None
Анализ европейских вакансий Data Scientist и Data Analyst с http://linkedin.com

# Table of Contents

# Описание проекта

**Цель проекта** - исследовать европейские вакансии DA и DS, спарсенные с http://linkedin.com, для выявления наиболее популярных навыков, предъявляемых в требованиях к кандидатам, получения географического среза вакансий, построения топ сфер деятельности и компаний, в которых ищут аналитиков, а также возможности удаленной работы.

## Техническое задание

**1. Распарсить предоставленные csv файлы** с помощью BS 4, создав следующие признаки:

- наименование вакансии;
- город;
- страна;
- тип занятости (online, hybride, on-site);
- компания;
- размер компании (количество работников);
- сфера деятельности компании;
- требуемые хард скилы;
- дата публикации вакансии (для DA время завершения парсинга 2022/09/07 17:00:00, для DS - 2022/09/08 22:00:00);
- количество кандидатов на вакансию.

**2. Подготовить данные к визуализации:**

- Отфильтровать датафрейм DA с оставлением только релевантных вакансий;
- Удалить дубликаты;
- Удалить ненужные признаки

**3. Визуализировать данные (дашборд):**

Дашборд должен содержать:

- фильтры (по стране и по типу занятости);
- количество вакансий (абсолютные значения) – индикатор;
- количество вакансий по странам (относительные значения) — horizontal bar chart;
- количество вакансий по городам - map;
- тип занятости — pie chart;
- список нанимающих компаний с указанием количества вакансий, отсортированный в порядке убывания — heat map;
- ТОП 10 сфер деятельности компаний, которые нанимают аналитиков — barchart;
- размер компаний и количество вакансий — pie chart
- хард скилы — опционально;
- зависимость количества кандидатов на вакансию от даты публикации объявления — линейный график

Дашборд доступен по ссылке

# Предобработка данных

## Извлечение данных из html

Импортируем библиотеки, загрузим датафреймы, исследуем элементы html-страницы с вакансией (первой попавшейся) для определения тегов, необходимых для извлечения требуемых данных.

In [1]:

```
import pandas as pd
from bs4 import BeautifulSoup
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
import requests
import datetime
from datetime import date
from datetime import timedelta
```

In [2]:

```
da = pd.read_csv('da_vac.csv')
ds = pd.read_csv('ds_vac.csv')
```

In [3]:

```
da['spec'] = 'DA'
ds['spec'] = 'DS'
```

Отметим для удобства теги из структуры страницы, в которых содержатся нужные нам данные, далее поочередно будем их извлекать

In [4]:

```
#- наименование вакансии - h2
#- город - <span class="jobs-unified-top-card__bullet">
#- страна - <span class="jobs-unified-top-card__bullet">
#- тип занятости (online, hybride, on-site) - <span class="jobs-unified-top-card__workplace-type">
#- компания <span class="jobs-unified-top-card__company-name">
#- размер компании (количество работников)  <div class="mt5 mb2"> - <li class="jobs-unified-top-card__job-ins
#- сфера деятельности компании - там же
#- требуемые хард скилы 'div', {'id':'job-details'}
#- дата публикации вакансии <span class="jobs-unified-top-card__posted-date"> (опосредованно)
#- количество кандидатов на вакансию <span class="jobs-unified-top-card__applicant-count"
```

In [5]:

```
display(da.info())
display(ds.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 772 entries, 0 to 771
Data columns (total 3 columns):
Unnamed: 0    772 non-null int64
html          772 non-null object
spec          772 non-null object
dtypes: int64(1), object(2)
memory usage: 18.2+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 939 entries, 0 to 938
Data columns (total 3 columns):
Unnamed: 0    939 non-null int64
html          939 non-null object
spec          939 non-null object
dtypes: int64(1), object(2)
memory usage: 22.1+ KB
None
```

### Link

Несмотря на отсутствие в ТЗ вытащим для каждой вакансии ссылку на нее - она будет уникальным идентификатором

In [6]:

```
da['link'] = da['html'].apply(lambda x: "https://linkedin.com" + BeautifulSoup(x).find('a').get('href'))
ds['link'] = ds['html'].apply(lambda x: "https://linkedin.com" + BeautifulSoup(x).find('a').get('href'))
```

### Наименование вакансии

In [7]:

```
da['title'] = da['html'].apply(lambda x:  BeautifulSoup(x).find('h2').text.strip())
ds['title'] = ds['html'].apply(lambda x:  BeautifulSoup(x).find('h2').text.strip())
```

### Регион

для дальнейшего выделения страны и города

In [8]:

```
def get_geo(cell):
    try:
        return BeautifulSoup(cell).find('span', class_ = 'jobs-unified-top-card__bullet').text.strip()
    except:
        return np.nan
```

In [9]:

```
da['geo'] = da['html'].apply(get_geo)
ds['geo'] = ds['html'].apply(get_geo)
```

### Тип занятости

(online, hybride, on-site)

In [10]:

```
def get_workplace_type(cell):
    try:
        return BeautifulSoup(cell).find('span', class_ = 'jobs-unified-top-card__workplace-type').text.strip()
    except:
```

```
        return np.nan
```

In [11]:

```
da['workplace_type'] = da['html'].apply(get_workplace_type)
ds['workplace_type'] = ds['html'].apply(get_workplace_type)
```

### Название компании

In [12]:

```
def get_company_name(cell):
    try:
        return BeautifulSoup(cell).find('span', class_ = 'jobs-unified-top-card__company-name').text.strip()
    except:
        return np.nan
```

In [13]:

```
da['company_name'] = da['html'].apply(get_company_name)
ds['company_name'] = ds['html'].apply(get_company_name)
```

### Размер и сфера деятельнности компании

In [14]:

```
def get_company_insights(cell):
    try:
        return BeautifulSoup(cell).find('div', class_ = 'mt5 mb2').find_all('li')[1].text.strip()
    except:
        return np.nan
```

In [15]:

```
da['company_insights'] = da['html'].apply(get_company_insights)
da['company_size'] = da['company_insights'].str.split(' · ', expand=True)[0]
da['company_sphere'] = da['company_insights'].str.split(' · ', expand=True)[1]

ds['company_insights'] = ds['html'].apply(get_company_insights)
ds['company_size'] = ds['company_insights'].str.split(' · ', expand=True)[0]
ds['company_sphere'] = ds['company_insights'].str.split(' · ', expand=True)[1]
```

### Skills

Для выяснения самых популярных hard skills составим их список и смэтчим с ним описание вакансий

In [16]:

```
skills_da_ds = ([' bi ', ' ml ', ' cv ', 'datahub', 'api', 'github', 'git', 'google analytics', 'adobe analyt
            'gitlab', 'erwin', 'hadoop', 'spark', 'hive', 'databricks', 'aws', 'gcp', 'azure','excel', 'sql',
            'redshift', 'bigquery', 'bigdata', 'snowflake',  'hana', 'grafana', 'kantar', 'spss',
           'asana', 'basecamp', 'jira', 'dbeaver','trello', 'miro', 'figma', 'salesforce',
           'rapidminer', 'thoughtspot',  'power point',  'docker', 'jenkins','integrate.io', 'talend', 'apache
             'azure data factory','xplenty','skyvia','iri voracity','xtract.io','dataddo', 'ssis',
             'hevo data','informatica','oracle data integrator','k2view','cdata sync','querysurge',
             'rivery', 'dbconvert', 'alooma', 'stitch', 'fivetran', 'matillion','streamsets','blendo',
             'iri voracity','logstash', 'etleap', 'singer', 'apache camel','actian', 'airflow', 'luidgi', 'dat
           'python', 'pandas',  'vba', 'scala', ' r ', 'java script', 'julia', 'matplotlib', 'matlab', 'java',
           'data studio', 'tableau', 'looker', 'powerbi', 'cognos', 'microstrategy', 'spotfire', 'seaborn'
             'sap business objects','microsoft sql server', 'oracle business intelligence', 'yellowfin',
             'webfocus','sas visual analytics', 'targit', 'izenda',  'sisense', 'statsbot', 'panorama', 'inets
             'birst', 'domo', 'metabase', 'redash', 'power bi', 'alteryx', 'dataiku', 'qlik sense', 'qlikview
              'artificial intelligence', 'a/b testing', 'a/b test', 'airflow', 'computer vision', 'machine lea
              'dashboard', 'deep learning', 'html', 'keras', 'stat', 'statistics', 'docker', 'numpy',
              'nlp', 'postgresql', 'mysql', 'power point', 'trello', 'vba', 'tensorflow', 'keras',
              'statsmodels', 'sklearn', 'plotly', 'visualisation', 'analitycal skills', 'coding', 'parsing',
                 'beautifulsoup', 'kmeans','k-means', 'tulips', 'stat', 'pvalue', 'cycle', 'function', 'data p
                 'statistical', 'supervised learning', 'linear algebra', 'numerical methods', 'dataframe', 't
                 'data extraction', 'unsupervised learning', 'forecast', 'predictionss', 'power point',
                 'ms access', 'anaconda', 'xml', 'twython', 'p-value', 'p value', 'gradient boost', 'regressi
                 'tree', 'tensor', 'neuro', 'scikit', 'mapreduce', 'jupiter', 'boost', 'graph'
          ])
```

In [17]:

```
def get_skills(cell):
    list_skills = []
    for skill in skills_da_ds:
        if skill in cell.lower().replace('powerbi', 'power bi'):
            list_skills.append(skill)
    return list_skills
```

In [18]:

```
da['description'] = da['html'].apply(lambda x: BeautifulSoup(x).find('div', {'id':'job-details'}).text.strip()
ds['description'] = ds['html'].apply(lambda x: BeautifulSoup(x).find('div', {'id':'job-details'}).text.strip()
```

In [19]:

```
da['skills'] = da['description'].apply(get_skills)
ds['skills'] = ds['description'].apply(get_skills)
```

### Дата публикации

Т.к. дата публикации указана опосредовано (как время до времени парсинга), для ее уточнения проведем несколько действий

In [20]:

```
def get_time_ago(cell):
    try:
        return BeautifulSoup(cell).find('span', class_ = 'jobs-unified-top-card__posted-date').text.strip()
    except:
        return np.nan
```

In [21]:

```
da['days_before'] = da['html'].apply(get_time_ago)
ds['days_before'] = ds['html'].apply(get_time_ago)

display(da['days_before'].unique())
display(ds['days_before'].unique())
```

Поле содержит X minutes, hour или hours, week (weeks), days ago. При этом дата сбора данных, от которой происходит отсчет этого "назад" - 2022/09/07 17:00:00. Напишем функцию, которая по разнице со временем парсинга определит исходную дату публикации

In [22]:

```
def date_da(cell):
    if 'minutes' in cell:
        for i in cell.split():
            if i.isdigit():
                cell = datetime.datetime(2022, 9, 7, 17) - timedelta(minutes=int(i))
                return cell
    elif 'hours' in cell or 'hour' in cell:
        for i in cell.split():
            if i.isdigit():
                cell = datetime.datetime(2022, 9, 7, 17) - timedelta(hours=int(i))
                return cell
    elif 'weeeks' in cell or 'week' in cell:
        for i in cell.split():
            if i.isdigit():
                cell = datetime.datetime(2022, 9, 7, 17) - timedelta(weeks=int(i))
                return cell
    else:
        for i in cell.split():
            if i.isdigit():
                cell = datetime.datetime(2022, 9, 7, 17) - timedelta(days=int(i))
                return cell
```

In [23]:

```
da['post_date'] = da['days_before'].apply(date_da)
```

In [24]:

```
da['post_date'].unique()
```

Out[24]:

In [25]:

```
def date_ds(cell):
    if 'minutes' in cell:
        for i in cell.split():
            if i.isdigit():
                cell = datetime.datetime(2022, 9, 8, 22) - timedelta(minutes=int(i))
                return cell
    elif 'hours' in cell or 'hour' in cell:
        for i in cell.split():
            if i.isdigit():
                cell = datetime.datetime(2022, 9, 8, 22) - timedelta(hours=int(i))
                return cell
    elif 'weeeks' in cell or 'week' in cell:
        for i in cell.split():
            if i.isdigit():
                cell = datetime.datetime(2022, 9, 8, 22) - timedelta(weeks=int(i))
                return cell
```

```
    else:
        for i in cell.split():
            if i.isdigit():
                cell = datetime.datetime(2022, 9, 8, 22) - timedelta(days=int(i))
                return cell
```

In [26]:

```
ds['post_date'] = ds['days_before'].apply(date_ds)
```

### Число кандидатов на вакансию

In [27]:

```
def get_candidates(cell):
    try:
        return BeautifulSoup(cell).find('span', class_ = 'jobs-unified-top-card__applicant-count').text.strip
    except:
        return np.nan
```

In [28]:

```
da['candidates'] = da['html'].apply(get_candidates)
ds['candidates'] = ds['html'].apply(get_candidates)
```

Посмотрим, какие даные получены и надо ли их дополнительно модифицировать

In [29]:

```
da.head(2)
```

Out[29]:

| | Unnamed: 0 | html | spec | link | title | geo | workplace_type | company_name | company_ins |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | \n \n <div class="\n jobs-deta... | DA | https://linkedin.com/jobs/view/3258155313/?alt... | Stage - Assistant Ingénieur Qualité - Beyrand ... | Limoges, Nouvelle-Aquitaine, France | On-site | Hermès | 10, employees · I Luxury Goods |
| 1 | 1 | \n \n <div class="\n jobs-deta... | DA | https://linkedin.com/jobs/view/3249651625/?alt... | développeur matlab/simulink, secteur automobil... | Toulouse, Occitanie, France | On-site | AUSY | 5,001-1 employee Services a |

In [30]:

```
ds.head(2)
```

Out[30]:

| | Unnamed: 0 | html | spec | link | title | geo | workplace_type | company_name | company_ins |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | \n \n <div class="\n jobs-deta... | DS | https://linkedin.com/jobs/view/3255084020/?alt... | Online Data Analyst \| French Speaker | Provence-Alpes-Côte d'Azur, France | Remote | TELUS International AI Data Solutions | 10, employee Services a Consu |
| 1 | 1 | \n \n <div class="\n jobs-deta... | DS | https://linkedin.com/jobs/view/3249651625/?alt... | développeur matlab/simulink, secteur automobil... | Toulouse, Occitanie, France | On-site | AUSY | 5,001-1 employee Services a |

### Страна, город

Выделим из региона страну и город

In [31]:

```
da['geo'].unique()
```

Out[31]:

In [32]:

```
def get_city(cell):
    if len(cell.split(',')) > 1:
```

```
        return cell.split(',')[0].strip()
    elif "Metropolitan" in cell or "Greater" in cell:
        return cell.replace('Greater', '').replace('Metropolitan', '').replace('Area', '').replace('Region','
    else:
        return np.nan
```

In [33]:

```
def get_country(cell):
    if len(cell.split(',')) > 1:
        return cell.split(',')[-1].strip()
    elif "Metropolitan" in cell or "Greater" in cell or "Region" in cell:
        return np.nan
    else:
        return cell
```

In [34]:

```
da['city'] = da['geo'].apply(get_city)
da['country'] = da['geo'].apply(get_country)

ds['city'] = ds['geo'].apply(get_city)
ds['country'] = ds['geo'].apply(get_country)
```

In [35]:

```
da[da['country'].isna()].head(2)
```

Out[35]:

| | Unnamed: 0 | html | spec | link | title | geo | workplace_type | company_name | company_insights |
|---|---|---|---|---|---|---|---|---|---|
| 9 | 9 | \n \n <div class="\n jobs-deta... | DA | https://linkedin.com/jobs/view/3247820547/?alt... | Data Engineer (with Python) | Wroclaw Metropolitan Area | Hybrid | Capgemini | 10,001+ employees · IT Services and IT Consulting |
| 11 | 11 | \n \n <div class="\n jobs-deta... | DA | https://linkedin.com/jobs/view/3257247449/?alt... | Data Analyst | Przemyśl Metropolitan Area | Remote | TELUS International AI Data Solutions | 10,001+ employees · IT Services and IT Consulting |

In [36]:

```
len(da[da['country'].isna()])
```

Out[36]:

```
46
```

In [37]:

```
len(ds[ds['country'].isna()])
```

Out[37]:

```
25
```

Загрузим справочник стран, чтобы подтянуть отсутствующие.

In [38]:

```
countries = pd.read_csv('worldcities.csv')
countries.head(2)
```

Out[38]:

| | city | city_ascii | lat | lng | country | iso2 | iso3 | admin_name | capital | population | id |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Tokyo | Tokyo | 35.6839 | 139.7744 | Japan | JP | JPN | Tōkyō | primary | 39105000.0 | 1392685764 |
| 1 | Jakarta | Jakarta | -6.2146 | 106.8451 | Indonesia | ID | IDN | Jakarta | primary | 35362000.0 | 1360771077 |

In [39]:

```
countries.loc[countries['city'].str.contains('Athens') == True]
```

Out[39]:

На примере города "Athens" видно, что при прямом объединении таблиц (da/ds и справочника стран) возникнет проблема, т.к. в разных странах существуют города с одинаковым названием, а координат у нас нет. Удалим из справочника американские города, т.к. в наших датасетах только европейские вакансии

In [40]:

```
countries = countries[countries['country'] != 'United States']
```

Учитываяя небольшое количество пропусков в распарсенном поле, создадим промежуточный справочник в полуручном режиме.

In [41]:

```
countries_reduced = countries[['city', 'city_ascii', 'country']]
```

In [42]:

```
nans = da[da['country'].isna()][['geo', 'city']].drop_duplicates().sort_values(by='geo')
nans_upd = nans.merge(countries_reduced, how='left', left_on = 'city', right_on= 'city', suffixes=('_geo', '_c
nans_upd
```

Out[42]:

In [43]:

```
nans_upd = nans_upd.loc[~nans_upd['country'].isin(['Venezuela', 'Philippines', 'Brazil', 'Colombia', 'Ecuador
```

In [44]:

```
nans_upd
```

Out[44]:

Дополним словари, подтянем оставшиеся данные

In [45]:

In [46]:

```
da.loc[da['country'].isna(), 'country'] = da.loc[da['country'].isna(), 'geo'].replace(country_dict_2)
```

In [47]:

```
da[da.country.isna()]
```

Out[47]:

| | Unnamed: 0 | html | spec | link | title | geo | workplace_type | company_name | company_insights | company_size | company_sphere | description | skills | d |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

In [48]:

```
# Аналогично, заполним пропуски во втором датафрейме
ds.loc[ds['country'].isna(), 'country'] = ds.loc[ds['country'].isna(), 'geo'].replace(country_dict_2)
ds[ds.country.isna()]
```

Out[48]:

| | Unnamed: 0 | html | spec | link | title | geo | workplace_type | company_name | company_insights | company_size | company_sphere | description | skills | d |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

**Все нужные, по ТЗ, признаки выделены - переходим к обработке.**

# Предобработка данных

Сначала отфильтуем вакансии, выделив только те, которые относятся к искомым. Затем проведем предобработку данных для визуализации (обработаем пропуски, удалим дубликаты).

In [49]:

```
da.title.unique()
```

Out[49]:

При просмотре списка "по диагонали" видно, что в нем есть как релевантные, так и совсем не относящиеся к DA. Кроме того, в списке присутствуют вакансии DS. Составим список для фильтрации

In [50]:

```
da[da['title'].str.lower().str.contains('data science|data scientist|machine learning')]
```

Out[50]:

In [51]:

```
# сначала посмотрим, какие аналитики есть в датасете, чтобы решить, нужно ли усложнять условие для фильтрации
da[da['title'].str.lower().str.contains('analyst')]['title'].unique()
```

Out[51]:

In [52]:

```
#Да, в списке есть бизнес-аналитики, которые нас не интересуют. Поэтому составим перечень для фильтрации
da_pat = 'data \w+ analyst|analyst \w+ data|financial analyst|bi analyst|sap analyst|product analyst|a/b|data

da_filtered = da[da['title'].str.lower().str.contains(da_pat, regex=True)]
```

In [53]:

```
len(da_filtered)
```

Out[53]:

```
318
```

Проведем аналогичную работу с файлом DS, выделим оттуда вакансии, которые относятся к аналитикам, а не DS, затем дополним каждый из списков вакансий таковыми из "соседнего" файла

In [54]:

```
ds['title'].unique()
```

Out[54]:

In [55]:

```
ds_pat = 'data science|data scientist|machine learning|nlp|cv|ml|computer vision|predict|DS|data \w+ scien|nat
```

In [56]:

```
ds[ds['title'].str.lower().str.contains(ds_pat, regex=True)]['title'].unique()
```

Out[56]:

In [57]:

```
ds_filtered = ds[ds['title'].str.lower().str.contains(ds_pat, regex=True)]
```

In [58]:

```
len(ds_filtered)
```

Out[58]:

```
278
```

In [59]:

```
# вакансии аналитиков в датасете DS и наоборот
ds_from_da = da[da['title'].str.lower().str.contains(ds_pat, regex=True)]

da_from_ds = ds[ds['title'].str.lower().str.contains(da_pat, regex=True)]
```

In [60]:

```
display(len(ds_from_da))
display(len(da_from_ds))
```

```
60
86
```

In [61]:

```
da_added = pd.concat([da_filtered, da_from_ds], ignore_index=True)
```

In [62]:

```
da_added.head(2)
```

Out[62]:

| | Unnamed: 0 | html | spec | link | title | geo | workplace_type | company_name | company_insights | co |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 | \n \n <div class="\n jobs-deta... | DA | https://linkedin.com/jobs/view/3248499929/?alt... | Online Data Analyst | Skara, Vastra Gotaland County, Sweden | Remote | TELUS International AI Data Solutions | 10,001+ employees · IT Services and IT Consulting | |
| 1 | 3 | \n \n <div class="\n jobs-deta... | DA | https://linkedin.com/jobs/view/3248879065/?alt... | Online Data Analyst - Belgium | West Flanders, Flemish Region, Belgium | Remote | TELUS International | 10,001+ employees · IT Services and IT Consulting | |

In [63]:

```
len(da_added)
```

Out[63]:

```
404
```

In [64]:

```
ds_added = pd.concat([ds_filtered, ds_from_da], ignore_index=True)
len(ds_added)
```

Out[64]:

```
338
```

Оставим те столбцы, которые нам нужны, затем исследуем данные более подробно и проведем необходимые манипуляции.

In [65]:

```
da_short = da_added[['title', 'workplace_type', 'country', 'city', 'company_name', 'company_size', 'company_sp
da_short.head(2)
```

Out[65]:

| | title | workplace_type | country | city | company_name | company_size | company_sphere | skills | post_date | candidates | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Online Data Analyst | Remote | Sweden | Skara | TELUS International AI Data Solutions | 10,001+ employees | IT Services and IT Consulting | [git, machine learning] | 2022-09-01 17:00:00 | 12 applicants | https://linkedin.cor |
| 1 | Online Data Analyst - Belgium | Remote | Belgium | West Flanders | TELUS International | 10,001+ employees | IT Services and IT Consulting | [git, machine learning, graph] | 2022-09-01 17:00:00 | 11 applicants | https://linkedin.cor |

In [66]:

```
ds_short = ds_added[['title', 'workplace_type', 'country', 'city', 'company_name', 'company_size', 'company_sp
ds_short.head(2)
```

Out[66]:

| | title | workplace_type | country | city | company_name | company_size | company_sphere | skills | post_date | candidates | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Data Scientist / Data Engineer débutant(e) | On-site | France | Pau | TotalEnergies | 10,001+ employees | Oil and Gas | [git, docker, python, computer vision, machine... | 2022-09-02 22:00:00 | NaN | https://linkedin.( |
| 1 | Data Scientist | On-site | France | Paris | BCG GAMMA | 1,001-5,000 employees | Business Consulting and Services | [api, git, spark, artificial intelligence, mac... | 2022-09-05 22:00:00 | 192 applicants | https://linkedin.( |

In [67]:

```
# получим общие данные. Для удобства, напишем функцию
def info(table):
    display(table.head(2))
    print('')
    print(table.info())
    print('')
    print('')

    #print('Число дубликатов', table.duplicated().sum())
    print('')
    print('Число пропусков')
    print(table.isna().sum())
    print('')
    display(table.describe())
```

In [68]:

```
info(da_short)
```

In [69]:

```
da_short.drop(columns=['skills']).duplicated().sum()
```

Out[69]:

```
0
```

In [70]:

```
info(ds_short)
```

In [71]:

```
ds_short.drop(columns=['skills']).duplicated().sum()
```

Out[71]:

```
0
```

In [72]:

```
da_short['workplace_type'].unique()
```

Out[72]:

```
array(['Remote', 'On-site', nan, 'Hybrid'], dtype=object)
```

In [73]:

```
def sign(table, column):
    '''
    Простая функция для быстрого просмотра уникальных значений признаков
    '''
    display(column)
    display(table[column].unique())
```

In [74]:
```
columns = ['workplace_type', 'company_size', 'company_sphere', 'candidates']

for col in columns:
    sign(da_short, col)
```

In [75]:
```
da_short[da_short['company_size'] == 'See how you compare to 14 applicants. Unlock more Premium insights for
```

Out[75]:
```
170    https://linkedin.com/jobs/view/3251175567/?alt...
Name: link, dtype: object
```

**Финальные необходимые преобразования:**

1. Для удобства анализа числа кандидатов на вакансию от даты публикации перейдем к числовому формату данного поля;
2. В сферах компаний удалим лишние пробелы, которые появились при разнесении insights по столбцам
3. В размер компаний пробрались посторонние данные, не имеющие отношения к числу сотрудников - удалим их
4. В категориальных столбцах заменим Nan на "Not specified"

Чтобы анализировать skills, их надо выделить с помощью .explode(), но тогда датасет искусственно удлиннится, чего нам не нужно в анализе остальных признаков. Поэтому для анализа необходимых скиллов выделим их в отдельные фреймы.

Финально перед переходом к построению дашборда посмотрим ключевые характеристики из ТЗ с базовой визуализацией.

In [76]:
```
pd.options.mode.chained_assignment = None

da_short['candidates'] = pd.to_numeric(da_short['candidates'].replace(to_replace=[' applicants', ' applicant'
da_short = da_short.replace('', np.nan)
da_short.head(2)
```

Out[76]:

| | title | workplace_type | country | city | company_name | company_size | company_sphere | skills | post_date | candidates | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Online Data Analyst | Remote | Sweden | Skara | TELUS International AI Data Solutions | 10,001+ employees | IT Services and IT Consulting | [git, machine learning] | 2022-09-01 17:00:00 | 12.0 | https://linkedin.cor |
| 1 | Online Data Analyst - Belgium | Remote | Belgium | West Flanders | TELUS International | 10,001+ employees | IT Services and IT Consulting | [git, machine learning, graph] | 2022-09-01 17:00:00 | 11.0 | https://linkedin.cor |

In [77]:
```
pd.options.mode.chained_assignment = None
da_short['company_sphere'] = da_short['company_sphere'].str.strip()
da_short['company_sphere'].unique()
```

Out[77]:

In [78]:
```
da_short['company_size'].loc[da_short['company_size'].str.contains('employees') == False] = 'Not specifieds'
```

In [79]:
```
da_short['company_size'] = da_short['company_size'].fillna('Not specified')
da_short['company_size'] = da_short['company_size'].str.strip()
da_short['company_size'].unique()
```

Out[79]:
```
array(['10,001+ employees', '201-500 employees', '1,001-5,000 employees',
       '51-200 employees', '5,001-10,000 employees',
       '501-1,000 employees', '11-50 employees', '1-10 employees',
       'Not specifieds', 'Not specified'], dtype=object)
```

In [80]:
```
da_short['workplace_type'] = da_short['workplace_type'].fillna('Not specified')
da_short['company_sphere'] = da_short['company_sphere'].fillna('Not specified')
da_short['company_name'] = da_short['company_name'].fillna('Not specified')
da_short['city'] = da_short['city'].fillna('Not specified')
```

In [81]:
```
columns = ['workplace_type', 'company_size', 'company_sphere', 'company_name']

for col in columns:
    sign(da_short, col)
```

In [82]:

```
ds_short['candidates'].unique()
```

Out[82]:

In [83]:

```
da_short.info()
```

In [84]:

```
# аналогично, для второго датафрейма

pd.options.mode.chained_assignment = None
ds_short = ds_short.replace('', np.nan)
ds_short['candidates'] = pd.to_numeric(ds_short['candidates'].replace(to_replace=[' applicants', ' applicant'
```

In [85]:

```
pd.options.mode.chained_assignment = None
ds_short['company_sphere'] = ds_short['company_sphere'].str.strip()
```

In [86]:

```
ds_short['company_size'].loc[ds_short['company_size'].str.contains('employees') == False] = 'Not specified'
ds_short['company_size'] = ds_short['company_size'].fillna('Not specified')
ds_short['company_size'] = ds_short['company_size'].str.strip()
ds_short['company_size'].unique()
```

Out[86]:

```
array(['10,001+ employees', '1,001-5,000 employees', '1-10 employees',
       '201-500 employees', '51-200 employees', '5,001-10,000 employees',
       '11-50 employees', 'Not specified', '501-1,000 employees'],
      dtype=object)
```

In [87]:

```
ds_short['workplace_type'] = ds_short['workplace_type'].fillna('Not specified')
ds_short['company_sphere'] = ds_short['company_sphere'].fillna('Not specified')
ds_short['company_name'] = ds_short['company_name'].fillna('Not specified')
ds_short['city'] = ds_short['city'].fillna('Not specified')
```

In [88]:

```
columns = ['workplace_type', 'company_size', 'company_sphere', 'company_name']

for col in columns:
    sign(ds_short, col)
```

In [89]:

```
ds_short.head(2)
```

Out[89]:

| | title | workplace_type | country | city | company_name | company_size | company_sphere | skills | post_date | candidates | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Data Scientist / Data Engineer débutant(e) | On-site | France | Pau | TotalEnergies | 10,001+ employees | Oil and Gas | [git, docker, python, computer vision, machine... | 2022-09-02 22:00:00 | NaN | https://linkedin.c |
| 1 | Data Scientist | On-site | France | Paris | BCG GAMMA | 1,001-5,000 employees | Business Consulting and Services | [api, git, spark, artificial intelligence, mac... | 2022-09-05 22:00:00 | 192.0 | https://linkedin.c |

In [90]:

```
ds_short.info()
```

In [91]:

```
# формируем финальные датасеты. Т.к. в Tableau нужны будут фильтры по стране и типу занятости, сохраним их в 
da_skills = da_short[['skills', 'country', 'workplace_type', 'link']]
da_fin = da_short.drop(columns = ['skills'])

ds_skills = ds_short[['skills', 'country', 'workplace_type', 'link']]
ds_fin = ds_short.drop(columns = ['skills'])
```

In [92]:

```
# убедимся в отстутствии пропусков кроме числового поля с числом кандидатов на вакансию
def nans(table):
    print('Число пропусков')
    print(table.isna().sum())
```

In [93]:

```
for tab in (da_skills, da_fin, ds_skills, ds_fin):
    nans(tab)
```

In [94]:

```
display(da_skills.shape)
display(ds_skills.shape)
display(da_fin.shape)
display(ds_fin.shape)
```

```
(404, 4)
(338, 4)
(404, 11)
(338, 11)
```

In [95]:

```
da_skills_expl = da_skills.explode('skills')
ds_skills_expl = ds_skills.explode('skills')
```

In [96]:

```
da_skills_expl
```

Out[96]:

In [97]:

```
# на случай, если захочется сравнивать DA и DS вакансии между собой
df_total = pd.concat([da_fin, ds_fin], ignore_index=True)
df_total.head(2)
```

Out[97]:

| | title | workplace_type | country | city | company_name | company_size | company_sphere | post_date | candidates | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Online Data Analyst | Remote | Sweden | Skara | TELUS International AI Data Solutions | 10,001+ employees | IT Services and IT Consulting | 2022-09-01 17:00:00 | 12.0 | https://linkedin.com/jobs/viev |
| 1 | Online Data Analyst - Belgium | Remote | Belgium | West Flanders | TELUS International | 10,001+ employees | IT Services and IT Consulting | 2022-09-01 17:00:00 | 11.0 | https://linkedin.com/jobs/viev |

In [98]:

```
nans(df_total)
```

```
Число пропусков
title              0
workplace_type     0
country            0
city               0
company_name       0
company_size       0
company_sphere     0
post_date          0
candidates       101
link               0
spec               0
dtype: int64
```

In [99]:

```
df_total.shape
```

Out[99]:

```
(742, 11)
```

In [100]:

```
da_skills_expl['spec'] = 'DA'
ds_skills_expl['spec'] = 'DS'
skills_total = pd.concat([da_skills_expl, ds_skills_expl], ignore_index=True)
skills_total.head(2)
```

Out[100]:

| | skills | country | workplace_type | link | spec |
|---|---|---|---|---|---|
| 0 | git | Sweden | Remote | https://linkedin.com/jobs/view/3248499929/?alt... | DA |
| 1 | machine learning | Sweden | Remote | https://linkedin.com/jobs/view/3248499929/?alt... | DA |

In [101]:

```
skills_total.shape
```

Out[101]:
```
(5954, 5)
```

# Визуализация в Python

Проведем короткий исследовательский анализ перед построением дашборда в соответствии с ТЗ (по ключевым параметрам ТЗ)

In [102]:

```
info(df_total)
```

In [103]:

```
df_total.loc[df_total['link'] == 'Not specified']
```

Out[103]:

| title | workplace_type | country | city | company_name | company_size | company_sphere | post_date | candidates | link | spec |
|---|---|---|---|---|---|---|---|---|---|---|

### Количество вакансий по странам

In [104]:

```
country_vac = df_total.pivot_table(index='country', columns='spec'
                                    , values='link', aggfunc='count', margins=True, margins_name='DA+DS').reset
country_vac = country_vac.sort_values(by='DA+DS', ascending=False
                                    ).loc[country_vac['country']!='DA+DS']
```

In [105]:

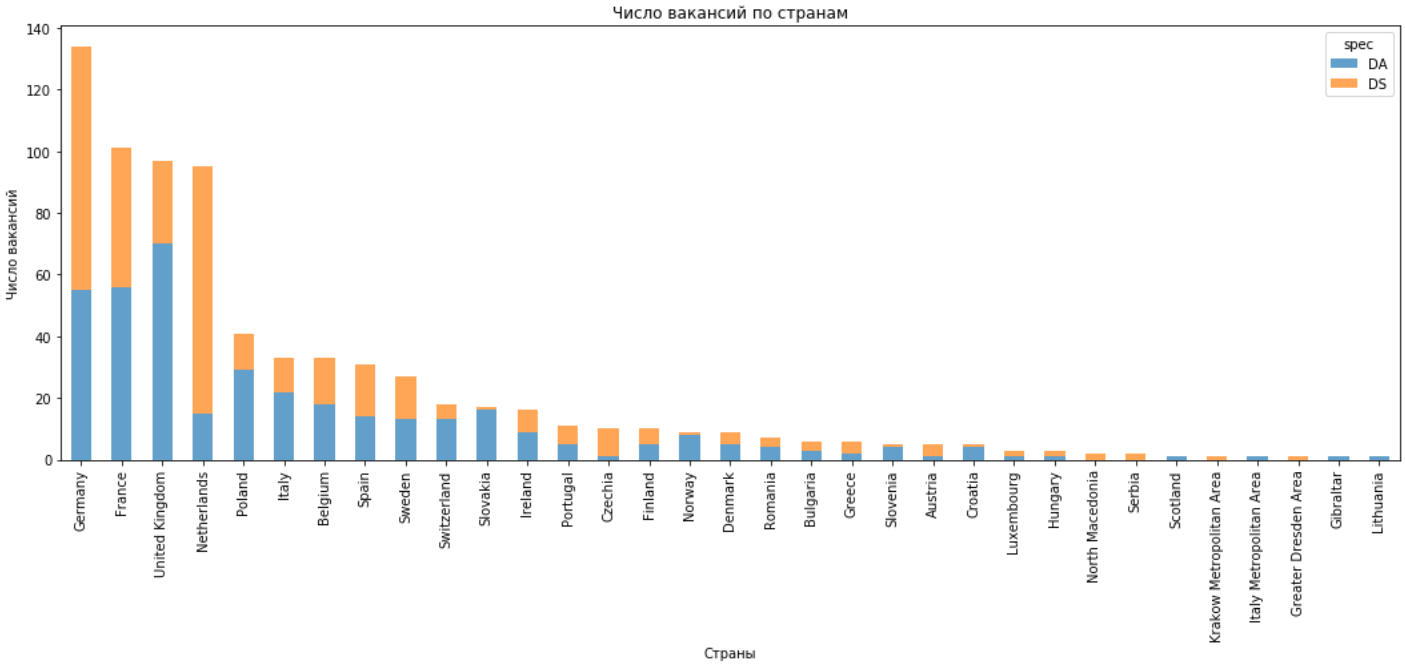
```
country_vac.head(20)
```

Out[105]:

| spec | country | DA | DS | DA+DS |
|---|---|---|---|---|
| 8 | Germany | 55.0 | 79.0 | 134 |
| 7 | France | 56.0 | 45.0 | 101 |
| 32 | United Kingdom | 70.0 | 27.0 | 97 |
| 19 | Netherlands | 15.0 | 80.0 | 95 |
| 22 | Poland | 29.0 | 12.0 | 41 |
| 14 | Italy | 22.0 | 11.0 | 33 |
| 1 | Belgium | 18.0 | 15.0 | 33 |
| 29 | Spain | 14.0 | 17.0 | 31 |
| 30 | Sweden | 13.0 | 14.0 | 27 |
| 31 | Switzerland | 13.0 | 5.0 | 18 |
| 27 | Slovakia | 16.0 | 1.0 | 17 |
| 13 | Ireland | 9.0 | 7.0 | 16 |
| 23 | Portugal | 5.0 | 6.0 | 11 |
| 4 | Czechia | 1.0 | 9.0 | 10 |
| 6 | Finland | 5.0 | 5.0 | 10 |
| 21 | Norway | 8.0 | 1.0 | 9 |
| 5 | Denmark | 5.0 | 4.0 | 9 |
| 24 | Romania | 4.0 | 3.0 | 7 |
| 2 | Bulgaria | 3.0 | 3.0 | 6 |
| 11 | Greece | 2.0 | 4.0 | 6 |

In [106]:

```
ax = country_vac[['country','DA','DS']].plot.bar(x='country', stacked=True, rot=90, figsize=(18,6), alpha=0.7)
ax.title.set_text('Число вакансий по странам')
ax.set(ylabel='Число вакансий')
ax.set(xlabel='Страны')
plt.show()
```

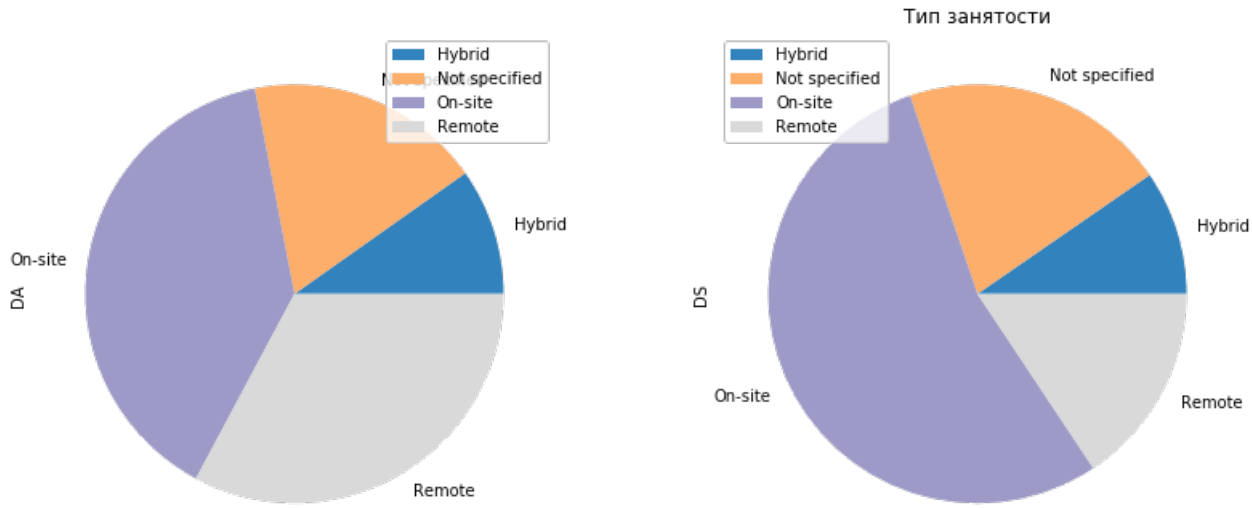



**Страны-лидеры** по числу вакансий: **Германия, Франция, Великобритания, Нидерланды.** При этом в одних странах выше спрос на аналитиков, а в других - на саентистов.

### Тип занятости

In [107]:

```
workplace = df_total.pivot_table(index='workplace_type', columns='spec', values='link', aggfunc='count')
workplace[['DA', 'DS']].plot.pie(subplots=True, figsize=(14, 6), colormap='tab20c')
plt.title("Тип занятости")
plt.show()
```




In [108]:

```
workplace
```

Out[108]:

| spec | DA | DS |
|---|---|---|
| **workplace_type** | | |
| Hybrid | 37 | 35 |
| Not specified | 69 | 75 |
| On-site | 148 | 197 |
| Remote | 124 | 57 |

Похоже, **"удаленка"** вновь **уступила место работе в офисе.** Если аналитики еще могут с почти равной вероятностью устроиться на remote формат, то для саентистов таких возможностей все меньше. Кроме того, в большом количестве вакансий место работы вообще не указывается работодателями.

### Самые нанимающие компании

In [109]:

```
top_companies_DA = da_fin.groupby(by='company_name').agg({'link':'count'}
                                                         ).sort_values(by='link', ascending=False
                                                                       ).head(10).rename(columns={'link':'qty_va
top_companies_DA
```

Out[109]:

| company_name | qty_vac_DA |
|---|---|
| TELUS International AI Data Solutions | 65 |
| TELUS International | 38 |
| Appen | 27 |
| Positive Thinking Company | 6 |
| Walters People | 5 |
| EY | 5 |
| TUI | 4 |
| LSEG (London Stock Exchange Group) | 4 |
| BNP Paribas | 4 |
| Liebherr Group | 4 |

In [110]:

```
top_companies_DS = ds_fin.groupby(by='company_name').agg({'link':'count'}
                                                         ).sort_values(by='link', ascending=False
                                                                       ).head(10).rename(columns={'link':'qty_va
top_companies_DS
```

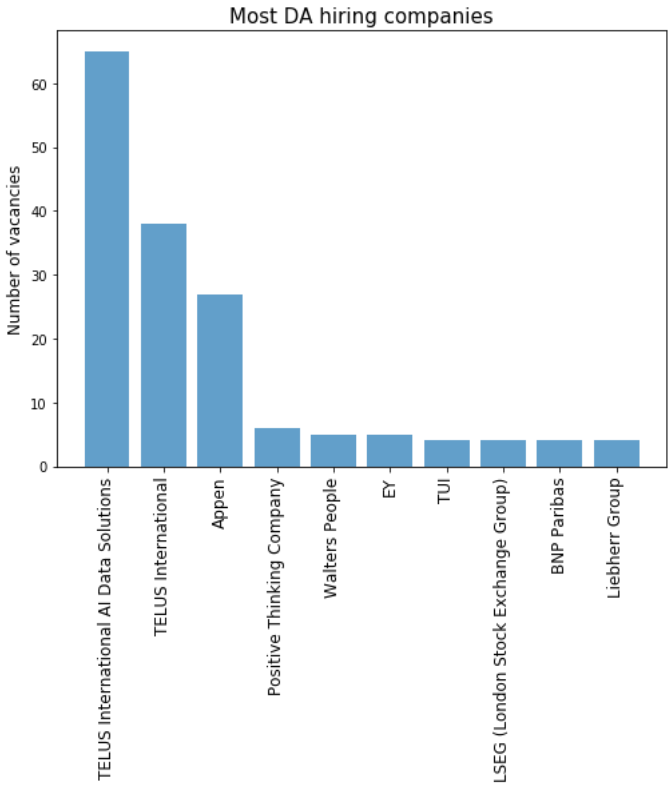
Out[110]:

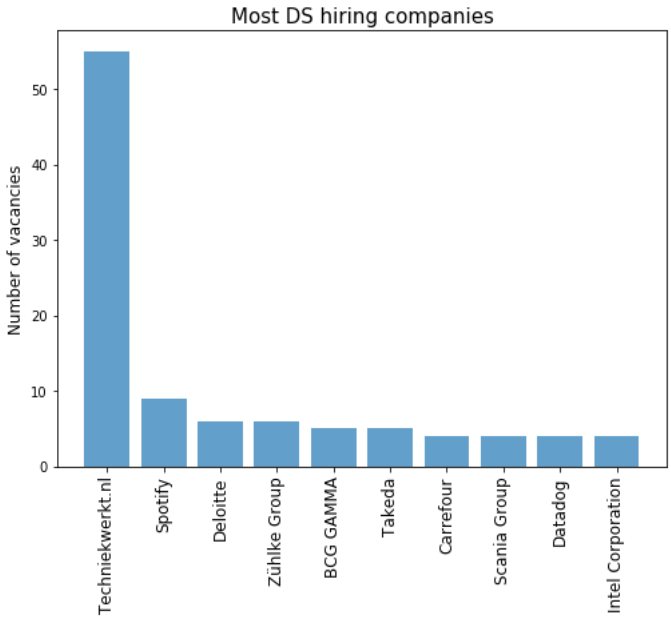
| company_name | qty_vac_DS |
|---|---|
| Techniekwerkt.nl | 55 |
| Spotify | 9 |
| Deloitte | 6 |
| Zühlke Group | 6 |
| BCG GAMMA | 5 |
| Takeda | 5 |
| Carrefour | 4 |
| Scania Group | 4 |
| Datadog | 4 |
| Intel Corporation | 4 |

In [111]:

```
fig, axes = plt.subplots(1, 2, figsize = (18,6))

axes[0].bar(top_companies_DA.index, top_companies_DA['qty_vac_DA'], alpha=0.7)
axes[1].bar(top_companies_DS.index, top_companies_DS['qty_vac_DS'], alpha=0.7)


axes[0].set_title('Most DA hiring companies', fontsize = 15)
axes[0].set_ylabel('Number of vacancies', fontsize = 12)
axes[0].tick_params(
    axis = 'x',
    labelrotation = 90,
    labelsize = 12)

axes[1].set_title('Most DS hiring companies', fontsize = 15)
axes[1].set_ylabel('Number of vacancies', fontsize = 12)
```

```
axes[1].tick_params(
    axis = 'x',
    labelrotation = 90,
    labelsize = 12)

plt.show()
```




**Самые нанимающие отрасли**

In [112]:

```
top_sphere_DA = da_fin.groupby(by='company_sphere').agg({'link':'count'}
                                                        ).sort_values(by='link', ascending=False
                                                                      ).head(10).rename(columns={'link':'qty_va

top_sphere_DS = ds_fin.groupby(by='company_sphere').agg({'link':'count'}
                                                        ).sort_values(by='link', ascending=False
                                                                      ).head(10).rename(columns={'link':'qty_va
```

In [113]:

```
ccolors = plt.cm.BuPu(np.full(len(top_sphere_DA.reset_index().columns), 0.2))

fig, axes = plt.subplots(1, 2)

tab_1 = axes[0].table(cellText=top_sphere_DA.reset_index().values, loc='left',
                      colLabels=top_sphere_DA.reset_index().columns,
                      cellLoc='center',
                      colColours=ccolors)

tab_1.set_fontsize(24)
tab_1.scale(4,5)
axes[0].axis('off')

tab_2 = axes[1].table(cellText=top_sphere_DS.reset_index().values, loc='right',
                      colLabels=top_sphere_DS.reset_index().columns,
                      cellLoc='center',
                      colColours=ccolors)

tab_2.set_fontsize(24)
tab_2.scale(4,5)
axes[1].axis('off')

plt.show()
```

| company_sphere | qty_vac_DA |
|---|---|
| IT Services and IT Consulting | 164 |
| Staffing and Recruiting | 41 |
| Not specified | 22 |
| Financial Services | 20 |
| Technology, Information and Internet | 13 |
| Banking | 11 |
| Software Development | 8 |
| Business Consulting and Services | 8 |
| Accounting | 7 |
| Retail | 7 |

| company_sphere | qty_vac_DS |
|---|---|
| Technology, Information and Internet | 60 |
| IT Services and IT Consulting | 44 |
| Business Consulting and Services | 30 |
| Staffing and Recruiting | 26 |
| Software Development | 21 |
| Not specified | 18 |
| Retail | 13 |
| Musicians | 9 |
| Motor Vehicle Manufacturing | 8 |
| Financial Services | 7 |

Среди топ-10 нанимающих компаний нет пересечений в секторах DA/DS. Да и сферы-лидеры совпадают не вполне. Саентистов охотно берут в консалтинг (включая IT), а аналитиков - в ИТ и рекрутинговые компании.

In [114]:

```
top_da_skills = da_skills_expl.groupby(by='skills').agg({'link':'count'}
                                                        ).sort_values(by='link', ascending=False
                                                                      ).head(10).rename(columns={'link':'qty_va
top_da_skills
```

Out[114]:

| skills | qty_vac_DA |
|---|---|
| stat | 244 |
| git | 181 |
| excel | 161 |
| sql | 125 |
| machine learning | 115 |
| python | 103 |
| dashboard | 91 |
| tableau | 89 |
| power bi | 77 |
| graph | 76 |

Занятно, что среди требований к аналитикам старый добрый Excel преобладает и над Python, и над SQL. Машинное обучение интересует работодателей также сильно и чаще, чем инструментарий Python

In [115]:

```
top_ds_skills = ds_skills_expl.groupby(by='skills'
                                       ).agg({'link':'count'}).sort_values(by='link', ascending=False
                                                                           ).head(10).rename(columns={'link':'q
top_ds_skills
```

Out[115]:

| skills | qty_vac_DS |
|---|---|
| stat | 528 |
| python | 282 |
| machine learning | 244 |
| sql | 201 |
| predict | 114 |
| statistical | 114 |
| git | 113 |
| airflow | 110 |
| tensor | 108 |
| tensorflow | 107 |

А для саентистов, по мнению работодателей, важнее всего статистика. Python и SQL - в пятёрке самых востребованных навыков.

Данные обработаны, первичные выводы сделаны.

Наконец, запишем данные в файлы - для визуализации в Tableau.

In [116]:

```
da_fin.to_csv('da_fin.csv', index=False)
```

```
ds_fin.to_csv('ds_fin.csv', index=False)
```

```
df_total.to_csv('df_total.csv', index=False)
```

```
da_skills_expl.to_csv('da_skills.csv', index=False)
```

```
ds_skills_expl.to_csv('ds_skills.csv', index=False)
```

```
skills_total.to_csv('skills total.csv', index=False)
```

# Построение дашборда в Tableau

Дашборд доступен по ссылке