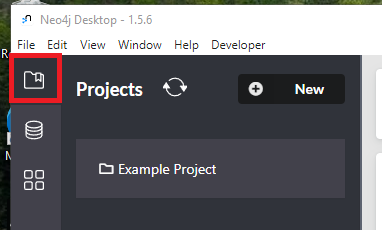
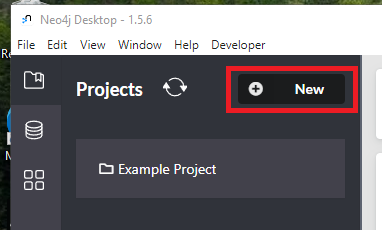
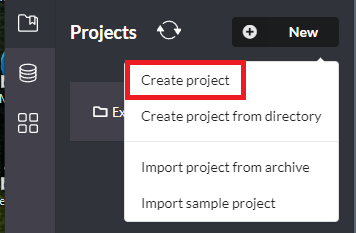
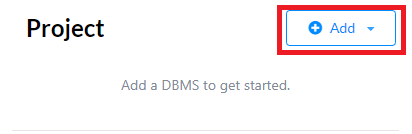
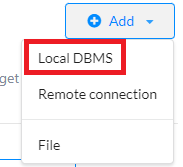
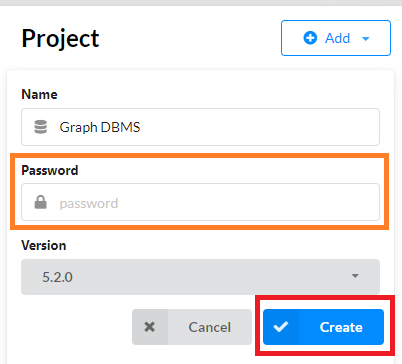
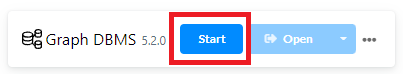
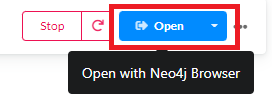
Graphing M365: Lab Three

# Objective

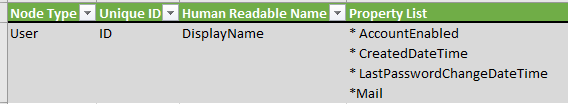
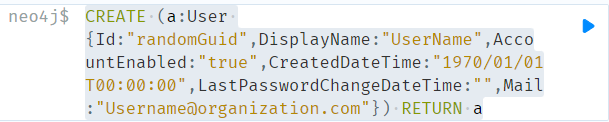
In this lab, we will manually create some objects in Neo4j, modeling a basic CA policy configuration from start to finish with an environment.

# Procedure

## Loading Neo4J

1. If Neo4j Desktop is not already running on your machine, launch the application now.
2. In the left side navigation bar, open the “Projects” View.
3. From this view, select **New** to create a new, blank database project.
4. In the dropdown, select **Create project**
5. By default, this will create a new Project named “Project”. To add a database, select the **Add** button.
6. In the drop down, select **Local DBMS**
7. Create a password for the database, then select **Create**.
8. ****Next, start the database by hovering over “Graph DBMS” and selecting **Start.**
9. When it is complete, select **Open** to connect to the graph database instance.

## Creating Nodes

1. In Lab Two, we identified nodes and edges as well as their properties we’d like to include in the graph. Now it’s time to create them in Neo4j.
2. To create a node, we’ll need to get a user to insert first. Run this command in PowerShell to gather the information of a single user.  
   $User = (Get-MgUser)[0]; $User | Format-List
3. Next, go to the node/edge spreadsheet to identify the critical properties  
   
4. Given the information from steps two and three, the create statement looks like this  
   CREATE (a:User {Id:"randomGuid",DisplayName:"UserName",AccountEnabled:"true",CreatedDateTime:"1970/01/01T00:00:00",LastPasswordChangeDateTime:"",Mail:"Username@organization.com"}) RETURN a
5. Insert the code in the neo4j prompt. 
6. The output should look like this if successful.  
   Graphical user interface, application

   Description automatically generated
7. Now repeat steps two through six for groups, applications, roles, and conditional access policies. When this is complete, you should have a graph database with one of each node type including user, group, role, application, and conditional access policy.
8. Run the code below to show all the nodes.  
   MATCH (a) RETURN (a)
9. The output of the code from line 8 should be like the following.  
   Chart, bubble chart

   Description automatically generated

## Creating Edges

1. For this lab, the theoretical edges connect all five nodes in the following manner:

(u:User)-[m:MemberOf]->(g:Group)<-[a:Assigned]-(r:Role)<-[e:Enforces]-(c:ConditionalAccessPolicy)-[p:Protects]->(s:Application)

1. Next, making sure to select all the appropriate nodes before making relationships. In this case, it’s straight forward with the following query:  
   MATCH (u:User), (g:Group), (r:Role), (c:ConditionalAccessPolicy), (s:Application) RETURN u,g,r,c,s

Chart, bubble chart

Description automatically generated

1. With the appropriate nodes defined in the match query, we can actually take the edge syntax in step one and apply it as a create statement.

MATCH (u:User), (g:Group), (r:Role), (c:ConditionalAccessPolicy), (s:Application)

CREATE (u)-[:MemberOf]->(g)<-[:Assigned]-(r)<-[:Enforces]-(c)-[:Protects]->(s)

1. Now rerun the following query to see the entire graph.

MATCH (a) RETURN (a)

1. The results should look like the following.  
   