pandas && □□□□□□□□□

□□

```
pip install pandas -i https://pypi.tuna.tsinghua.edu.cn/simple

# excel□□
pip install openpyxl -i https://pypi.tuna.tsinghua.edu.cn/simple
pip install xlsxwriter  -i https://pypi.tuna.tsinghua.edu.cn/simple
```

□□□□

□□□□CSV,JSON,Excel,mysql□□

□□□□https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html

api□□□□□□□□

- □□□read_csv , read_json , read_excel,read_sql
- □□□to_csv , to_json□to_excel , to_sql

□□□mysql

```
pip install pandas -i https://pypi.tuna.tsinghua.edu.cn/simple
pip install sqlalchemy -i https://pypi.tuna.tsinghua.edu.cn/simple
```

```
# □□pandas□□□□
import pandas as pd

# □□SQLAlchemy□□□
from sqlalchemy import create_engine
```

```
# □□□□
if True:
    engine = create_engine(
        "mysql+pymysql://root:password@192.168.0.10:33318/renren_cloud_basic"
    )
    query = "SELECT * FROM park_user"
    # □□chunksize□□□□□□□□□
    df = pd.read_sql(query, engine)
    # □□□□□□□□□□□□□
    print(df.info())
    # □□□□
    print(df.columns)
```

□□□mysql□□□□□

- read_sql_table(table_name,connection)
- read_sql_query(sql,connection)
- read_sql(sql,connection)

□□□□

- □□□□□□

```
row_size, col_size = grouped.shape
log.info(f"row_size:{row_size},col_size:{col_size}")

# row_size:13,col_size:7
```

- □□□□□□□□□□□□□□□

```
# □□□□□□□□
dataFrame["□□"] = pd.to_datetime(dataFrame["□□"]).dt.year
```

- □□□□□□□□□
- □□□□□□□□□□□□□□□
- □□□□□□□□□□□□

- 数据清洗：清洗重复数据
- 分组聚合

```
grouped = (
    dataFrame.groupby(["客户名称", "合同编号", "年份"])
    .agg(
        {
            "减免总金额": "sum",
            "政策减免金额": "sum",
            "运营减免金额": "sum",
            "其他减免金额": "sum",
        }
    )
    .reset_index()  # 使分组变成一个平面
)

# 设置索引列的名称为“序号”
grouped.index.name = '序号'
```

- 筛选数据

```
# 提取整列数据
print(grouped["客户名称"])



# 筛选客户名称列数据
print(grouped[grouped["客户名称"] == "山东智拓大数据有限公司"])
```

写入Eexcel时设置样式

pandas是做数据处理的，默认导出并不能设置样式、合并单元格等，需要记住其他工具。

```
# 自定义写入程序，使用xlsxwriter写入
with pd.ExcelWriter("./files/output.xlsx", engine="xlsxwriter") as writer:


    # 直接写入数据，没有样式
```

```
# grouped.to_excel(writer, sheet_name="Sheet1", startrow=1, startcol=1, index=True)

# □□□□□
workbook = writer.book
worksheet = None
# □□worksheet□□□□□□□□□□□□
if "Sheet1" in writer.sheets:
    worksheet = writer.sheets["Sheet1"]
else:
    worksheet = workbook.add_worksheet("Sheet1")

# □□□□
default_format = {
    "bold": False,
    # □□
    "border": 1,
    # □□□□
    "valign": "vcenter",
    # □□□□
    "text_wrap": True,
    # □□□□
    # "align": "center",
    # □□□□
    "font_size": 9,
}

# □□□□
cell_format_default = workbook.add_format(default_format)
cell_format_head = workbook.add_format(
    {
        **default_format,
        **{
            "align": "center",
            "bold": True,
        },
    }
)

# □□□□□□□□□□□□□□□□□□□□□□□□□□□□
worksheet.set_column("A:A", 10)
worksheet.set_column("B:B", 30)
```

```
worksheet.set_column("C:C", 20)
worksheet.set_column("E:H", 12)

# □□logo□□
worksheet.insert_image(
    "A1",
    "./files/logo.png",
    {
        "x_scale": first_row_height / logo_image_height,
        "y_scale": first_row_height / logo_image_height,
        "x_offset": 5,
        "y_offset": 5,
    },
)
# □□□□□
worksheet.merge_range("A1:B1", "", cell_format_head)
worksheet.merge_range("C1:H1", "□□□□□", cell_format_head)
# □□□□□□□□□□□
worksheet.set_row(0, first_row_height)

# □□□□□
worksheet.write("A2", "□□", cell_format_head)
worksheet.write_column("A3", range(1, len(grouped) + 1), cell_format_head)

# □□□□□□□□□
row_offset = 2
col_offset = 1

# □□□□□□□□□□□□xlsxwriter□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
for row_num, row_data in grouped.iterrows():
    for col_num, col_data in enumerate(row_data):
        worksheet.write(
            row_num + row_offset,
            col_num + col_offset,
            col_data,
            cell_format_default,
        )

# □□□□□□□□□□□□□□□□□□□□□□□
start_row = 0
for row_num, row_data in grouped.iterrows():
```

```
        current_value = row_data["□□□□"]
        next_row = row_num + 1
        next_value = (
            grouped.iloc[next_row]["□□□□"] if next_row < len(grouped) else None
        )
        if current_value != next_value:
            log.info(f"{start_row} - {row_num} - {current_value}")
            if row_num - start_row > 0:
                worksheet.merge_range(
                    start_row + row_offset,
                    grouped.columns.get_loc("□□□□") + col_offset,
                    row_num + row_offset,
                    grouped.columns.get_loc("□□□□") + col_offset,
                    current_value,
                    cell_format_default,
                )
            start_row = next_row
```

□□□ #22
□ □□□ □□□ 11 □□ 2025 19:11:35
□ □□□ □□□ 18 □□ 2025 03:45:38