Graphing M365: Lab Two

# Objective

Students will spend their time modeling out their database, focusing on Conditional Access Policies, Users, Groups, Roles, and Application. This will create a foundational core for expansion into other areas within Microsoft 365.

# Procedure

## Standing up the M365 Tenant

*Note: if you already have access to a sandbox environment, jump to setting up PowerShell*

1. Open a browser and navigate to [Developer Program | Microsoft 365 Dev Center](https://developer.microsoft.com/en-us/microsoft-365/dev-program)
2. Start the process by selecting **Join Now**
3. Logon with your M365 account.
4. Fill out the form information and select **Next**
5. Select the appropriate radio option for your objectives. Then select, **Next.**
6. Select **Microsoft Graph** and then click on **Save** to complete the initial registration.
7. When asked, select **Instant Sandbox** and then click on **Next** to instantiate your new tenant.  
   Graphical user interface, text, application, email

   Description automatically generated
8. Create an Admin Credential and select **Continue.**
9. Add your phone number for Multi-Factor Authentication to the Sandbox.
10. Your tenant should be created with your new credentials ready to administer the environment.
11. Sign on to your test tenant at <https://portal.azure.com/>

## Configuring PowerShell

*Note: if you are running on Windows 10 or 11, skip this section*

*Note: PowerShell 5.1 and 7 will both work acceptably for this course.*

For Non-Windows systems, please install the latest version of PowerShell for your operating system following the appropriate instructions here:  
  
[GitHub - PowerShell/PowerShell: PowerShell for every system!](https://github.com/PowerShell/PowerShell)

## Modeling Data

The bulk of the lab is held in this section. Here you will grab different types of data from the standard sandbox tenant and start modeling in documentation the types of nodes and edges we care about in M365.

1. Open a PowerShell prompt
2. Connect to Microsoft Graph with the sandbox administration account  
   Connect-MgGraph
3. Switch graph to the beta profile  
   Select-MgProfile -Name Beta

Using the beta API gives us access to all the fully released features and many of the unreleased features as well. While the beta is not meant for production use, Microsoft uses it within many of their portals exposed through the administrative portal(s) and is safe to use for reporting systems.

### Modeling a User Account

1. With the PowerShell graph connected, run the following command to get all the users in the tenant.  
   $Users = Get-MgUser
2. To look at a user data profile, select one and show all it’s properties using the following command:  
   $Users[0] | Format-List
3. In the output, there are many properties. Some of these properties are useful for security purposes and others are not. As with all models we will build, we’ll need to find a way to organize this information for later. Out of the gate we need to identify the following pieces of information:  
   1. Unique ID from Microsoft
   2. Human Readable (recognizable) name
   3. [optional] Other properties of interest

If we look at the user account, the following fields are interesting:

1. Unique Identifier: ID
2. Human Readable Name: DisplayName
3. Other Properties:
   1. AccountEnabled
   2. CreatedDateTime
   3. LastPasswordChangeDateTime
   4. Mail

We can now revise the original get users command to extract just this information going forward:  
  
$Users = Get-MgUser | Select-Object ID,DisplayName, AccountEnabled,CreatedDateTime,LastPasswordChangeTime,Mail

1. Next, open the document “Models.xlsx” in the lab packet.  
     
   *If you don’t have software to handle spreadsheets installed on your machine, office.com offers free access to excel which will be sufficient for this lab.*
2. Open the **Nodes** spreadsheet and fill out the information according to the image below.  
   Graphical user interface, text, application

   Description automatically generated
3. Now complete the same steps of gathering data for the following node types.  
   1. Groups (Get-MgGroup)
   2. Roles (Get-MgRoleManagementDirectoryRoleDefinition)
   3. Enterprise Applications (Get-MgServicePrincipal)
   4. Conditional Access Policy (Get-MgIdentityConditionalAccessPolicy)  
        
      *Don’t forget to save as you make progress…*
4. With some of the preliminary models complete for nodes, now we transition to edges. To start, let’s get the membership edge for users and groups. With the structure of the graph, first we get a group, then we pass that group ID into a secondary command to retrieve all the members.  
     
   $Group = (Get-MgGroup)[0]  
   $GroupMembers = Get-MgGroupMember -GroupId $Group.Id  
     
   This returns a list of the User IDs which are members of the group and some selected properties about the user.
5. Reviewing the members, we can start to define our first edge model. At minimum for an edge, we need the following information:  
   1. Relationship Type
   2. Source (owns the relationship)
   3. Target
   4. [optional] Other properties of interest

For the group we can define the fields as follows:

1. Relationship: member
2. Source: GroupId
3. Target: UserID
4. Navigating to the **Edges** spreadsheet, enter in this information according to the image below:  
   A picture containing timeline

   Description automatically generated
5. Complete mapping out additional edges between:  
   1. Groups to Users/Groups (Get-MgGroupMember)
   2. Roles to Users/Groups (Get-MgRoleManagementDirectoryRoleAssignment)
   3. Conditional Access Policy (Get-MgIdentityConditionalAccessPolicy)
      1. Users
      2. Groups
      3. Roles
      4. Enterprise Applications