目　录

XXXXX 数据分析报告

一、数据背景说明

首先说明拟分析的内容：如：为了解近 20 年来中国就业人口的分布情况和变化规律，作者通过国家统计局网站（国家数据）下载了相关数据，经简单处理（调整为便于 python 处理的格式），分别命名为《jiuyerenkou_chanye》《jiuyerenkou_chengxiang》《jiuyerenkou_chengzhen》《jiuyerenkou_xiangcun》。

如下内容将基于这四个表格展开操作和分析。

二、数据清洗

主要任务和内容

在数据分析中，清洗数据是一个重要的步骤。这个步骤的主要任务是导入数据，并对各数据表中的重复值和缺失值进行处理。首先，我们会读取多个 CSV 文件，随后对缺失值进行填充和处理，确保数据的完整性和一致性。

```
# 读取数据
chengzhen_data = pd.read_csv( filepath_or_buffer: 'jiuyerenkou_chengzhen.csv', sep=',', encoding="gbk")
xiangcun_data = pd.read_csv( filepath_or_buffer: 'jiuyerenkou_xiangcun.csv', sep=',', encoding="gbk")
jiuyerenkou_chanye_data = pd.read_csv( filepath_or_buffer: 'jiuyerenkou_chanye.csv', sep=',', encoding="utf-8")
jiuyerenkou_chengxiang_data = pd.read_csv( filepath_or_buffer: 'jiuyerenkou_chengxiang.csv', sep=',', encoding="utf-8")
```

（二）清洗数据

1. 导入数据结果

```
# 合并数据
merged_data = pd.concat( objs: [chengzhen_data, xiangcun_data, jiuyerenkou_chanye_data, jiuyerenkou_chengxiang_data], ignore_index=True)
merged_data = merged_data.fillna(0)  # 将空值设置为0
```

2. 清洗数据结果

```
C:\ProgramData\anaconda3\envs\base2\python.exe "F:\danzi\新建文件夹 (4)\10\1.py"
                    2023年     2022年     2021年  ...   2006年   2005年   2004年
指标                                              ...
个体乡村就业人员(万人)            0.0       0.0       0.0  ...    2147    2123    2066
个体城镇就业人员(万人)            0.0       0.0       0.0  ...    3012    2778    2521
乡村就业人员(万人)           54018.0   54840.0   55758.0  ...   90696   92516   93942
国有单位城镇就业人员(万人)          0.0    5612.0    5633.0  ...    6430    6488    6710
城镇就业人员(万人)           94064.0   91862.0   93546.0  ...   59260   56778   54586
城镇集体单位城镇就业人员(万人)        0.0     235.0     262.0  ...     764     810     897
外商投资单位城镇就业人员(万人)        0.0    1164.0    1220.0  ...     796     688     563
就业人员(万人)           148082.0  146702.0  149304.0  ...  149956  149294  148528
有限责任公司城镇就业人员(万人)        0.0    6506.0    6526.0  ...    1920    1750    1436
港澳台商投资单位城镇就业人员(万人)      0.0    1114.0    1175.0  ...     611     557     470
私营企业乡村就业人员(万人)          0.0       0.0       0.0  ...    2632    2366    2024
私营企业城镇就业人员(万人)          0.0       0.0       0.0  ...    3954    3458    2994
第一产业就业人员(万人)         16882.0   17663.0   17072.0  ...   31941   33442   34830
第三产业就业人员(万人)         35639.0   34583.0   35868.0  ...   24143   23439   22725
第二产业就业人员(万人)         21520.0   21105.0   21712.0  ...   18894   17766   16709
联营单位城镇就业人员(万人)          0.0      19.0      22.0  ...      45      45      44
股份合作单位城镇就业人员(万人)        0.0      58.0      62.0  ...     178     188     192
股份有限公司城镇就业人员(万人)        0.0    1684.0    1789.0  ...     741     699     625
```

三、数据处理

在这一部分中，我们将对数据进行排序、合并和分组。

```python
# 合并数据
merged_data = pd.concat( objs: [chengzhen_data, xiangcun_data, jiuyerenkou_chanye_data, jiuyerenkou_chengxiang_data], ignore
merged_data = merged_data.fillna(0)  # 将空值设置为0

# 排序
merged_data = merged_data.sort_values(by='指标')  # 根据指标排序

# 分组
grouped_data = merged_data.groupby('指标').sum()  # 按指标分组求和

# 查看结果
print(grouped_data)
```

结果：

```
C:\ProgramData\anaconda3\envs\base2\python.exe "F:\danzi\新建文件夹 (4)\10\1.py"
                        2023年     2022年     2021年  ...   2006年   2005年   2004年
指标                                                  ...
个体乡村就业人员(万人)             0.0       0.0       0.0  ...    2147    2123    2066
个体城镇就业人员(万人)             0.0       0.0       0.0  ...    3012    2778    2521
乡村就业人员(万人)           54018.0   54840.0   55758.0  ...   90696   92516   93942
国有单位城镇就业人员(万人)           0.0    5612.0    5633.0  ...    6430    6488    6710
城镇就业人员(万人)           94064.0   91862.0   93546.0  ...   59260   56778   54586
城镇集体单位城镇就业人员(万人)          0.0     235.0     262.0  ...     764     810     897
外商投资单位城镇就业人员(万人)          0.0    1164.0    1220.0  ...     796     688     563
就业人员(万人)           148082.0  146702.0  149304.0  ...  149956  149294  148528
有限责任公司城镇就业人员(万人)          0.0    6506.0    6526.0  ...    1920    1750    1436
港澳台商投资单位城镇就业人员(万人)        0.0    1114.0    1175.0  ...     611     557     470
私营企业乡村就业人员(万人)           0.0       0.0       0.0  ...    2632    2366    2024
私营企业城镇就业人员(万人)           0.0       0.0       0.0  ...    3954    3458    2994
第一产业就业人员(万人)          16882.0   17663.0   17072.0  ...   31941   33442   34830
第三产业就业人员(万人)          35639.0   34583.0   35868.0  ...   24143   23439   22725
第二产业就业人员(万人)          21520.0   21105.0   21712.0  ...   18894   17766   16709
联营单位城镇就业人员(万人)           0.0      19.0      22.0  ...      45      45      44
股份合作单位城镇就业人员(万人)          0.0      58.0      62.0  ...     178     188     192
股份有限公司城镇就业人员(万人)          0.0    1684.0    1789.0  ...     741     699     625

[18 rows x 20 columns]
<class 'pandas.core.frame.DataFrame'>
Index: 21 entries, 13 to 6
Data columns (total 21 columns):
```

四、数据分析

进行基本统计、分组分析和结构分析：

```python
# 基本统计
statistics = merged_data.describe()

# 分组分析
grouped_stats = merged_data.groupby('指标').agg(['mean', 'sum', 'count'])

# 结构分析
structure = merged_data.info()

# 输出统计结果
print(statistics)
print(grouped_stats)
```

结果：

```
Index: 21 entries, 13 to 6
Data columns (total 21 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   指标      21 non-null     object
 1   2023年   21 non-null     float64
 2   2022年   21 non-null     float64
 3   2021年   21 non-null     float64
 4   2020年   21 non-null     float64
 5   2019年   21 non-null     int64
 6   2018年   21 non-null     int64
 7   2017年   21 non-null     int64
 8   2016年   21 non-null     int64
 9   2015年   21 non-null     int64
 10  2014年   21 non-null     int64
 11  2013年   21 non-null     int64
 12  2012年   21 non-null     int64
 13  2011年   21 non-null     int64
 14  2010年   21 non-null     int64
 15  2009年   21 non-null     int64
 16  2008年   21 non-null     int64
 17  2007年   21 non-null     int64
 18  2006年   21 non-null     int64
 19  2005年   21 non-null     int64
 20  2004年   21 non-null     int64
dtypes: float64(4), int64(16), object(1)
memory usage: 3.6+ KB
              2023年         2022年  ...         2005年         2004年
count     21.000000     21.000000  ...     21.000000     21.000000
mean   17628.809524  18245.095238  ...  18818.333333  18660.095238
std    24838.585696  24006.156481  ...  24144.760519  24188.586405
min        0.000000      0.000000  ...     45.000000     44.000000
25%        0.000000     58.000000  ...    810.000000    897.000000
50%        0.000000   5612.000000  ...   3458.000000   2994.000000
75%    27009.000000  27420.000000  ...  28389.000000  27293.000000
max    74041.000000  73351.000000  ...  74647.000000  74264.000000
```

```
[8 rows x 20 columns]
                    2023年                  ...    2004年
                     mean       sum count  ...     mean     sum count
指标                                        ...
个体乡村就业人员(万人)            0.0       0.0     1  ...   2066.0    2066     1
个体城镇就业人员(万人)            0.0       0.0     1  ...   2521.0    2521     1
乡村就业人员(万人)           27009.0   54018.0     2  ...  46971.0   93942     2
国有单位城镇就业人员(万人)          0.0       0.0     1  ...   6710.0    6710     1
城镇就业人员(万人)           47032.0   94064.0     2  ...  27293.0   54586     2
城镇集体单位城镇就业人员(万人)        0.0       0.0     1  ...    897.0     897     1
外商投资单位城镇就业人员(万人)        0.0       0.0     1  ...    563.0     563     1
就业人员(万人)             74041.0  148082.0     2  ...  74264.0  148528     2
有限责任公司城镇就业人员(万人)        0.0       0.0     1  ...   1436.0    1436     1
港澳台商投资单位城镇就业人员(万人)      0.0       0.0     1  ...    470.0     470     1
私营企业乡村就业人员(万人)          0.0       0.0     1  ...   2024.0    2024     1
私营企业城镇就业人员(万人)          0.0       0.0     1  ...   2994.0    2994     1
第一产业就业人员(万人)         16882.0   16882.0     1  ...  34830.0   34830     1
第三产业就业人员(万人)         35639.0   35639.0     1  ...  22725.0   22725     1
第二产业就业人员(万人)         21520.0   21520.0     1  ...  16709.0   16709     1
联营单位城镇就业人员(万人)          0.0       0.0     1  ...     44.0      44     1
股份合作单位城镇就业人员(万人)        0.0       0.0     1  ...    192.0     192     1
股份有限公司城镇就业人员(万人)        0.0       0.0     1  ...    625.0     625     1

[18 rows x 60 columns]
None
```

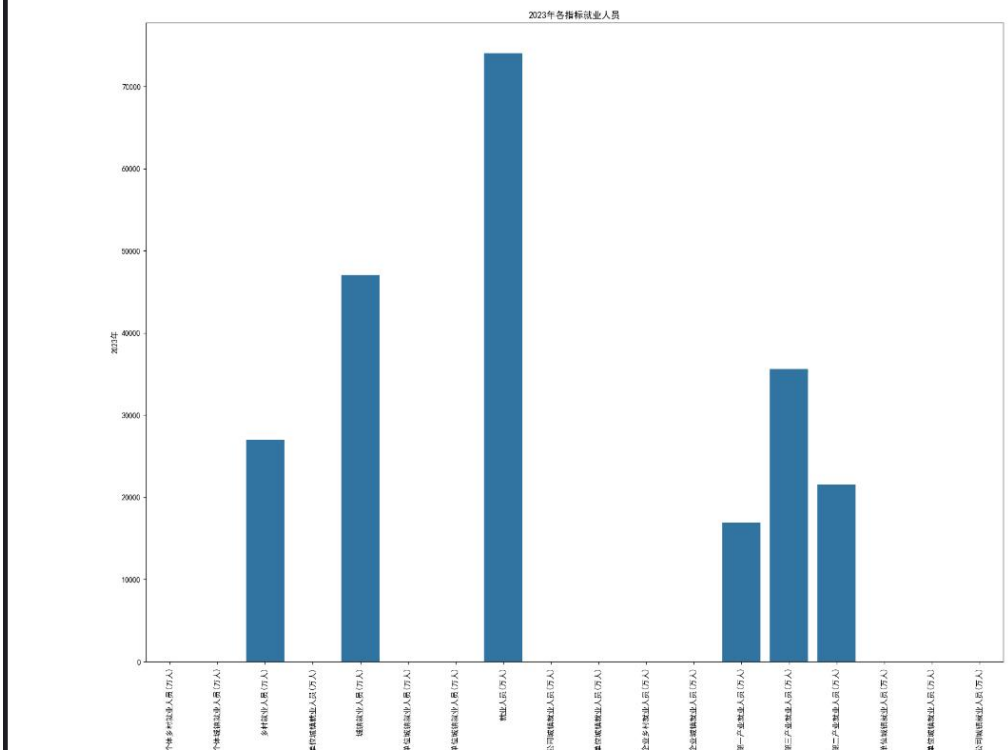
五、数据可视化

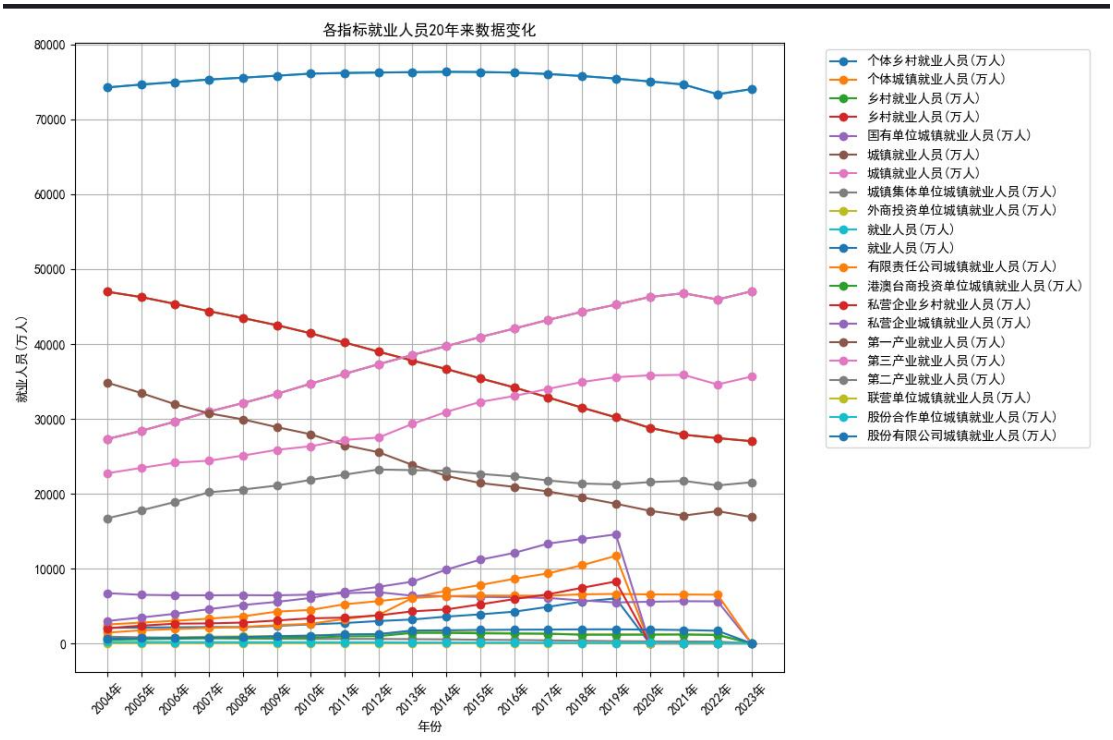
在这一部分中，我们将使用 Matplotlib 和 Seaborn 创建静态图，以及使用 Plotly 创建动态图：

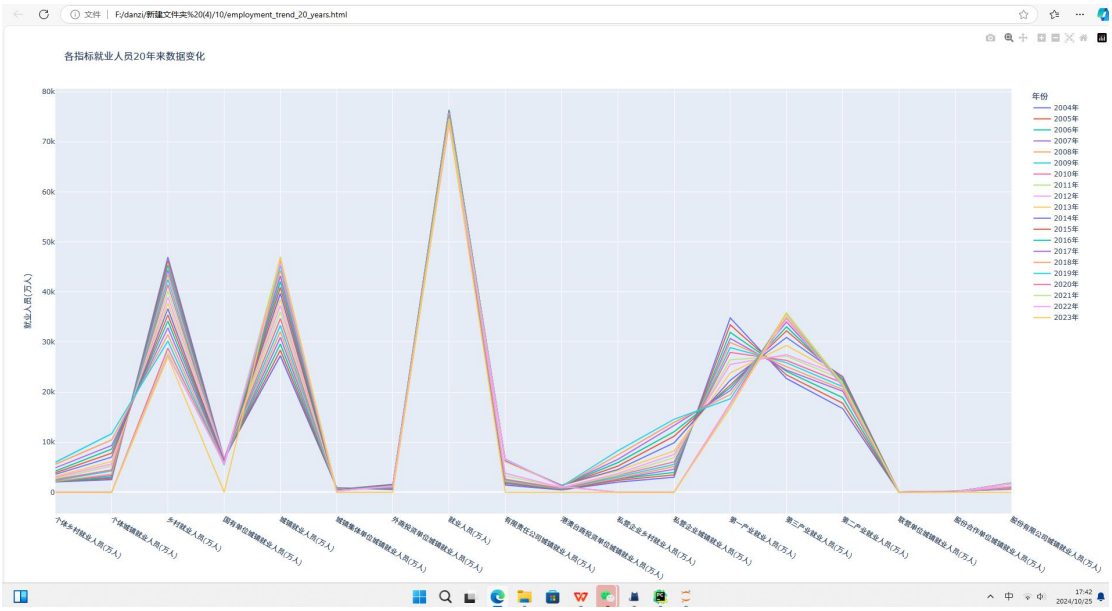
```
import matplotlib.pyplot as plt
import seaborn as sns
plt.rcParams['font.sans-serif']=['SimHei'] #用来正常显示中文标签
plt.rcParams['axes.unicode_minus']=False #用来正常显示负号 #有中文出现的情
# 静态图示例
plt.figure(figsize=(20, 15))
sns.barplot(x='指标', y='2023年', data=merged_data)
plt.title('2023年各指标就业人员')
plt.xticks(rotation=90)
plt.savefig('2023_year_employment.png')  # 保存图形
plt.show()


years = [str(year) + '年' for year in range(2004, 2024)]  # 2004年至2023年

# 绘制折线图
plt.figure(figsize=(12, 8))

for index, row in merged_data.iterrows():
    plt.plot( *args: years, row[years], marker='o', label=row['指标'])

plt.title('各指标就业人员20年来数据变化')
plt.xlabel('年份')
plt.ylabel('就业人员(万人)')
plt.xticks(rotation=45)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')  # 将图例放在外面
plt.grid()

# 保存图形
plt.tight_layout()  # 自动调整子图参数
plt.savefig('employment_change_over_20_years.png')  # 保存图形
plt.show()
```

```python

# 动态图示例（使用Plotly）
import plotly.express as px

years = [str(year) + '年' for year in range(2004, 2024)]  # 2004年至2023年
merged_data_years = merged_data.set_index('指标')[years].reset_index()

# 动态图示例
fig = px.line(merged_data_years, x='指标', y=years,
              title='各指标就业人员20年来数据变化',
              labels={'value': '就业人员(万人)', 'variable': '年份'})

# 保存动态图
fig.write_html('employment_trend_20_years.html')  # 保存动态图

```

结果：







六、导出文件

最后，我们将分析后的数据表和图形导出保存。

```python
# 导出合并后的数据
merged_data.to_csv('merged_employment_data.csv', index=False)

# 导出统计结果
statistics.to_csv('employment_statistics.csv')

# 导出分组分析结果
grouped_stats.to_csv('grouped_employment_analysis.csv')

# 导出结构分析结果（可以用文本文件保存）
with open('data_structure_info.txt', 'w') as f:
    f.write(str(structure))
```

结果：

七、总结与分析

通过本次分析，我们获得了对就业人员变化的全面理解。数据的清洗、处理和可视化方法使我们能够有效地呈现出就业趋势，为相关政策制定和社会经济研究提供了有力支持。

可以看出，无论是总体就业人数还是各方面的就业人员，近几年来均有略微的下降趋势。这一现象可能受到经济增长放缓、产业结构转型、政策调整以及技术进步等多重因素的影响。