Peter Tadrous

CUS 725 - Advanced Database Systems

Project 3

Examining the data

The first step before creating any constraints or loading any data would be to examine the csv files to understand what the data looks like. We can see there are 27 columns in the Appts-Fixed.csv file, and 9 columns in Positions.csv. We can, however, skip the 7 columns DeptScandalFlag, ScandalFlag, Bachelors, Masters, MD, Doctorate and NotesDiscrepanices in Appts-Fixed.csv, leaving us with 20.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Key	Administra	Party	Departmer	Name	LastName	ServiceSta	ServiceEnc	DeptScand	ScandalFla	ScandalDe	EduDesc	Bachelors	Masters	MD
2	1	Washingtc	None, Fed	President	George Wa	Washingtc	1789-04-3	1797-03-03				College of	William and	Mary	
3	2	Washingtc	None, Fed	Vice Presic	John Adam	Adams	1789-04-3	1797-03-03				Harvard U	1		
4	3	Washingtc	None, Fed	Secretary (	John Jay	Jay	1789-04-3	1789-09-26				Columbia I	1	1	
5	4	Washingtc	None, Fed	Secretary (	Thomas Je	Jefferson	1789-09-2	1794-12-02				College of	1		
6	5	Washingtc	None, Fed	Secretary (	Edmund Ra	Randolph	1794-12-0	1795-08-20				College of	1		
7	6	Washingtc	None, Fed	Secretary (	Timothy Pi	Pickering	1795-12-1	1800-05-12				Harvard U	1		
8	7	Washingtc	None, Fed	Secretary (	Alexander	Hamilton	1789-09-1	1795-02-02				Columbia University			
9	8	Washingtc	None, Fed	Secretary (	Oliver Wol	Wolcott	1795-02-0	1801-01-01				Yale Unive	1		
10	9	Washingtc	None, Fed	Secretary (	Henry Kno	Knox	1789-09-1	1795-01-02				no college			

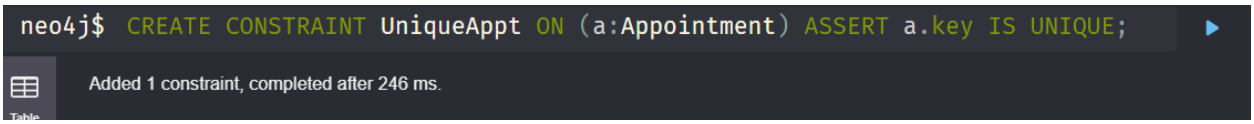
	A	B	C	D	E	F	G	H	I
1	Departmer	Appointme	Establishe	Removed	CabinetLe	FormerNa	FormerDat	RenamedT	DateRenamed
2	Bureau of	Director o	1939	1970	0			Office of M	1970
3	Central Int	Director o	1995	2001	1				
4	Council of	Chair of th	1946		0				
5	Departmer	Secretary (	1862		1				
6	Departmer	Secretary (	1903		1	Secretary (	1913		

Some columns should be displayed as dates and some as integers, so we need to make sure when loading the data in that we handle these values correctly.

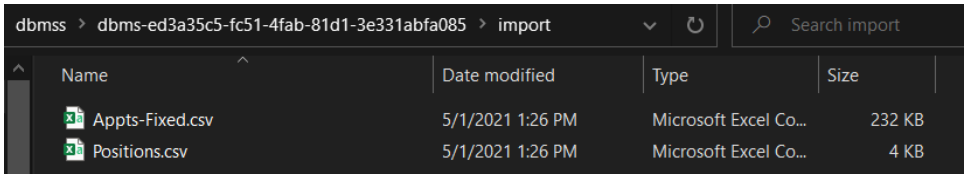
Loading the data

The first step is to create the constraints on key property in appointments so that they are unique, using the following statement:

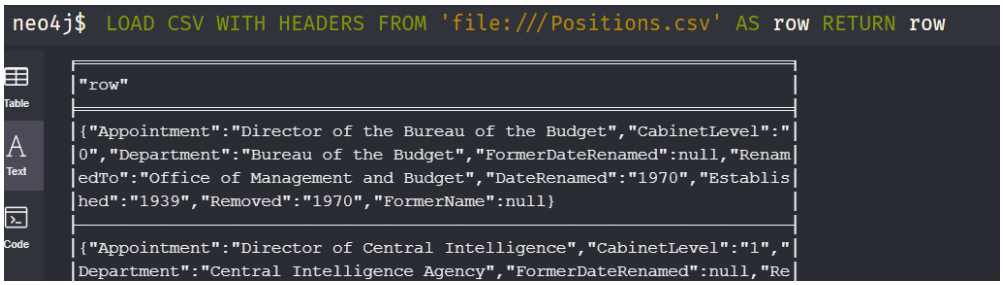
```
CREATE CONSTRAINT UniqueAppt ON (a:Appointment) ASSERT a.key IS UNIQUE;
```



Then I added the csv files to the database's import folder for my load statements.

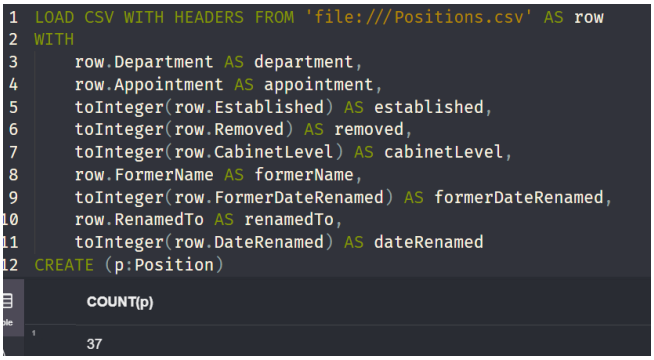


I want to make sure I can access the data correctly so I will just return the row for now. We can see that the data is loading in just fine:



Now I can add the **37 positions** to the database using **CREATE** (**note**, this was done a second time for reason stated below):

```
LOAD CSV WITH HEADERS FROM 'file:///Positions.csv' AS row WITH row.Department
AS department, row.Appointment AS appointment, toInteger(row.Established) AS
established, toInteger(row.Removed) AS removed, toInteger(row.CabinetLevel) AS
cabinetLevel, row.FormerName AS formerName, toInteger(row.FormerDateRenamed) AS
formerDateRenamed, row.RenamedTo AS renamedTo, toInteger(row.DateRenamed) AS
dateRenamed CREATE (p:Position) SET p.department = department, p.appointment =
appointment, p.established = established, p.removed = removed, p.cabinetLevel =
cabinetLevel, p.formerName = formerName, p.formerDateRenamed =
formerDateRenamed, p.renamedTo = renamedTo, p.dateRenamed = dateRenamed RETURN
COUNT(p)
```



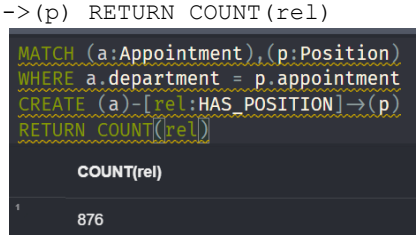
And I can add the 905 appointments using **MERGE** on the key property (**note**, this was done a second time for reason stated below, **with only 904 appointments** that second time):

```
LOAD CSV WITH HEADERS FROM 'file:///Appts-Fixed.csv' AS row WITH
toInteger(row.Key) AS key, row.Administration AS administration, row.Party AS
party, row.Department AS department, row.Name AS name, row.LastName AS
lastName, date(row.ServiceStart) AS serviceStart, date(row.ServiceEnd) AS
serviceEnd, row.ScandalDesc AS scandalDesc, row.EduDesc AS eduDesc,
toInteger(row.Law) AS law, row.MilitaryService AS militaryService,
row.MilitaryDatesServed AS militaryDatesServed, row.MilitaryBranch AS
militaryBranch, toInteger(row.Gender) AS gender, toInteger(row.ForeignBorn)
AS foreignBorn, toInteger(row.Minority) AS minority,
toInteger(row.DiedOffice) AS diedOffice, row.SenateVotesFor AS senateVotesFor,
row.SenateVotesAgainst AS senateVotesAgainst MERGE (a:Appointment {key: key})
SET a.key = key, a.administration = administration, a.party = party,
a.department = department, a.name = name, a.lastName = lastName,
a.serviceStart = serviceStart, a.serviceEnd = serviceEnd, a.scandalDesc =
scandalDesc, a.eduDesc = eduDesc, a.law = law, a.militaryService =
```

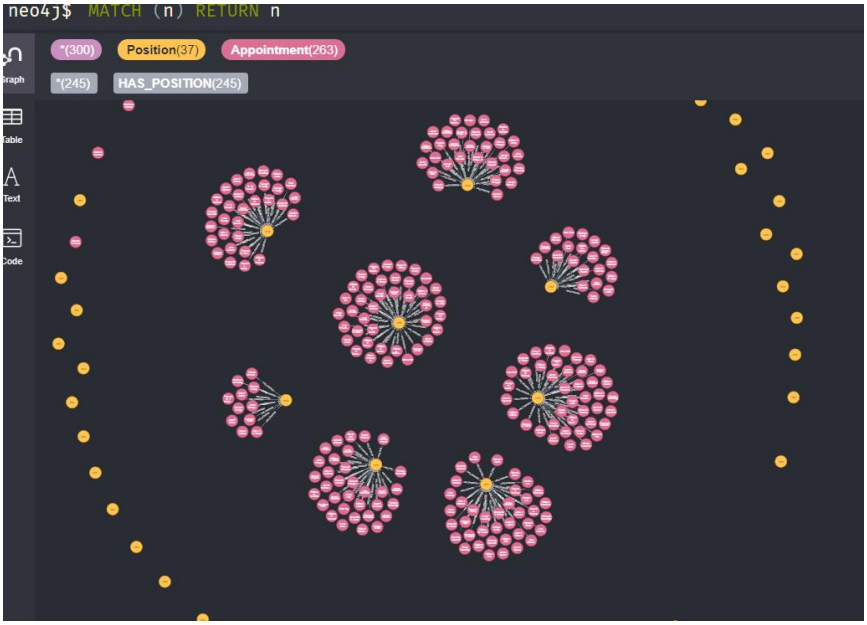
```
militaryService,  a.militaryDatesServed = militaryDatesServed,
a.militaryBranch = militaryBranch,  a.gender = gender,  a.foreignBorn =
foreignBorn,  a.minority = minority,  a.diedOffice = diedOffice,
a.senateVotesFor = senateVotesFor,  a.senateVotesAgainst = senateVotesAgainst
RETURN COUNT(a)
```

Lastly, I can add the 876 relationships, matching on a.department and p.appointment (**note**, this was done a second time for reason stated below, with **888 relationships** that second time):

```
MATCH (a:Appointment),(p:Position) WHERE a.department = p.appointment CREATE
(a)-[rel:HAS_POSITION]->(p) RETURN COUNT(rel)
```

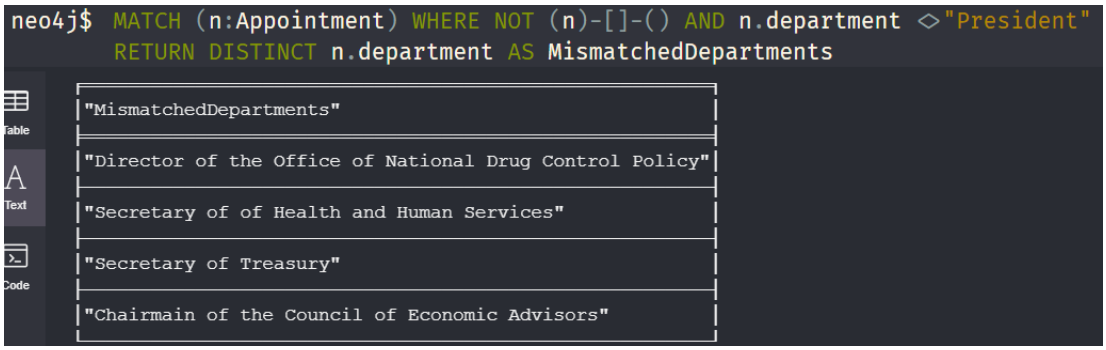


We can view the entire graph by using: `MATCH (n) RETURN n`. We can see there was a mismatch with the data because there are some appointments not matched to a position and some positions not matched to an appointment.



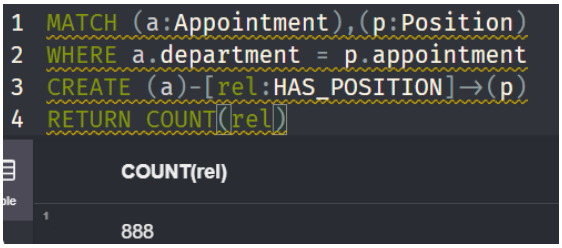
Upon closer inspection, the appointments who did not get matched to a position were majority "President", for which there is no position record. However, some of them had typos in their "Department" name. For example, two "of", or a "the" missing, "Chairman" instead of "Chair", or a trailing whitespace. I chose to correct these 12 records, as well as deleting the very last Appointment record since it is blank.

```
MATCH (n:Appointment) WHERE NOT (n)-[]-() AND n.department <> "President" RETURN
DISTINCT n.department AS MismatchedDepartments
```



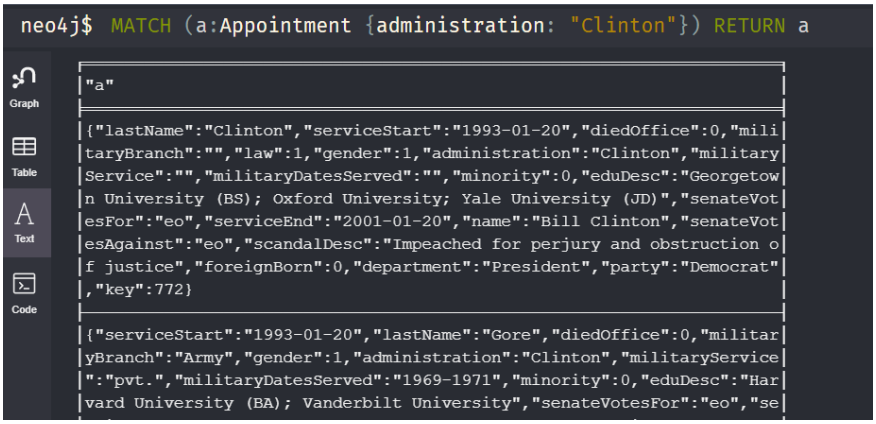
After correcting the Appointment records, I now have **888 relationships**, matching on a.department and p.appointment:

```
MATCH (a:Appointment),(p:Position) WHERE a.department = p.appointment CREATE
(a)-[rel:HAS_POSITION]->(p) RETURN COUNT(rel)
```

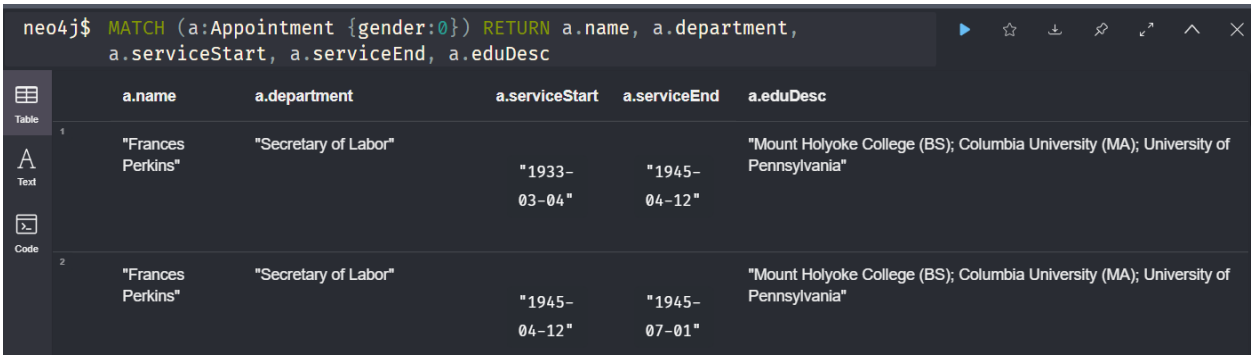


Queries

1. Return all appointments made in Clinton's administration.
   - `MATCH (a:Appointment {administration: "Clinton"}) RETURN a`



2. Return all the name, position, start date, end date, and college for all female cabinet appointments.
   - `MATCH (a:Appointment {gender:0}) RETURN a.name, a.department, a.serviceStart, a.serviceEnd, a.eduDesc`



```
neo4j$ MATCH (a:Appointment {gender:0}) RETURN a.name, a.department,
       a.serviceStart, a.serviceEnd, a.eduDesc
```

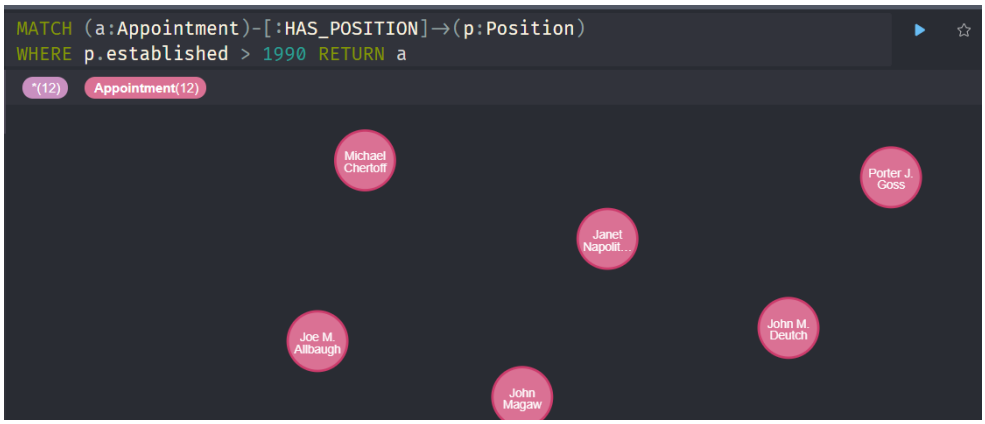
	a.name	a.department	a.serviceStart	a.serviceEnd	a.eduDesc
1	"Frances Perkins"	"Secretary of Labor"	"1933-03-04"	"1945-04-12"	"Mount Holyoke College (BS); Columbia University (MA); University of Pennsylvania"
2	"Frances Perkins"	"Secretary of Labor"	"1945-04-12"	"1945-07-01"	"Mount Holyoke College (BS); Columbia University (MA); University of Pennsylvania"

3. Return the name, start date and administration for all postmaster generals from 1850 to 1950. Note – you can do date comparisons and you can specify a date constant like this: date("1850-01-01")
   - `MATCH (a:Appointment)-[:HAS_POSITION]->(p:Position {appointment:"Postmaster General"}) WHERE a.serviceStart > date('1850-01-01') AND a.serviceStart < date('1950-01-01') RETURN a.name, a.serviceStart, a.administration`

```
1 MATCH (a:Appointment)-[:HAS_POSITION]→(p:Position {appointment:"Postmaster General"})
2 WHERE a.serviceStart > date('1850-01-01') AND a.serviceStart < date('1950-01-01')
3 RETURN a.name, a.serviceStart, a.administration
```

	a.name	a.serviceStart	a.administration
1	"Charles E. Smith"	"1901-09-14"	"Theodore Roosevelt"
2	"James A. Gary"	"1897-03-05"	"McKinley"
3	"William Dennison"		"Lincoln"

4. Return all appointments in positions established after 1990. Note that year established in Position is an integer rather than a date.
   - `MATCH (a:Appointment)-[:HAS_POSITION]->(p:Position) WHERE p.established > 1990 RETURN a`



5. Return all appointments in positions that have been renamed.
    - `MATCH (a:Appointment)-[]->(p:Position) WHERE p.renamedTo IS NOT NULL RETURN a.name, a.department, p.renamedTo`

```
MATCH (a:Appointment)-[]→(p:Position)
WHERE p.renamedTo IS NOT NULL
RETURN a.name, a.department, p.renamedTo
```

	a.name	a.department	p.renamedTo
1	"Oscar S. Straus"	"Secretary of Commerce and Labor"	"Secretary of Commerce, Secretary of Labor"
2	"George B. Cortelyou"	"Secretary of Commerce and Labor"	"Secretary of Commerce, Secretary of Labor"
3	"Victor H. Metcalf"	"Secretary of Commerce and Labor"	"Secretary of Commerce, Secretary of Labor"
4	"Charles Nagel"	"Secretary of Commerce and Labor"	"Secretary of Commerce, Secretary of Labor"

6. Return all appointments and the department name where the position name contains LABOR.
    - `MATCH (a:Appointment) WHERE toUpper(a.department) CONTAINS "LABOR" RETURN a.name, a.department`

```
MATCH (a:Appointment)
WHERE toUpper(a.department) CONTAINS "LABOR"
RETURN a.name, a.department
```

	a.name	a.department
1	"George B. Cortelyou"	"Secretary of Commerce and Labor"
2	"Victor H. Metcalf"	"Secretary of Commerce and Labor"
3	"Oscar S. Straus"	"Secretary of Commerce and Labor"

7. Return the appointment name, position, administration, and year the position was established, ordered by the year established.
    - `MATCH (a)-[]->(p) RETURN a.name, a.department, a.administration, p.established ORDER BY p.established`

```
neo4j$ MATCH (a)-[]→(p) RETURN a.name, a.department, a.administration,
       p.established ORDER BY p.established
```

Table
Text
Code

	a.name	a.department	a.administration	p.established
419	"Hubert Work"	"Postmaster General"	"Harding"	1829
420	"Walter Q. Gresham"	"Postmaster General"	"Arthur"	1829
421	"James N. Tyner"	"Postmaster General"	"Grant"	1829

8. Return a count of appointments for people who served in the navy.
   - `MATCH (a:Appointment) WHERE a.militaryBranch CONTAINS "Navy" RETURN COUNT(a)`

```
4j$ MATCH (a:Appointment) WHERE a.militaryBranch CONTAINS "Navy" RETURN
    COUNT(a)
```

COUNT(a)
61

9. Return the number of women in appointments grouped by department.
   - `MATCH (a:Appointment {gender:0}) RETURN a.department AS Department, COUNT(*) AS CountOfWomen`

```
1 MATCH (a:Appointment {gender:0})
2 RETURN a.department AS Department, COUNT(*) AS CountOfWomen
```

"Department"	"CountOfWomen"
"Secretary of Labor"	8
"Secretary of Health, Education, and Welfare"	1
"Secretary of Housing and Urban Development"	2
"Secretary of Commerce"	3
"Secretary of Health and Human Services"	5

10. Return the number of appointees who served in military grouped by service branch (**note**, since some values are semicolon delimited, this is not an entirely accurate count).
    - `MATCH (a:Appointment) WHERE a.militaryBranch <> "" RETURN a.militaryBranch AS MilitaryBranch, COUNT(*) AS NumServed`

```
MATCH (a:Appointment) WHERE a.militaryBranch <> ""
RETURN a.militaryBranch AS MilitaryBranch, COUNT(*) AS NumServed
```

"MilitaryBranch"	"NumServed"
"Army"	180
"Militia/National Guard"	23
"Army; Militia/National Guard; Army"	1
"Army; Militia/National Guard"	6
"Militia/National Guard; Army; Army"	2
"Navy"	59
"Army; Army"	9