Data Warehouse Project 2 Submission PDF

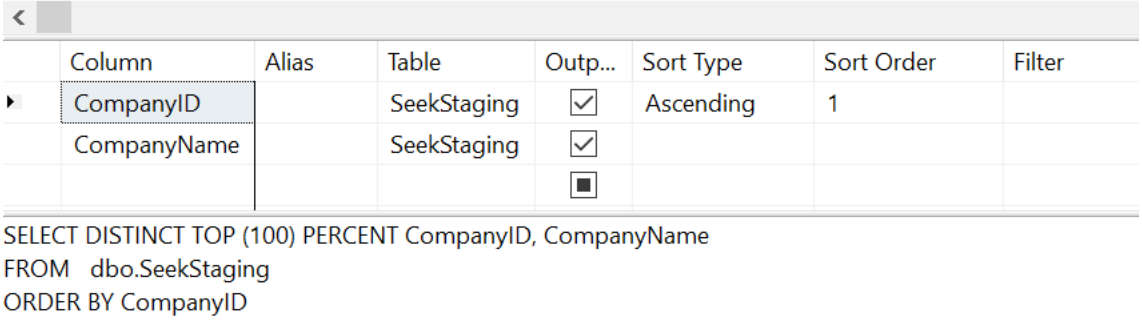
Lachlan Campbell 21498227

Association Rule Mining

1.Preparation

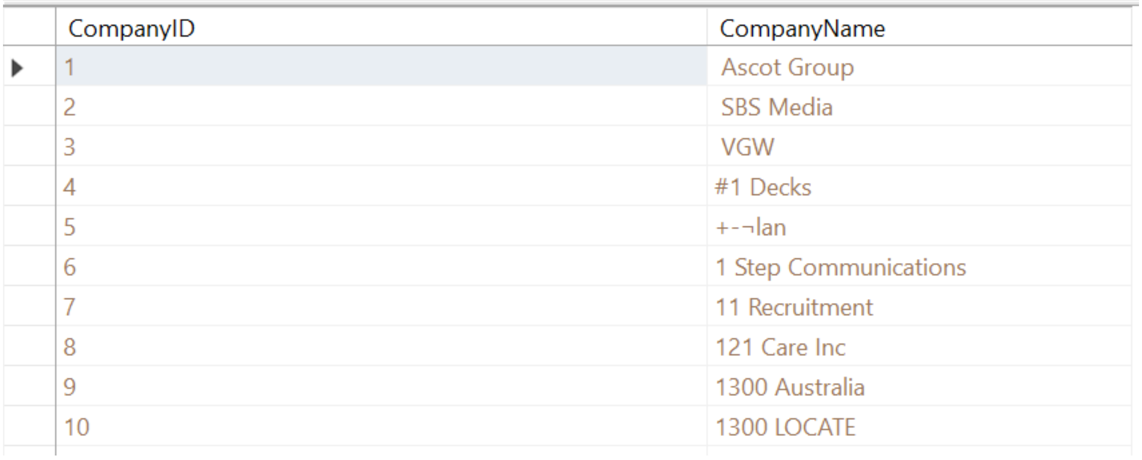
In preparation for the Association Rule mining, I created in SSMS:

1. A view which queried the distinct Companies plus their ID for the case table:



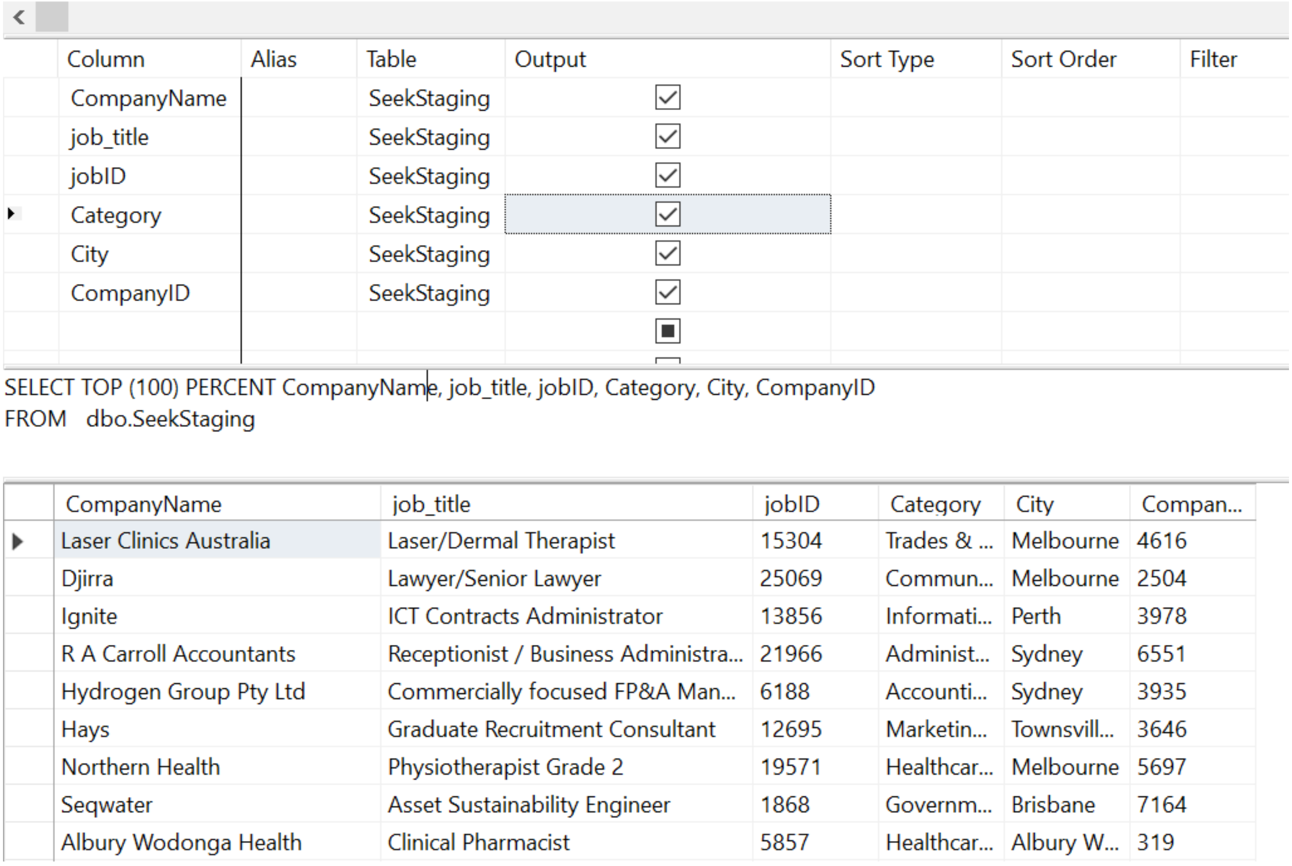
Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter
CompanyID		SeekStaging	☑	Ascending	1	
CompanyName		SeekStaging	☑			
			■			

```
SELECT DISTINCT TOP (100) PERCENT CompanyID, CompanyName
FROM   dbo.SeekStaging
ORDER BY CompanyID
```



CompanyID	CompanyName
1	Ascot Group
2	SBS Media
3	VGW
4	#1 Decks
5	+-¬lan
6	1 Step Communications
7	11 Recruitment
8	121 Care Inc
9	1300 Australia
10	1300 LOCATE

2. A view of all job postings plus the foreign key of the company that posted the ad for the nested table:



Column	Alias	Table	Output	Sort Type	Sort Order	Filter
CompanyName		SeekStaging	☑			
job_title		SeekStaging	☑			
jobID		SeekStaging	☑			
Category		SeekStaging	☑			
City		SeekStaging	☑			
CompanyID		SeekStaging	☑			
			■			

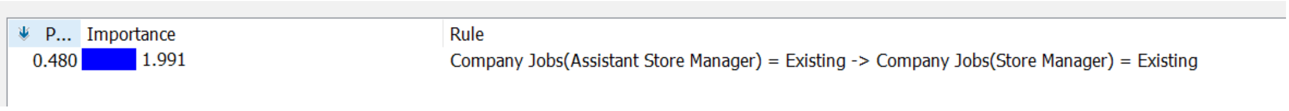
```
SELECT TOP (100) PERCENT CompanyName, job_title, jobID, Category, City, CompanyID
FROM   dbo.SeekStaging
```

CompanyName	job_title	jobID	Category	City	Compan...
Laser Clinics Australia	Laser/Dermal Therapist	15304	Trades & ...	Melbourne	4616
Djirra	Lawyer/Senior Lawyer	25069	Commun...	Melbourne	2504
Ignite	ICT Contracts Administrator	13856	Informati...	Perth	3978
R A Carroll Accountants	Receptionist / Business Administra...	21966	Administ...	Sydney	6551
Hydrogen Group Pty Ltd	Commercially focused FP&A Man...	6188	Accounti...	Sydney	3935
Hays	Graduate Recruitment Consultant	12695	Marketin...	Townsvill...	3646
Northern Health	Physiotherapist Grade 2	19571	Healthcar...	Melbourne	5697
Seqwater	Asset Sustainability Engineer	1868	Governm...	Brisbane	7164
Albury Wodonga Health	Clinical Pharmacist	5857	Healthcar...	Albury W...	319

For the nested table, I included columns which I felt could be used to discover useful association rules.

2.Mining

For Job association rules between companies, my model discovered only one association rule:



P...	Importance	Rule
0.480	■ 1.991	Company Jobs(Assistant Store Manager) = Existing -> Company Jobs(Store Manager) = Existing

Figure 1: Association rule discovered between jobs

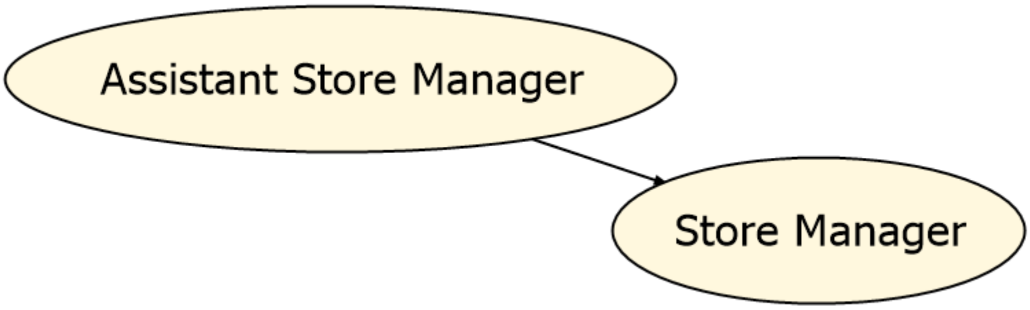
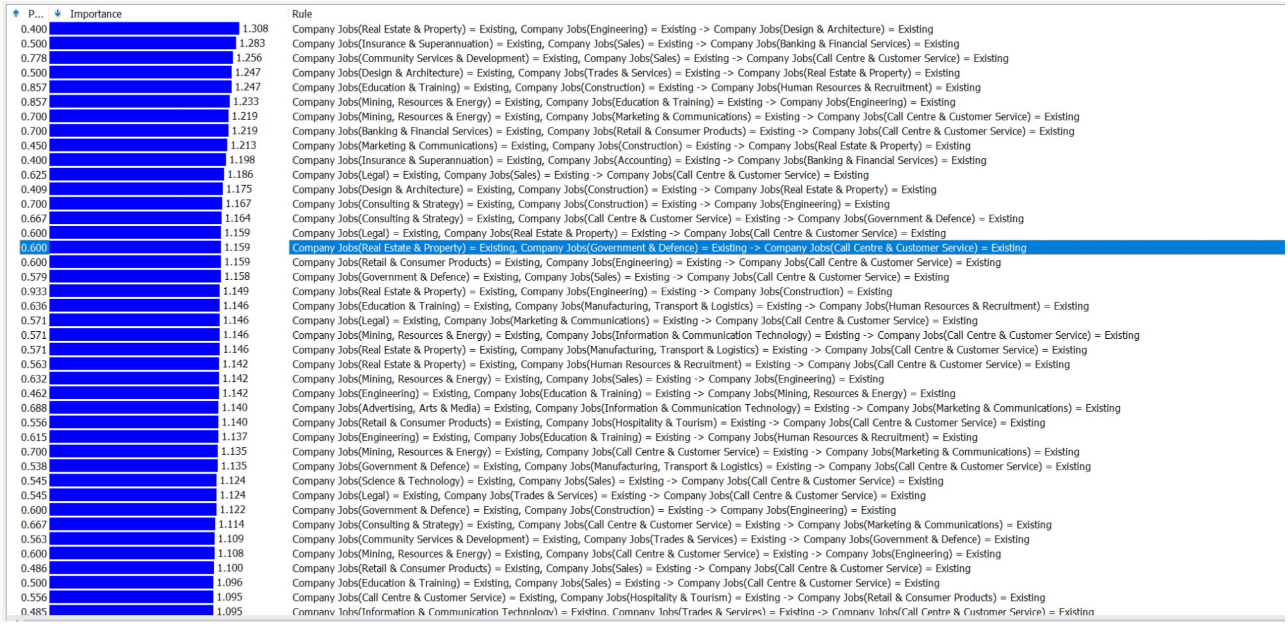


Figure 2: Network Graph for association rule between Job titles.

This rule means that if a company posts an ad for an “Assistant Store Manager”, they are likely to post an ad for a “Store Manager”.

The limited number of association rules found may be due to my data cleaning process. Initially, my data returned no association rules for jobs posted by companies, but revealed the above once I used a less aggressive fuzzy grouping for the job titles. There will likely be many more association rules that will be revealed once I had a more rigorous data cleaning, to allow relevant jobs to be properly grouped together, Instead of being processed as distinct. Many companies have posted Job titles that differ in text quite significantly from another company for what would rationally be regarded as the same job. A lot of job postings also include things like salaries, locations, and other non relevant information appended to the job_title field which complicates filtering, and that I have not completely resolved.

I found some other Association rules that may be useful. For Categories of jobs posted, we can look at what category a company is likely to post, if they have already posted ads in certain categories:



P...	Importance	Rule
0.400	1.308	Company Jobs(Real Estate & Property) = Existing, Company Jobs(Engineering) = Existing -> Company Jobs(Design & Architecture) = Existing
0.500	1.283	Company Jobs(Insurance & Superannuation) = Existing, Company Jobs(Sales) = Existing -> Company Jobs(Banking & Financial Services) = Existing
0.778	1.256	Company Jobs(Community Services & Development) = Existing, Company Jobs(Sales) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.500	1.247	Company Jobs(Design & Architecture) = Existing, Company Jobs(Trades & Services) = Existing -> Company Jobs(Real Estate & Property) = Existing
0.857	1.247	Company Jobs(Education & Training) = Existing, Company Jobs(Construction) = Existing -> Company Jobs(Human Resources & Recruitment) = Existing
0.857	1.233	Company Jobs(Mining, Resources & Energy) = Existing, Company Jobs(Education & Training) = Existing -> Company Jobs(Engineering) = Existing
0.700	1.219	Company Jobs(Mining, Resources & Energy) = Existing, Company Jobs(Marketing & Communications) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.700	1.219	Company Jobs(Banking & Financial Services) = Existing, Company Jobs(Retail & Consumer Products) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.450	1.213	Company Jobs(Marketing & Communications) = Existing, Company Jobs(Construction) = Existing -> Company Jobs(Real Estate & Property) = Existing
0.400	1.198	Company Jobs(Insurance & Superannuation) = Existing, Company Jobs(Accounting) = Existing -> Company Jobs(Banking & Financial Services) = Existing
0.625	1.186	Company Jobs(Legal) = Existing, Company Jobs(Sales) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.409	1.175	Company Jobs(Design & Architecture) = Existing, Company Jobs(Construction) = Existing -> Company Jobs(Real Estate & Property) = Existing
0.700	1.167	Company Jobs(Consulting & Strategy) = Existing, Company Jobs(Construction) = Existing -> Company Jobs(Engineering) = Existing
0.667	1.164	Company Jobs(Consulting & Strategy) = Existing, Company Jobs(Call Centre & Customer Service) = Existing -> Company Jobs(Government & Defence) = Existing
0.600	1.159	Company Jobs(Legal) = Existing, Company Jobs(Real Estate & Property) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.600	1.159	Company Jobs(Real Estate & Property) = Existing, Company Jobs(Government & Defence) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.600	1.159	Company Jobs(Retail & Consumer Products) = Existing, Company Jobs(Engineering) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.579	1.158	Company Jobs(Government & Defence) = Existing, Company Jobs(Sales) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.933	1.149	Company Jobs(Real Estate & Property) = Existing, Company Jobs(Engineering) = Existing -> Company Jobs(Construction) = Existing
0.636	1.146	Company Jobs(Education & Training) = Existing, Company Jobs(Manufacturing, Transport & Logistics) = Existing -> Company Jobs(Human Resources & Recruitment) = Existing
0.571	1.146	Company Jobs(Legal) = Existing, Company Jobs(Marketing & Communications) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.571	1.146	Company Jobs(Mining, Resources & Energy) = Existing, Company Jobs(Information & Communication Technology) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.571	1.146	Company Jobs(Real Estate & Property) = Existing, Company Jobs(Manufacturing, Transport & Logistics) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.563	1.142	Company Jobs(Real Estate & Property) = Existing, Company Jobs(Human Resources & Recruitment) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.632	1.142	Company Jobs(Mining, Resources & Energy) = Existing, Company Jobs(Sales) = Existing -> Company Jobs(Engineering) = Existing
0.462	1.142	Company Jobs(Engineering) = Existing, Company Jobs(Education & Training) = Existing -> Company Jobs(Mining, Resources & Energy) = Existing
0.688	1.140	Company Jobs(Advertising, Arts & Media) = Existing, Company Jobs(Information & Communication Technology) = Existing -> Company Jobs(Marketing & Communications) = Existing
0.556	1.140	Company Jobs(Retail & Consumer Products) = Existing, Company Jobs(Hospitality & Tourism) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.615	1.137	Company Jobs(Engineering) = Existing, Company Jobs(Education & Training) = Existing -> Company Jobs(Human Resources & Recruitment) = Existing
0.700	1.135	Company Jobs(Mining, Resources & Energy) = Existing, Company Jobs(Call Centre & Customer Service) = Existing -> Company Jobs(Marketing & Communications) = Existing
0.538	1.135	Company Jobs(Government & Defence) = Existing, Company Jobs(Manufacturing, Transport & Logistics) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.545	1.124	Company Jobs(Science & Technology) = Existing, Company Jobs(Sales) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.545	1.124	Company Jobs(Legal) = Existing, Company Jobs(Trades & Services) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.600	1.122	Company Jobs(Government & Defence) = Existing, Company Jobs(Construction) = Existing -> Company Jobs(Engineering) = Existing
0.667	1.114	Company Jobs(Consulting & Strategy) = Existing, Company Jobs(Call Centre & Customer Service) = Existing -> Company Jobs(Marketing & Communications) = Existing
0.563	1.109	Company Jobs(Community Services & Development) = Existing, Company Jobs(Trades & Services) = Existing -> Company Jobs(Government & Defence) = Existing
0.600	1.108	Company Jobs(Mining, Resources & Energy) = Existing, Company Jobs(Call Centre & Customer Service) = Existing -> Company Jobs(Engineering) = Existing
0.486	1.100	Company Jobs(Retail & Consumer Products) = Existing, Company Jobs(Sales) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.500	1.096	Company Jobs(Education & Training) = Existing, Company Jobs(Sales) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing
0.556	1.095	Company Jobs(Call Centre & Customer Service) = Existing, Company Jobs(Hospitality & Tourism) = Existing -> Company Jobs(Retail & Consumer Products) = Existing
0.485	1.095	Company Jobs(Information & Communication Technology) = Existing, Company Jobs(Trades & Services) = Existing -> Company Jobs(Call Centre & Customer Service) = Existing

There are many rules found between categories. The top one means that if a company posts an ad for a job in both “Real Estate & Property” and “Engineering”, they are likely to also post an ad in the category of “Design & Architecture”. Visual Studio was unable to generate a network chart for these rules.

Neo4J Graph Database

1.Design

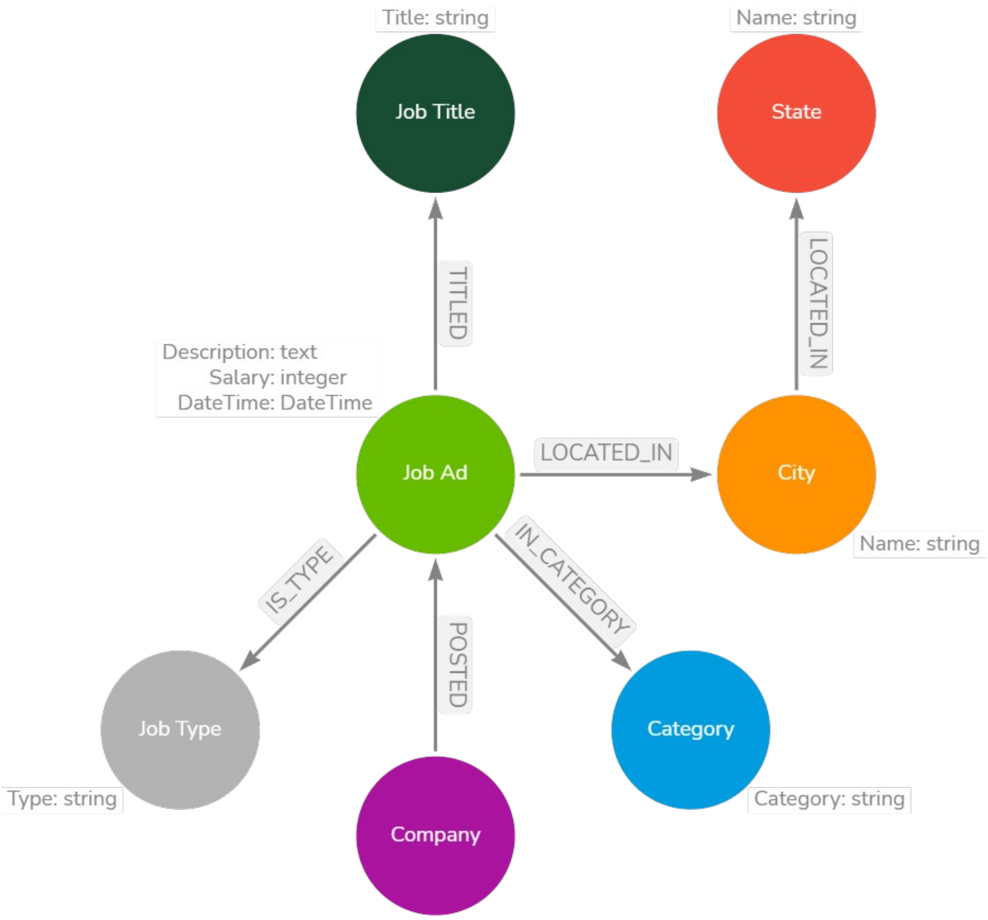


Figure 4: Design of my Graph Database

I designed my graph around the questions in the project specification, aiming for efficiency in queries about relations. However, because many of the records in the original data are now represented as nodes, it will be difficult to do complex analysis over aggregate data for those records (not the case for Job Ad properties such as dates, salary etc.).

The Questions:

1. How many jobs are advertised for a given job category in a specified city?
2. Find job_ids that share the same job_title
3. Find all companies that offer jobs in different categories
4. find jobs based on the presence of a keyword

5. find jobs posted during a specified period of time

How each question can be answered efficiently with this design:

1. Look at which Ad nodes are connected to both the specified category node and city node. Does not offer any efficiency over a RDBMS.
2. Look at each Ad, return the ID property of those related to the specified title node. Does not offer any efficiency over a RDBMS.
3. We need to search depth 1 without branching. We only look at 2 nodes and 2 relations for each ad posted for each company (worst case).
4. All keywords related to each ad will either be found in the description, or in the Job Title. This requires looking at each Ad node once, and a single related node per Ad(constant time). Does not offer any efficiency over a RDBMS.
5. Simple comparison over each Ad node. No efficiency over a RDBMS. If the dates were nodes, rather than a property, we would have to look at the relations between the dates and the Ads, while still looking at each Ad node.

Some more specific pros and cons about this design:

Pro's:

- Ad specific details as properties allows quick comparison on simple queries without blowing up the size of the graph unnecessarily (uniqueness of description and date-time)
- We can visualise relationships between jobs, categories, companies etc. easily
- Job Title is a separate node which allows us to more easily visualise some queries such as which categories certain job titles usually are in, or which type are they normally
- City to State hierarchy implemented in structure
- Some queries such as 3. have better performance than a RDBMS design

Cons

- Job Title is a separate node, so some queries such as 4 may lessen performance as looking at the relationship rather than just the node properties are required
- State is branched out a few nodes away from the Ads, so queries based on states may have slightly less performance

2.Neo4J Cyphers/Construction:

The following Cyphers are to build the Database. They can be done in any order as long as the JobAds one is done last. I built all the csv files by using simple SQL queries to separate out the distinct nodes, and then used the already built SeekStaging table from project 1 to link the nodes together.

Load Cities

```
LOAD CSV WITH HEADERS FROM 'file:///DistinctCity.csv' AS row
CREATE (n:City {
   CityName: row.City
})
```

Load Categories

```
LOAD CSV WITH HEADERS FROM 'file:///DistinctCategory.csv' AS row
CREATE (n:Category {
   Category: row.Category
})
```

Load Job Types

```
LOAD CSV WITH HEADERS FROM 'file:///DistinctJobType.csv' AS row
CREATE (n:JobType {
   Type: row.job_type
})
```

Load Companies

```
LOAD CSV WITH HEADERS FROM 'file:///DistinctCompany.csv' AS row
CREATE (n:Company {
   CompanyName: row.CompanyName,
})
```

Load Job Titles

```
LOAD CSV WITH HEADERS FROM 'file:///DistinctJobTitles.csv' AS row
CREATE (n:JobTitle {
   Title: row.job_title
})
```

Load JobAds

```
LOAD CSV WITH HEADERS FROM 'file:///SeekStaging.csv' AS row
MATCH (c:Category) WHERE c.Category = row.Category
MATCH (ci:City) WHERE ci.CityName = row.City
MATCH (t:JobType) WHERE t.Type = row.job_type
MATCH (co:Company) WHERE co.CompanyName = row.CompanyName
MATCH (ti:JobTitle) WHERE ti.Title = row.job_title
CREATE (p:JobAd {
   JobTitle: row.job_title,
   Description: row.job_description,
   DateTime: datetime(row.post_date),
   Salary: toInteger(row.salary_i)

})
CREATE (p)-[:IN_CATEGORY ]->(c)
CREATE (p)-[:LOCATED_IN]->(ci)
CREATE (p)-[:IS_TYPE]->(t)
CREATE (p)-[:TITLED]->(ti)
CREATE (co)-[:POSTED]->(p)
```

3.Cypher Answers

1.How many jobs are advertised for a given job category in a specified city?

For question 1, I am using the category Accounting and Sydney as an example:

MATCH (a:JobAd)-[:IN_CATEGORY]->(c:Category {Category: "Accounting"})

MATCH(a)-[:LOCATED_IN]->(ci:City {CityName: "Sydney"})

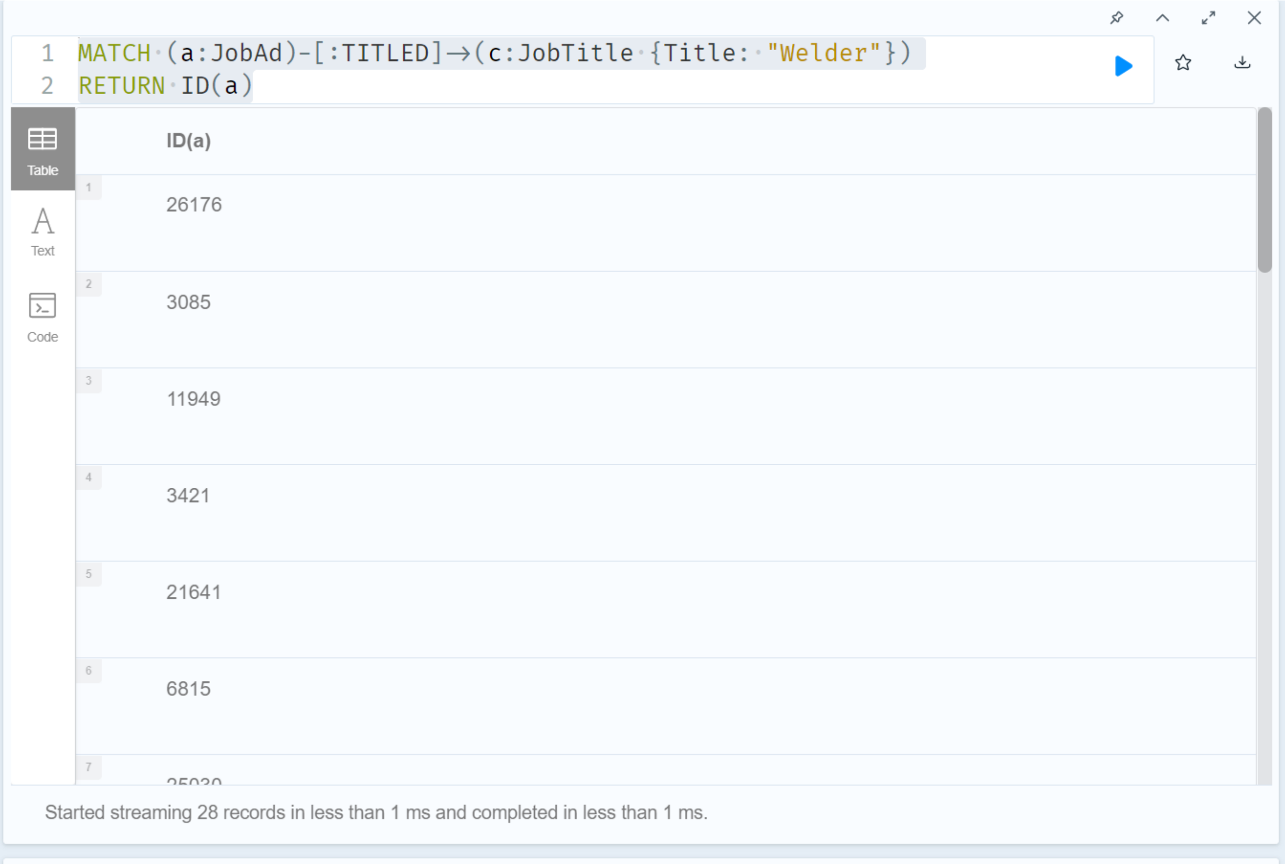
RETURN a

2.Find job_ids that share the same job_title

For question 2, I am using Welder as an example.

MATCH (a:JobAd)-[:TITLED]->(c:JobTitle {Title: "Welder"})

RETURN Id(a)



```
MATCH (a:JobAd)-[:TITLED]→(c:JobTitle {Title: "Welder"})
RETURN ID(a)
```

	ID(a)
1	26176
2	3085
3	11949
4	3421
5	21641
6	6815
7	25030

Started streaming 28 records in less than 1 ms and completed in less than 1 ms.

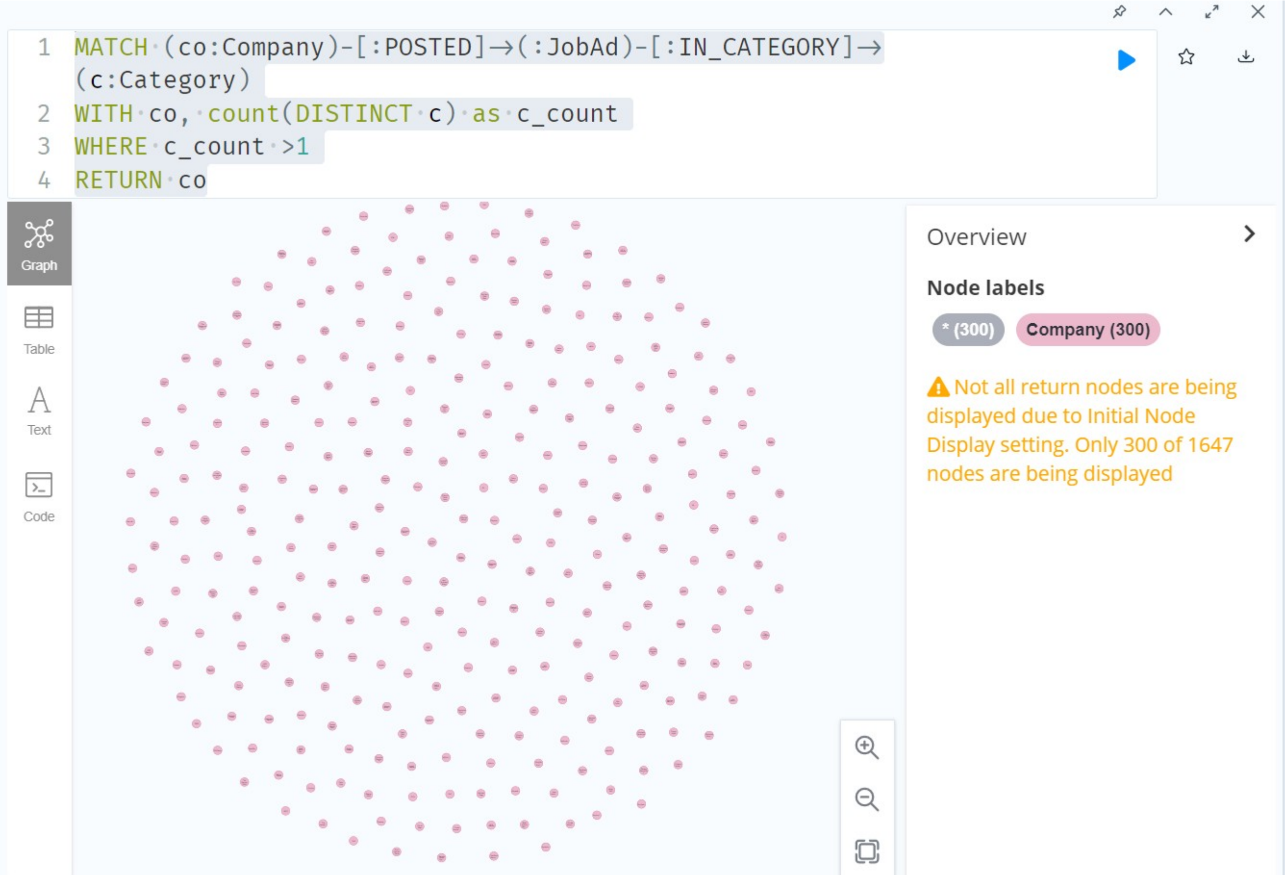
3.Find all companies that offer jobs in different categories

MATCH (co:Company)-[:POSTED]->(:JobAd)-[:IN_CATEGORY]->(c:Category)

WITH co, count(DISTINCT c) as c_count

WHERE c_count >1

RETURN co



```
MATCH (co:Company)-[:POSTED]→(:JobAd)-[:IN_CATEGORY]→
(c:Category)
WITH co, count(DISTINCT c) as c_count
WHERE c_count >1
RETURN co
```

⚠ Not all return nodes are being displayed due to Initial Node Display setting. Only 300 of 1647 nodes are being displayed

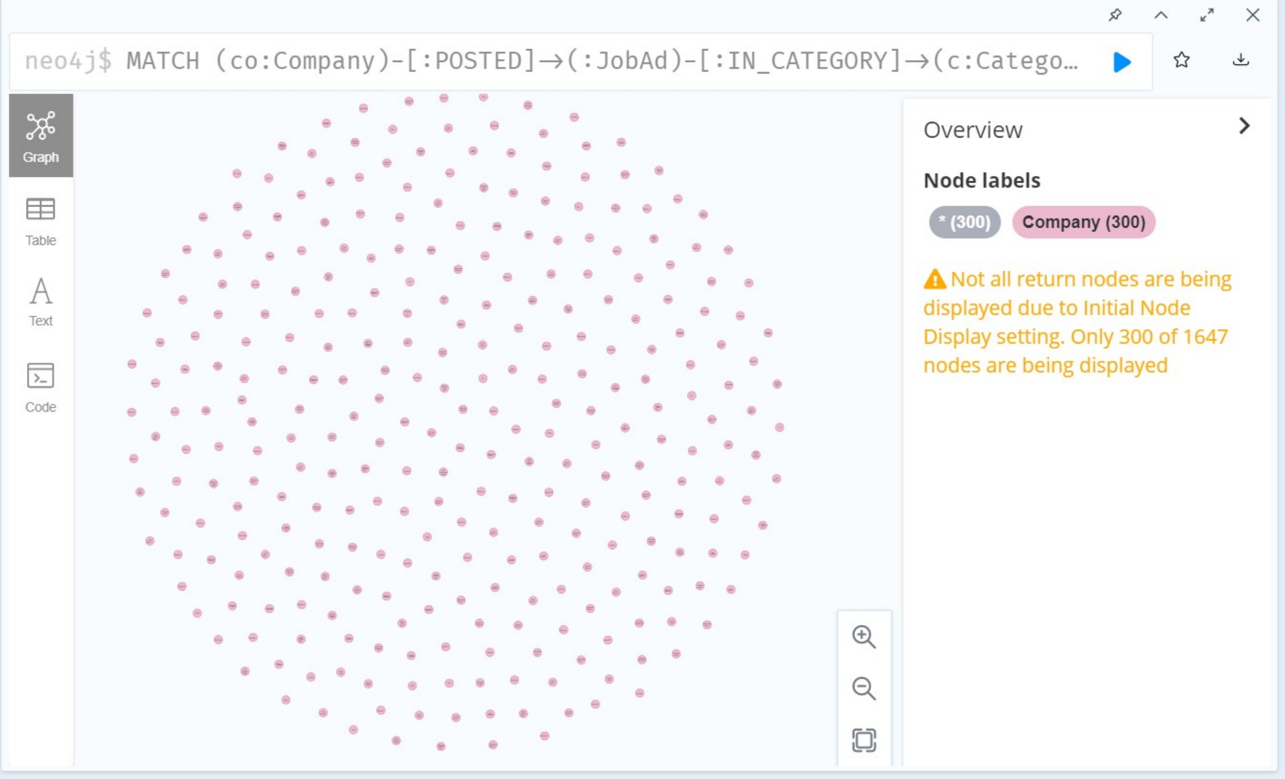
4.find jobs based on the presence of a keyword

For Question 4, I used “weld” as the keyword. I used regular expressions to search the descriptions and titles.

MATCH (a:JobAd)-[:TITLED]->(t:JobTitle)

WHERE `a.Description =~ '.*weld.*' OR t.Title =~ '.*weld.*'`

RETURN a



```
neo4j$ MATCH (co:Company)-[:POSTED]→(:JobAd)-[:IN_CATEGORY]→(c:Catego…
```

Node labels

* (300) Company (300)

⚠ Not all return nodes are being displayed due to Initial Node Display setting. Only 300 of 1647 nodes are being displayed

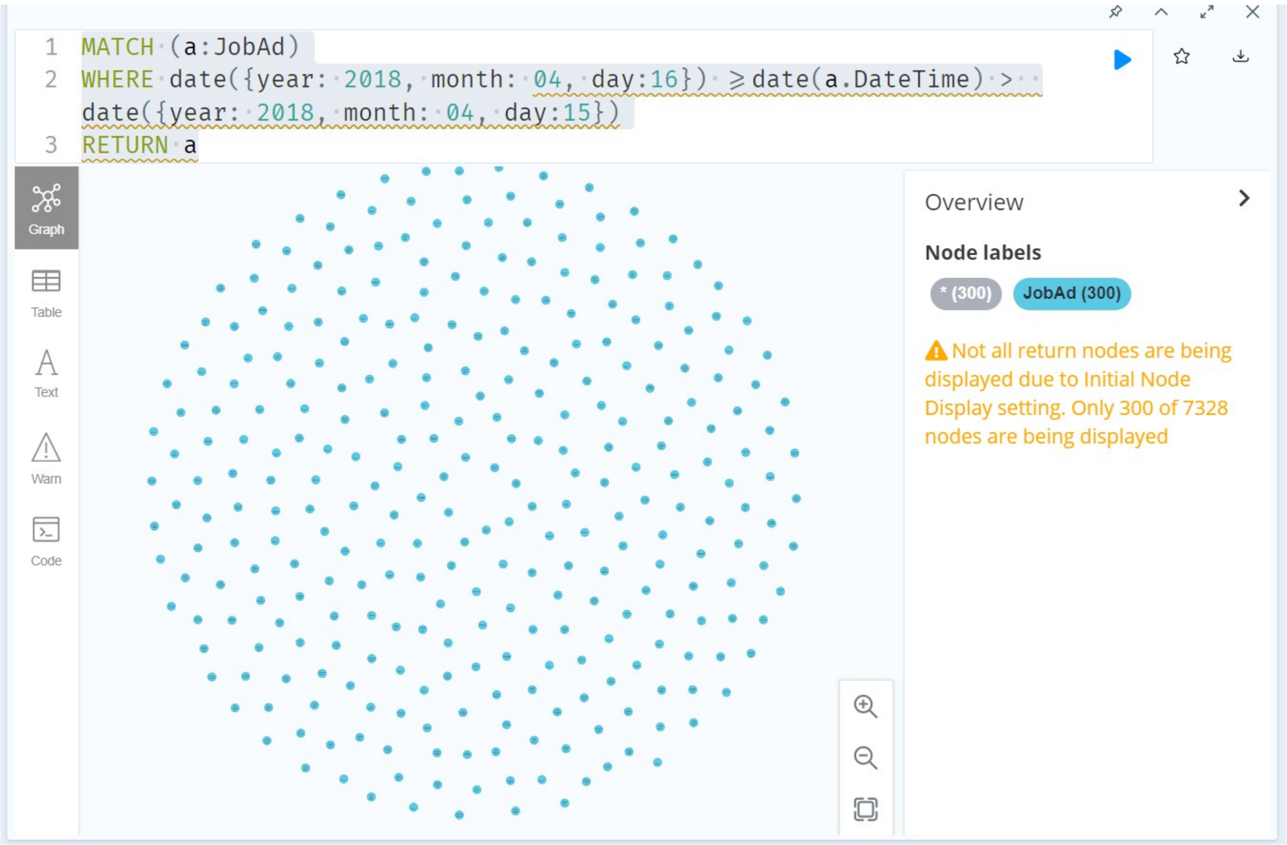
5.find jobs posted during a specified period of time.

For Question 5, I used the dates 15/04/2018 and 16/04/2018.

MATCH (a:JobAd)

WHERE date({year: 2018, month: 04, day:16}) >=date(a.DateTime) > date({year: 2018, month: 04, day:15})

RETURN a



```
MATCH (a:JobAd)
WHERE date({year: 2018, month: 04, day:16}) >= date(a.DateTime) > date({year: 2018, month: 04, day:15})
RETURN a
```

⚠ Not all return nodes are being displayed due to Initial Node Display setting. Only 300 of 7328 nodes are being displayed

Additional Questions

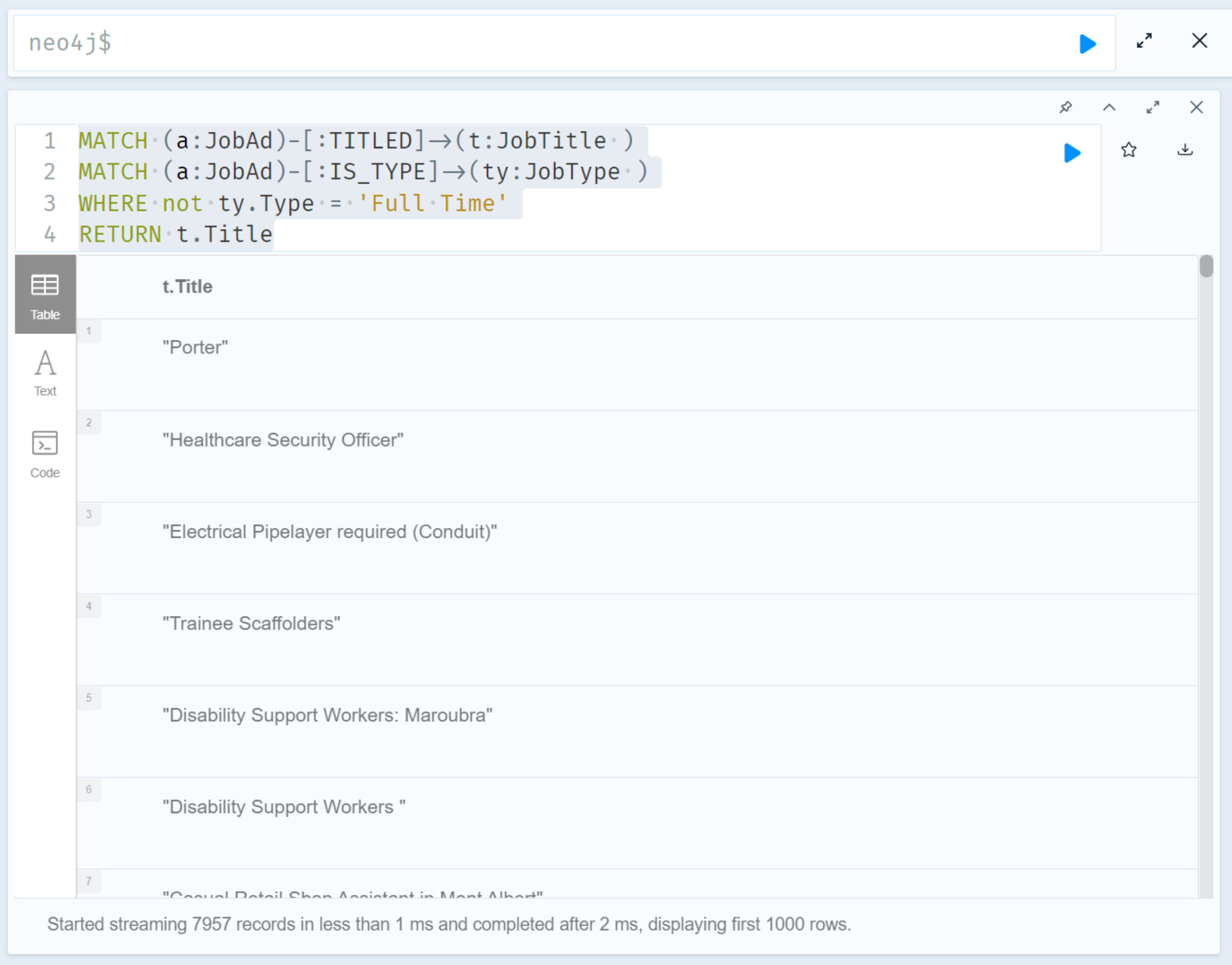
1. Find all job titles that are never advertised as full time.

MATCH (a:JobAd)-[:TITLED]->(t:JobTitle)

MATCH (a:JobAd)-[:IS_TYPE]->(ty:JobType)

WHERE not ty.Type = 'Full Time'

RETURN t.Title



```
neo4j$
```

```
MATCH (a:JobAd)-[:TITLED]→(t:JobTitle )
MATCH (a:JobAd)-[:IS_TYPE]→(ty:JobType )
WHERE not ty.Type = 'Full Time'
RETURN t.Title
```

	t.Title
1	"Porter"
2	"Healthcare Security Officer"
3	"Electrical Pipelayer required (Conduit)"
4	"Trainee Scaffolders"
5	"Disability Support Workers: Maroubra"
6	"Disability Support Workers "
7	"Casual Retail Shop Assistant in Mont Albert"

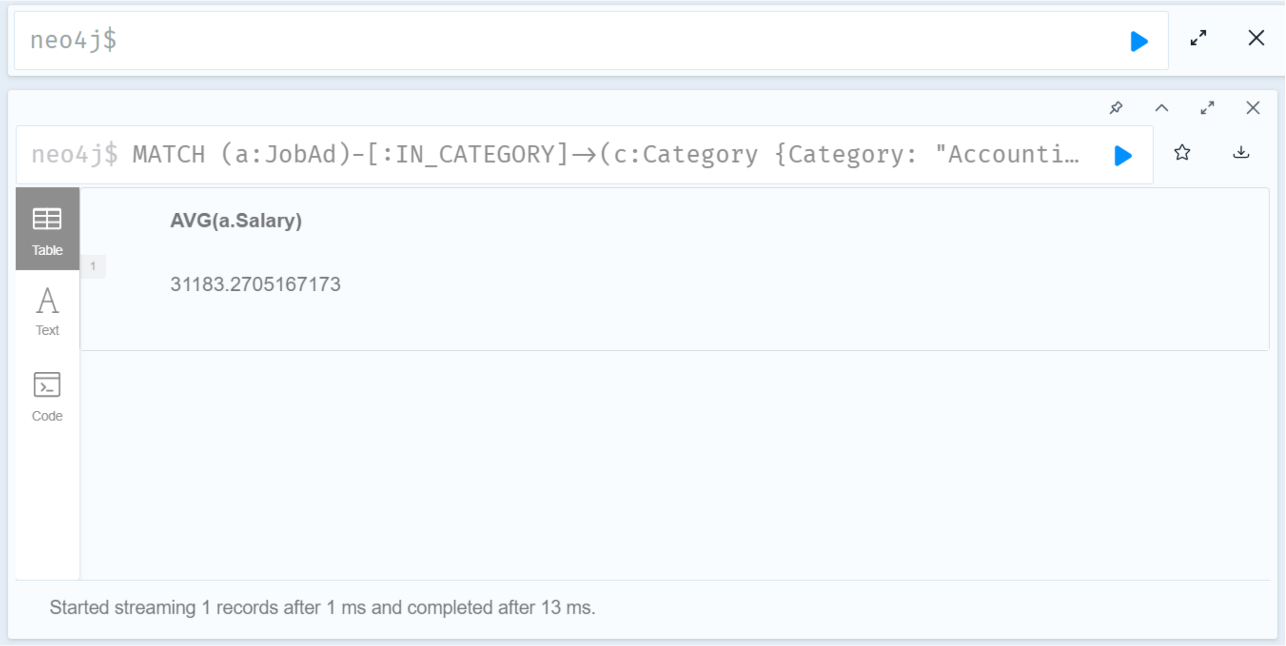
Started streaming 7957 records in less than 1 ms and completed after 2 ms, displaying first 1000 rows.

2. What is the average salary of Accounting (category) Jobs posted in Sydney?

MATCH (a:JobAd)-[:IN_CATEGORY]->(c:Category:"Accounting")

MATCH(a)-[:LOCATED_IN]->(ci:City {CityName: "Sydney"})

RETURN AVG(a.Salary)



```
neo4j$ MATCH (a:JobAd)-[:IN_CATEGORY]→(c:Category {Category: "Accounti…
```

```
AVG(a.Salary)
1  31183.2705167173
```

Started streaming 1 records after 1 ms and completed after 13 ms.

This may seem deceptively low, as many accountants will only be offered limited work from some companies. My ETL process converts all rates into annual.

Video Submission URL:

https://youtu.be/g-h7c0nFZV4