходной таблицы
fill_data = drop_data.copy()
#создаем словарь имя столбца: число(признак) на который надо заменить про
values = {'Опыт работы (месяц)': fill_data['Опыт работы (месяц)'].median(
#заполняем пропуски в соответствии с заявленным словарем
fill_data = fill_data.fillna(values)
#выводим результирующую долю пропусков
display(round(fill_data['Опыт работы (месяц)'].mean()))
```

114

4). Мы добрались до ликвидации выбросов. Сначала очистим данные вручную. Удалите резюме, в которых указана заработная плата либо выше 1 млн. рублей, либо ниже 1 тыс. рублей.

In [ ]:
```
hh_df = fill_data.copy()
idx = hh_df[(hh_df['ЗП (руб)'] < 1000) | (hh_df['ЗП (руб)'] > 1000000)].i
hh_df = hh_df.drop(index=idx)
display(fill_data.shape[0] - hh_df.shape[0])
```

89

5). В процессе разведывательного анализа мы обнаружили резюме, в которых **опыт работы в годах превышал возраст соискателя**. Найдите такие резюме и удалите их из данных

In [ ]:
```
hh_data= hh_df.copy()
idx = hh_data[(hh_data['Возраст']*12) < hh_data['Опыт работы (месяц)']].i
hh_data = hh_data.drop(index=idx)
display(hh_df.shape[0] - hh_data.shape[0])
```

7

6). В результате анализа мы обнаружили потенциальные выбросы в признаке **"Возраст"**. Это оказались резюме людей чересчур преклонного возраста для поиска работы. Попробуйте построить распределение признака в **логарифмическом масштабе**. Добавьте к графику линии, отображающие **среднее и границы интервала метода трех сигм**. Напомним, сделать это можно с помощью метода axvline. Например, для построение линии среднего будет иметь вид:

```
histplot.axvline(log_age.mean(), color='k', lw=2)
```

В какую сторону асимметрично логарифмическое распределение? Напишите об этом в комментарии к графику. Найдите выбросы с помощью метода z-отклонения и удалите их из данных, используйте логарифмический масштаб. Давайте сделаем послабление на **1 сигму** (возьмите 4 сигмы) в **правую сторону**.

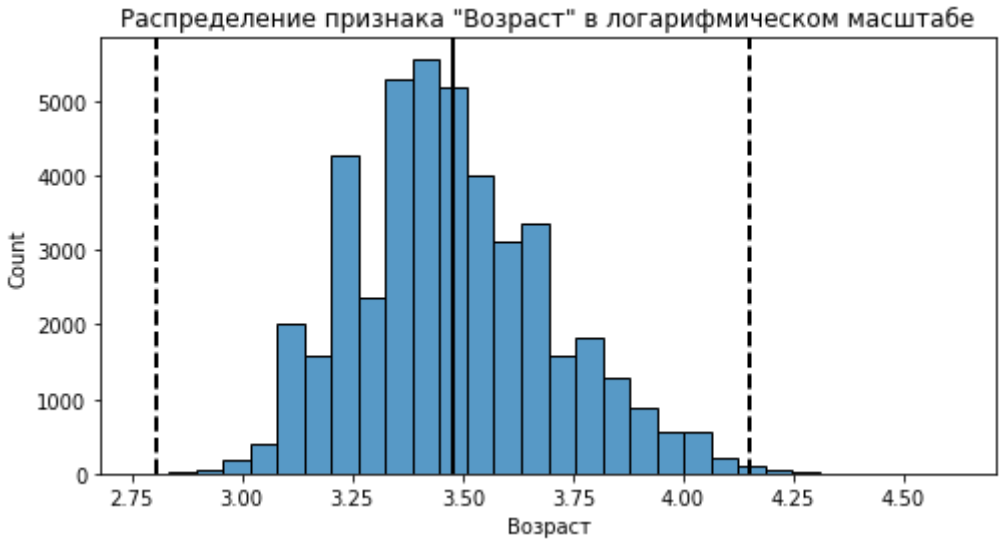
Выведите таблицу с полученными выбросами и оцените, с каким возрастом соискатели попадают под категорию выбросов?

In [ ]:
```
hh_df = hh_data.copy()

fig, ax = plt.subplots(1, 1, figsize=(8, 4))
log_age = np.log(hh_df['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()+ 3 * log_age.std(), color='k', ls='--', l
histplot.axvline(log_age.mean()- 3 * log_age.std(), color='k', ls='--', l
histplot.set_title('Распределение признака "Возраст" в логарифмическом ма

def outliers_z_score_mod(data, feature, log_scale=False, left=3, right=3)
    """Определение выбросов по методу z-отклонения и возвращение очищенны

    Args:
        data (_type_): набор данных
        feature (_type_): имя признака, на основе которого происходит пои
        log_scale (bool, optional): режим логарифмирования. По умолчанию
        left (int, optional): число сигм (стандартных отклонений) влево.
        right (int, optional): число сигм (стандартных отклонений) вправо

    Returns:
        _type_: число выбросов, число записей без выбросов
    """
    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
    outliers = data[(x < lower_bound) | (x > upper_bound)]
    cleaned = data[(x > lower_bound) & (x < upper_bound)]
    return outliers, cleaned

outliers, cleaned = outliers_z_score_mod(hh_df, 'Возраст', log_scale=True
display(f'Число выбросов по методу z-отклонения: {outliers.shape[0]}')
display(f'Результирующее число записей: {cleaned.shape[0]}')
display(outliers)
```

```
'Число выбросов по методу z-отклонения: 3'
'Результирующее число записей: 44482'
```

	Ищет работу на должность:	Последнее/нынешнее место работы	Последняя/нынешняя должность	Обновление резюме	Авто	Об
31137	Менеджер по работе с клиентами	ООО "ФёрстКэшКомпани"	Менеджер по работе с клиентами	2019-04-06	Не указано	
32950	Тестировщик игр	ООО ЖМЫХ	Тестировщик ПО	2019-04-09	Не указано	с
33654	Frontend-разработчик	Freelance	Frontend-разработчик	2019-04-19	Не указано	с

3 rows × 23 columns



Логарифмическое распределение асимметрично в правую сторону. В категорию выбрасов попали соискатели возрастом 15 и 100 лет.