《软件实践》课程实验报告

暑期学校实验项目：高考志愿填报助手

小组名称	kgA 组						
姓　　名	王明灏	专业	人工智能	班级	091181	学号	09118139
实验时间	2020.8.31-2020.9.23		指导教师	孔祥龙		成绩	

一、实验背景和目的

基于每年高考成绩公布后家长和学生对志愿填报的无所适从，加之社会上各种以“志愿填报”为名号的收费机构的鱼龙混杂，在授课教师孔祥龙老师的带领下，人工智能学院大三本科生决定利用暑期学校的时间共同开发一个高考志愿填报助手，旨在帮助高考后迷茫的家长和学生顺利完成志愿填报，减少时间和金钱的代价，更重要的是数据来源公开透明，使用人工智能算法合理推荐分数段内合适的学校，最大程度上减少了退档、滑档、掉档等影响考生命运的意外的发生。本系统的知识图谱部分基于 neo4j 图数据库完成，使生成的图谱清晰易懂，操作简单方便，入门门槛极低，也提供了一些基础问答功能。

二、小组任务和个人任务

小组任务：

任务 1：数据源

本项目需要用到的数据源：是第一组清洗的包含学校，专业，省份，分数，年份的 csv 文件。

任务：从之前的小组获取报考省份及相应分数线的讯息

任务 2：知识图谱设计与优化

利用已有的数据构建一个小型的报考知识图谱(知识库)，通过调用该图　谱可以实现如下功能：

1.已知自己某分数能上什么学校

2.某个特定的专业哪个学校分数最高

3.已知自己的分数判断自己能学什么样的专业

4.查询某学校的特定专业

5.我只想学 XX 专业，能去什么学校？

任务 3：知识图谱数据准备

此任务包括：

1.对专业名称进行消歧处理

2.为有需要的实体生成标识符

任务 4：创建可以导入Neo4j 的 csv 文件

在第一个任务里，我们已经分获取了了包含所有信息的 csv，但这些文件不能直接导入到 Neo4j 数据库。所以需要做一些处理，并生成能够直接导入 Neo4j 的 csv 格式。 我们需要生成这几个文件：暨设计环节决定的实体及这些实体之间相互关系对应的 csv 文件。

任务 5：利用上面的 csv 文件生成数据库

使用 neo4j 命令把所有的数据导入到 Neo4j 中，数据默认存在 graph.db 文件夹中。

重启 Neo4j 服务，通过 localhost:7474 观察知识图谱。

使用自带的命令进行简单查询的测试，如：

```
# 查询 node

MATCH (n:Concept) RETURN n LIMIT 25# 查询 relationship

MATCH p=()-[r:industry_of]->() RETURN p LIMIT 100
```

任务 6：基于构建好的知识图谱，构建显示网页

个人任务：

任务 4：创建可以导入Neo4j 的 csv 文件

在第一个任务里，我们已经分获取了了包含所有信息的 csv，但这些文件不能直接导入到 Neo4j 数据库。所以需要做一些处理，并生成能够直接导入 Neo4j 的 csv 格式。 我们需要生成这几个文件：暨设计环节决定的实体及这些实体之间相互关系对应的 csv 文件。

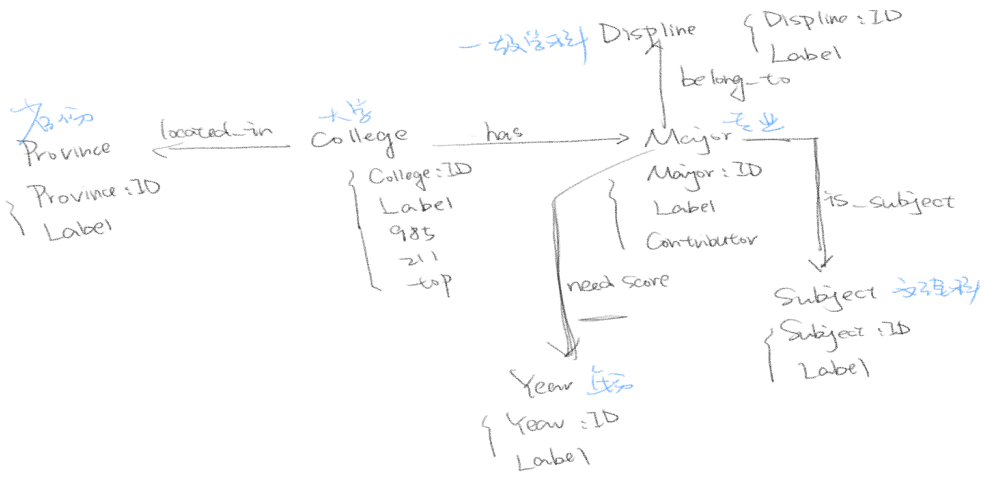
三、个人任务需求分析

1.沟通组员，确定最后生成知识图谱的模式。

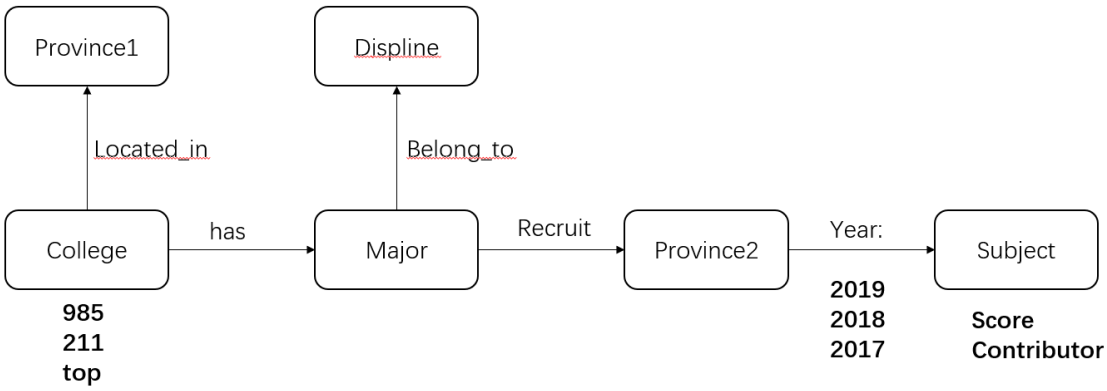
2.对于消歧后的文件进行进一步处理，以便处理后的 csv 文件格式可以导入至 neo4j 中。

四、实验过程（需附上关键代码及相关说明）

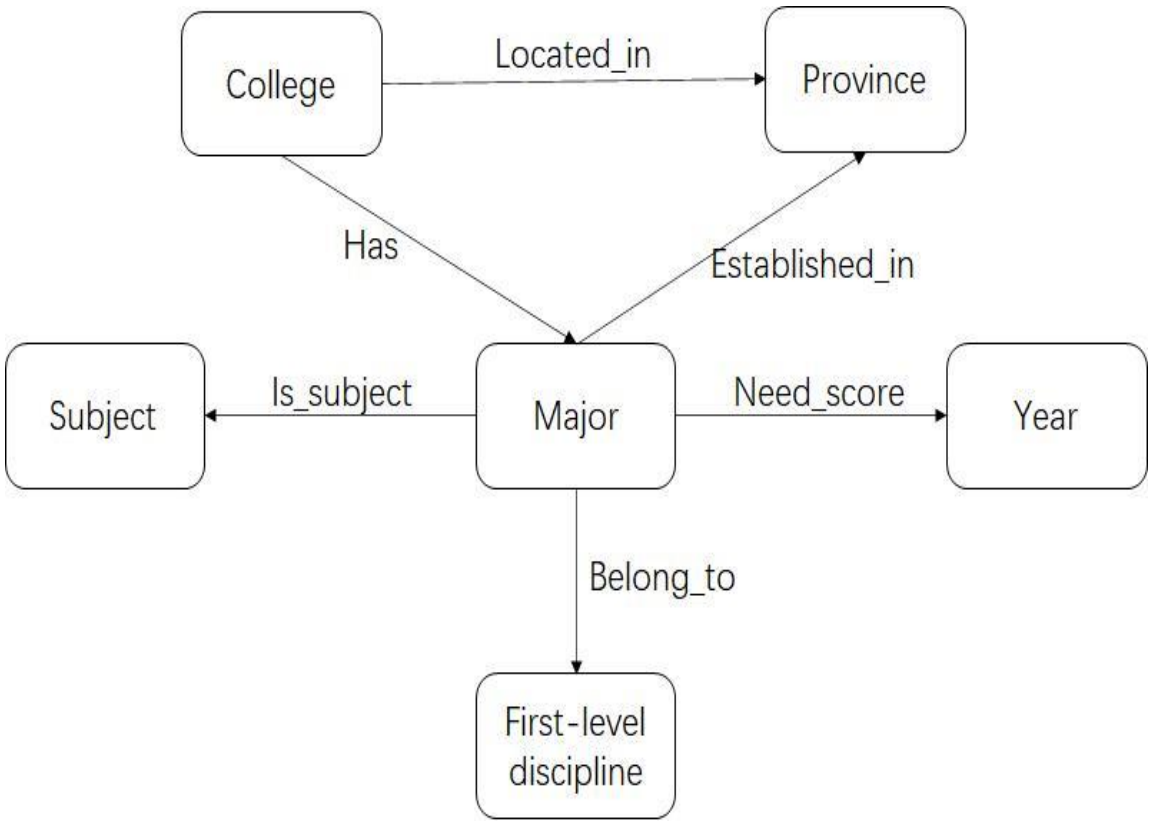
首先确定最后生成知识图谱的模式。这部分由我和罗琦晴、王靖婷三人共同完成。我们经过多次讨论和沟通，一共得出了三种模式：



这种模式是在项目进行到中期之后由朱佳涛、白劭宸组产生的模式，看起来很清晰，实现起来也比较容易。但因为项目在逐步推进，这个模式产生的比较晚，不太符合前期工作中对数据的处理结果，因而我、王靖婷、罗琦晴在讨论之后基于现有的结果和已有 csv 文件的 schema 画出了一个同样高效清晰的模式。



与第一幅图不同的是，我们对 schema 进行了一定程度的修改。Province 这一实体出现了两次，这样虽然会造成整体数据的部分冗余，但是在运行时并不会过多的占用内存，反而这样处理之后使最后的图谱结果清晰易读，有较好的实用性；此外，我们将 year 作为一个关系对应 province 和 subject 之间，而不是原本设想的将其作为一个实体。这两种选择各有利弊，我们讨论以后认为可以将其作为关系设计到我们的模式中。除此之外就是每个实体对应的属性，这张图很清晰地展示了实体、属性及其之间的关系。在最后实现的过程中，我们认为如果将 year 作为一个关系可能不符合思考的认知模式，因而又对 schema 进行了最后一次修改，修改后最终的 schema 如下：



我们三人按照这种数据模式对后续的 csv 文件进行处理，使结构清晰，实现起来很容易。

其次是处理消歧后的文件，我和王靖婷、罗琦晴三个人共同承担这部分任务，从同组的其他同学中获得了消歧的文件 disambiguated.csv, preprocess.zip, 但并不能直接导入知识图谱中，还需要进一步处理使之符合格式要求。以下是我处理的五个 csv 文件时的代码：

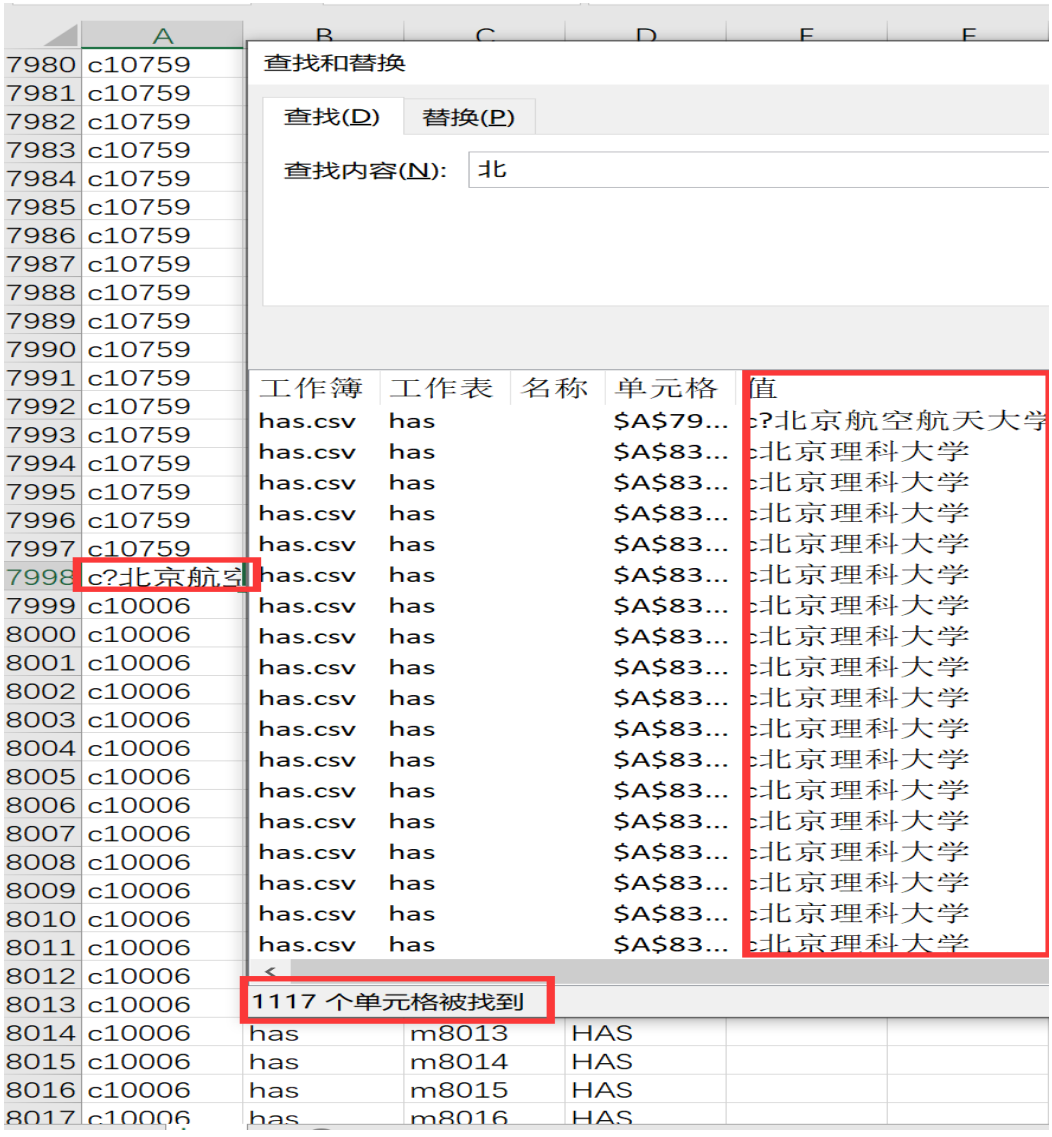
```
import pandas as pd
file_path = 'D:\\seu-2020\\softwarepractise\\preprocess\\entity\\'
file_name= 'college.csv'
file=file_path+file_name
csv_file=pd.read_csv(file, encoding='utf-8')
csv_df=pd.DataFrame(csv_file)
csv_df.rename(columns={'College:ID':'CollegeID:ID','label':'CollegeName','985':'985:int','211':'211:int','top':'Top:int'},inplace=True)
csv_df[':LABEL']='college'
csv_df.to_csv('D:\\seu-2020\\softwarepractise\\result'+file_name, index=False,encoding='utf-8')


file_name='displine.csv'
csv_file=pd.read_csv(file, encoding='utf-8')
```

```
csv_df=pd.DataFrame(csv_file)
csv_df.rename(columns={'Displine:ID':'DisplineID:ID','label':'DisplineName'},inplace=True)
csv_df[':LABEL']='First-level discipline'
csv_df.to_csv('D:\\seu-2020\\softwarepractise\\result'+file_name, index=False,encoding='utf-8’)

file_name='belong_to.csv'
csv_file=pd.read_csv(file, encoding='utf-8')
csv_df=pd.DataFrame(csv_file)
csv_df.drop(['belong_to'],axis=1,inplace=True)
csv_df.to_csv('D:\\seu-2020\\softwarepractise\\result'+file_name, index=False,encoding='utf-8')

file_name='is_subject.csv'
csv_file=pd.read_csv(file, encoding='utf-8')
csv_df=pd.DataFrame(csv_file)
csv_df.drop(['is_subject'],axis=1,inplace=True)
csv_df.rename({'IS_SUBJECT'=':TYPE'},inplace=True)
csv_df.to_csv('D:\\seu-2020\\softwarepractise\\result'+file_name, index=False,encoding='utf-8')

file_name='has.csv'
csv_file=pd.read_csv(file, encoding='utf-8')
csv_df=pd.DataFrame(csv_file)
csv_df.drop(['has'],axis=1,inplace=True)
csv_df.to_csv('D:\\seu-2020\\softwarepractise\\result'+file_name, index=False,encoding='utf-8')
```

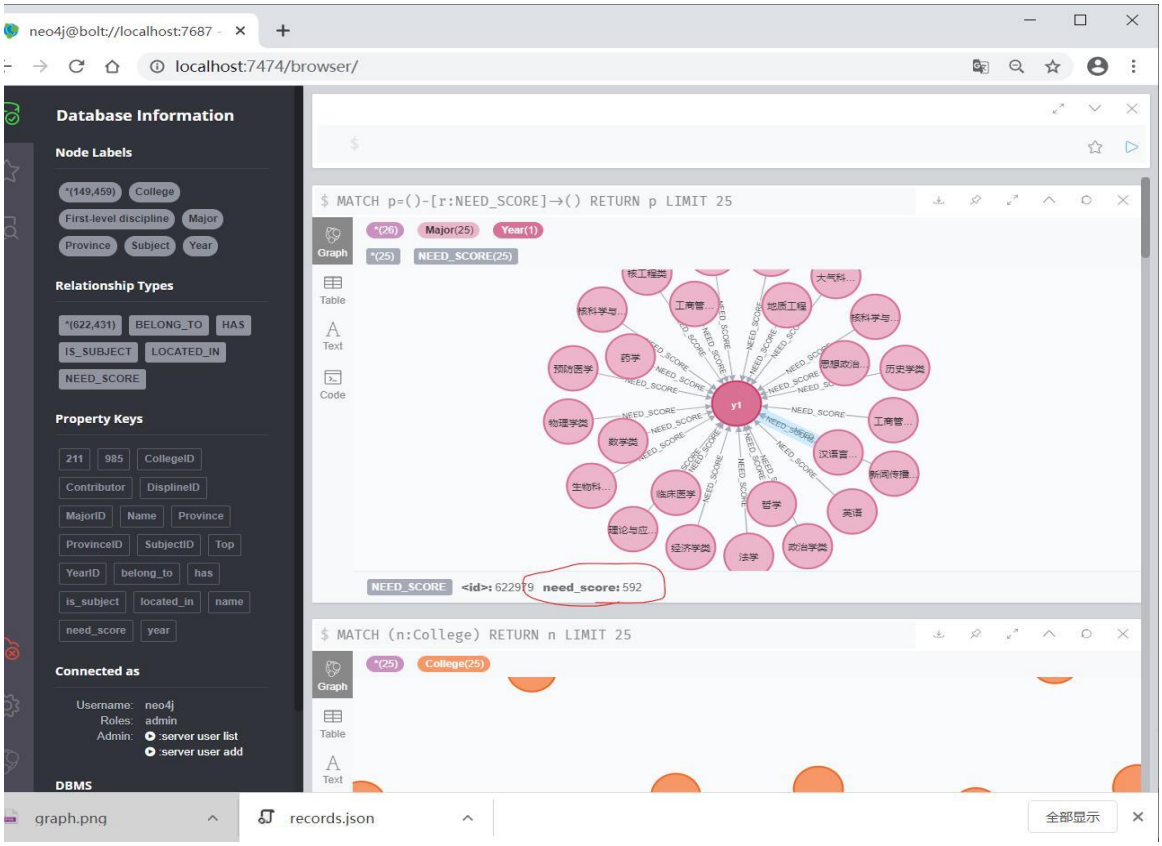
由于数据源出了问题，导致消歧后的文件仍然存在各种各样的问题，这些问题在我们初次尝试导入 neo4j 的时候报错(见下图)。

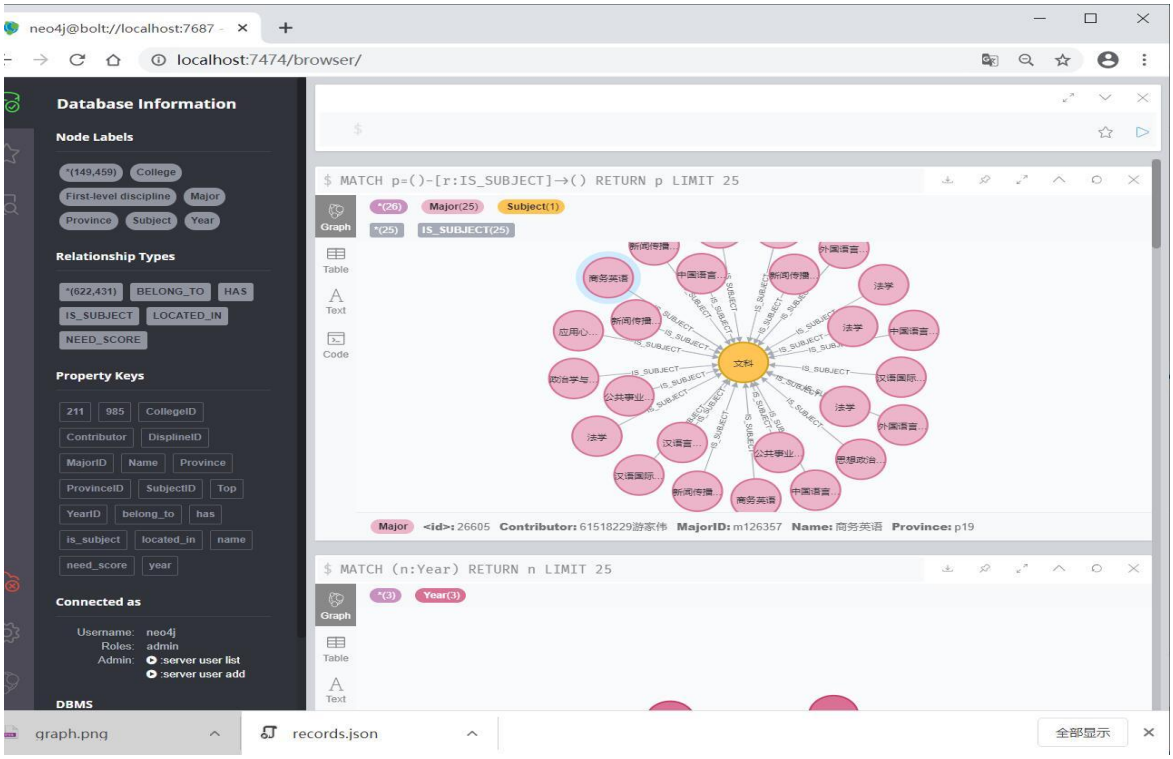
```
命令提示符
..........  ..........  ..........  ..........  ..........   85% ?3ms
..........  ..........  ..........  ..........  ..........   90% ?5ms
..........  ..........  ..........  ..........  ..........   95% ?3ms
..........  ..........  ..........  ..........  ..........  100% ?3ms


IMPORT FAILED in 2s 667ms.
Data statistics is not available.
Peak memory usage: 1.00 GB
Error in input data
Caused by:ERROR in input
  data source: BufferedCharSeeker[source:D:\neo4j-community-3.5.22\bin\has.csv, position:174831, line:7997]
  in field: :TYPE:4
  for header: [:START_ID, has:string, :END_ID, :TYPE]
  raw field value: HAS
  original error: c?北京航空航天大学 (global id space)-[HAS]->m7997 (global id space) referring to missing n
ode c?北京航空航天大学

WARNING Import failed. The store files in D:\neo4j-community-3.5.22\data\databases\graph.db are left as they
 are, although they are likely in an unusable state. Starting a database on these store files will likely fa
il or observe inconsistent records so start at your own risk or delete the store manually
unexpected error: ERROR in input
  data source: BufferedCharSeeker[source:D:\neo4j-community-3.5.22\bin\has.csv, position:174831, line:7997]
  in field: :TYPE:4
  for header: [:START_ID, has:string, :END_ID, :TYPE]
  raw field value: HAS
  original error: c?北京航空航天大学 (global id space)-[HAS]->m7997 (global id space) referring to missing n
ode c?北京航空航天大学

D:\neo4j-community-3.5.22\bin>
```

经过查找之后发现是原始数据出了问题，约有一千组数据没有映射成编号(如北京航空航天大学)。

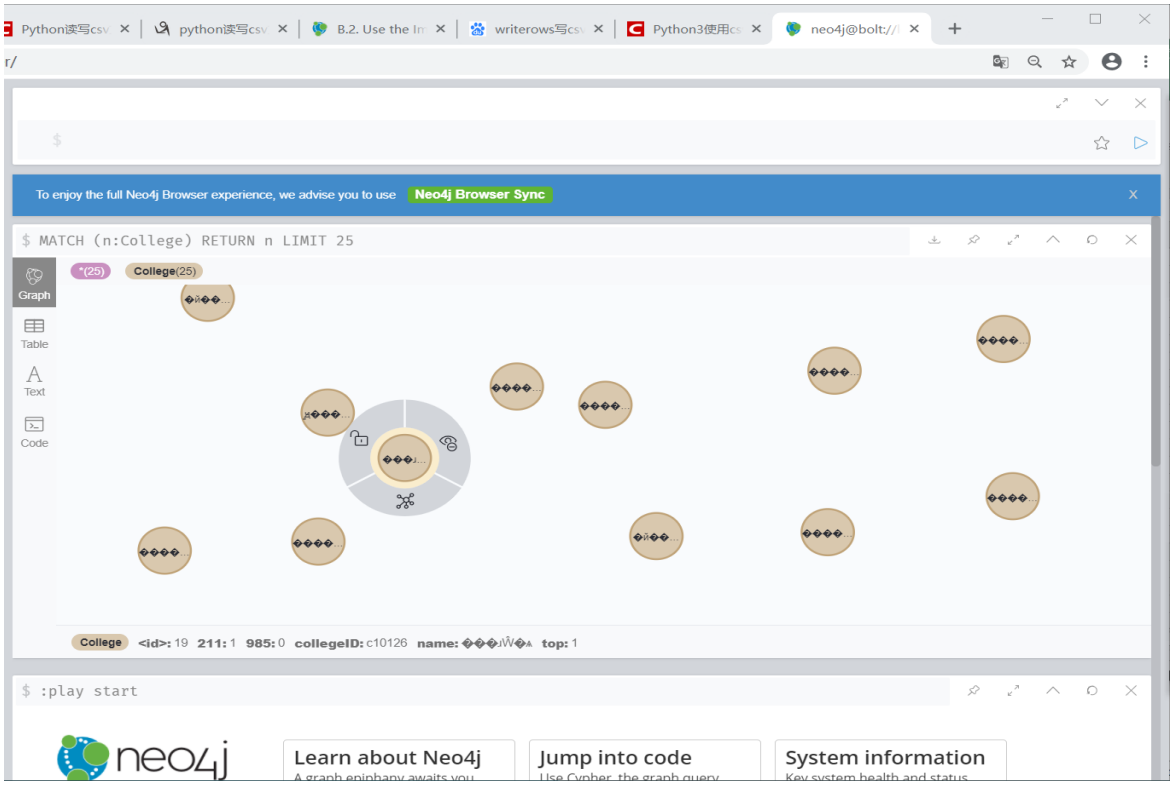


对于这些学校我们采用手工修改的方式将其还原成对应映射的编号，最后将其导入 neo4j 中。以下是一些导入时的进程。





当然，在实验过程中也出现了一些问题，比如在导入知识图谱时出现了文字乱码的问题。



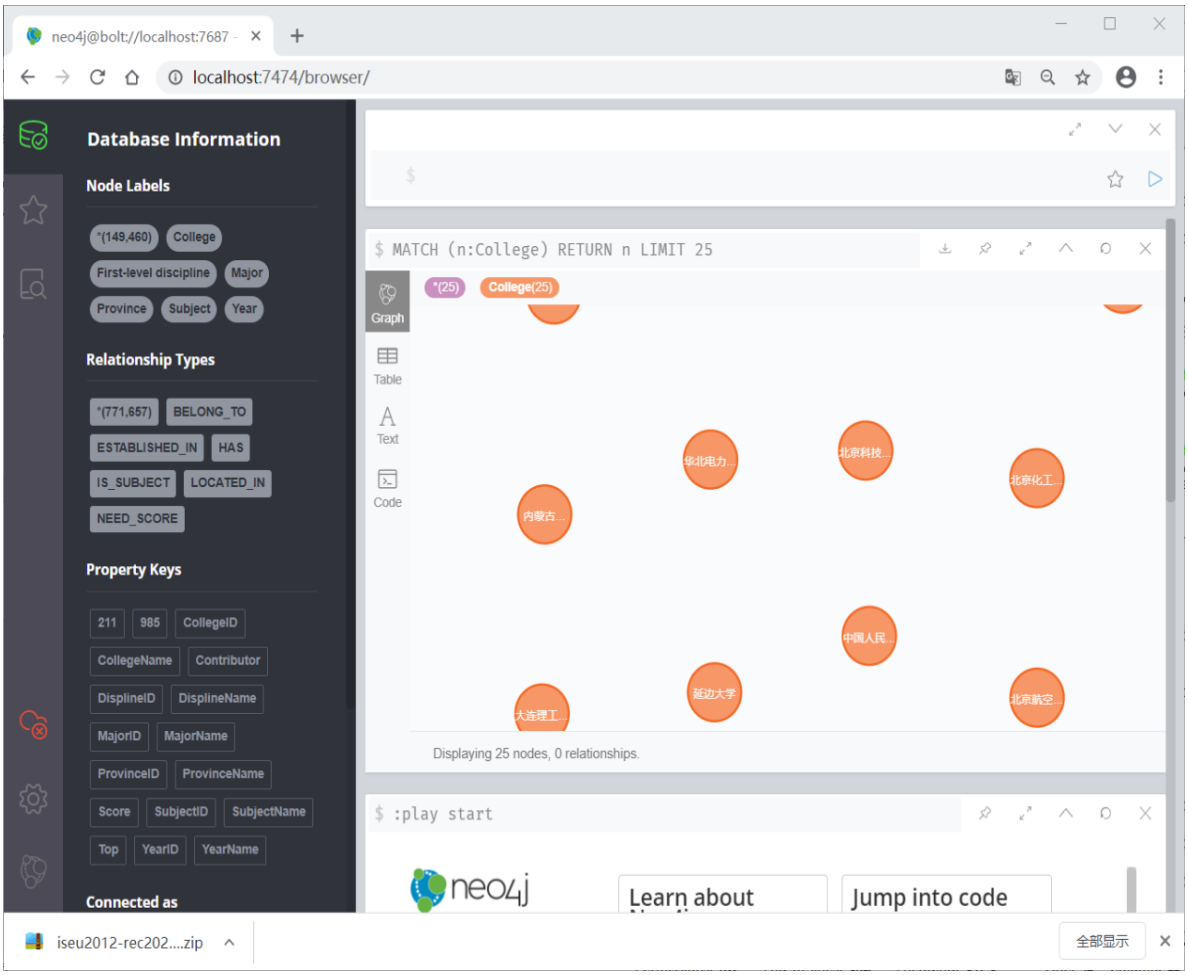
经过检查之后我们发现，是因为 char 类型数据的存储和读取格式问题。因此，对于已处理的 csv 文件，我们做以下处理：

```
import pandas as pd


csv_files = ["college.csv", "major.csv", "displine.csv", "province.csv", "subject.csv", "year.csv"]


for i in csv_files:
    file_path = ' D:\\seu-2020\\softwarepractise\\result ' + i
    csv_file = pd.read_csv(file_path, encoding='utf-8')
    csv_df = pd.DataFrame(csv_file)
    #print(csv_df)
    #for item in csv_df.iloc[1:,1]:
     #    item = "\"" + item + "\""
    csv_df.to_csv(' D:\\seu-2020\\softwarepractise\\result '+i, index=False, quoting = 1)
    #print('-----------------------')
    #print(csv_df)
```

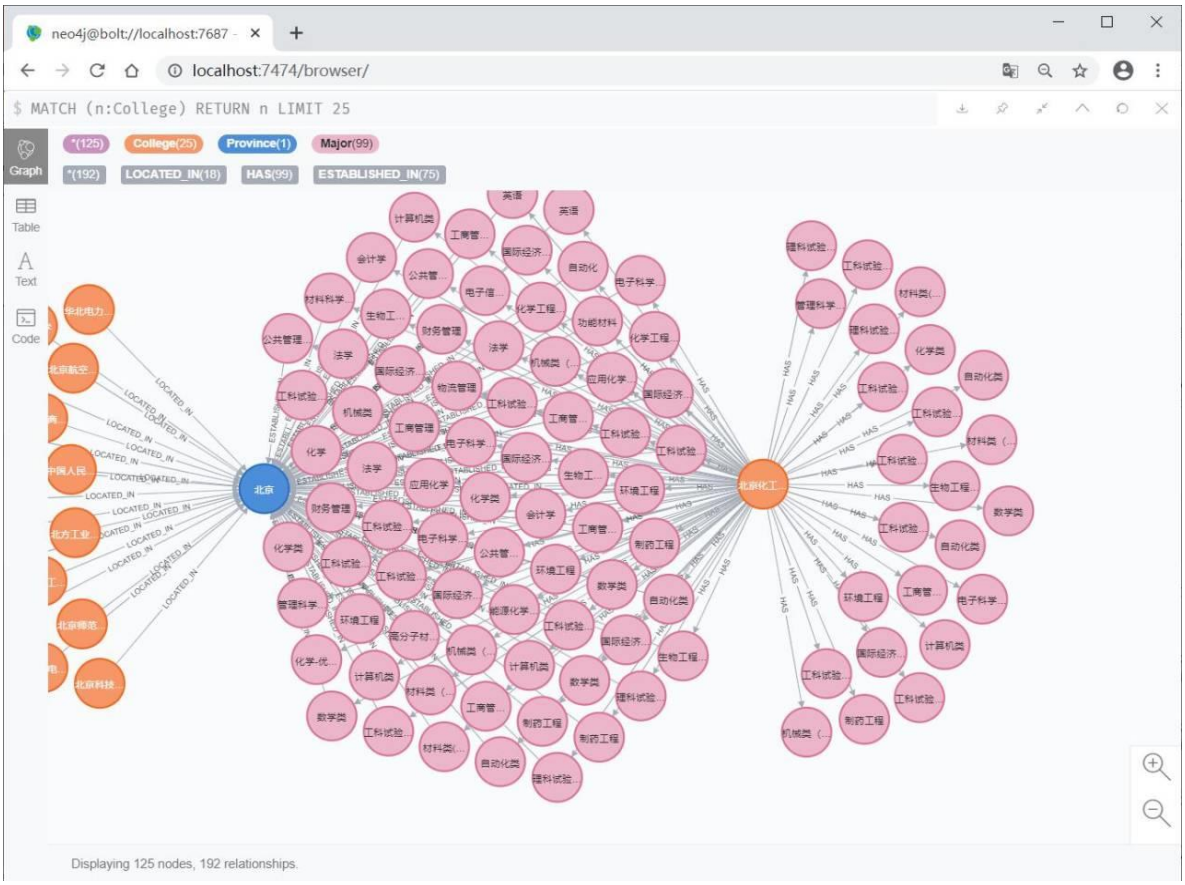
这样使生成的文件格式正确，可以被输入进 neo4j 中并得到正确的结果。

五、实验结果与分析

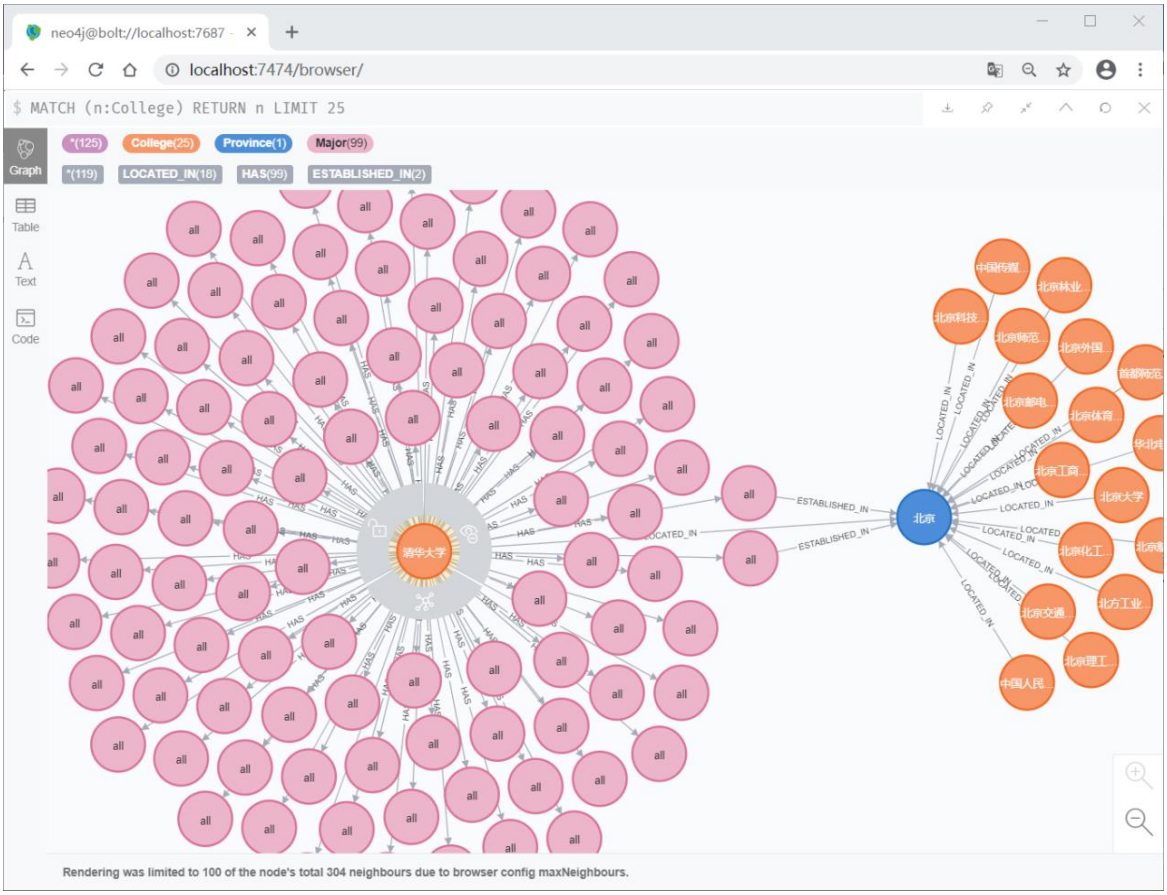
将处理好的 csv 文件导入 neo4j 中，可以看到结果如下：



点击大学之后，结果如下：



对于有些没有具体分数线的专业，我们用 all 来代表这些专业对应的分数



结果分析：我们对获得的数据以及最后生成的知识图谱进行了一些分析，仍然存在一些问题。

1. 有的实体因为数据清洗疏忽的原因，名称与原本名称有出入，导致生成关系时未匹配到对应的标识符。
2. 处理一对多的关系时，将一个实体分配了多个不同的标识符，导致标识符不唯一。
3. 知识图谱在设计时，首先将一条记录作为一个 major 实体，直接与学校连接关系，导致对于某一学校的不同省份不同年份的同一专业名多次出现，在视觉感受上十分混乱。
4. 对后续的改进，我们可能会在学校和专业间增加省份的实体和年份的关系，使查询时更加直观形象，这些都是后续改进时可能会更新的。

六、实验总结与心得体会

这次实验，我的两个任务分别是设计数据模式和导入 neo4j。在设计数据模式这部分，我主要和王靖婷、罗琦晴三人合作，多次沟通，在不断的尝试和试错中共同得到了最后一个比较合理的数据模式，也为后续工作带来了很大的便利；neo4j 是我对知识图谱的第一次接触。这次实验也让我初步了解知识图谱技术，以及 neo4j 的使用。此外，一个任务分给一百个人做也是之前从来没有过的尝试，这次尝试也让我感受到了并不是人多就一定能把事情完美的做好，在组与组之间的沟通，组内成员的分工方面，我个人认为都出现了很大的问题，这些问题一方面造成了进度迟迟难以推进，另一方面也造成了严重的冗余，使工作效率低下，这些都是我在以后的学习和之后工作时要格外注意的，如何高效沟通，如何合理分配任务，使任务更明确，是之后我会更加关注的方面。但多人分工合作也让我意识到一个很难的项目在分成一个一个小任务之后实现起来就很容易了。

2020 年 9 月制