EDU Auto-Deploy Solution Installation Guide

# Prerequisites

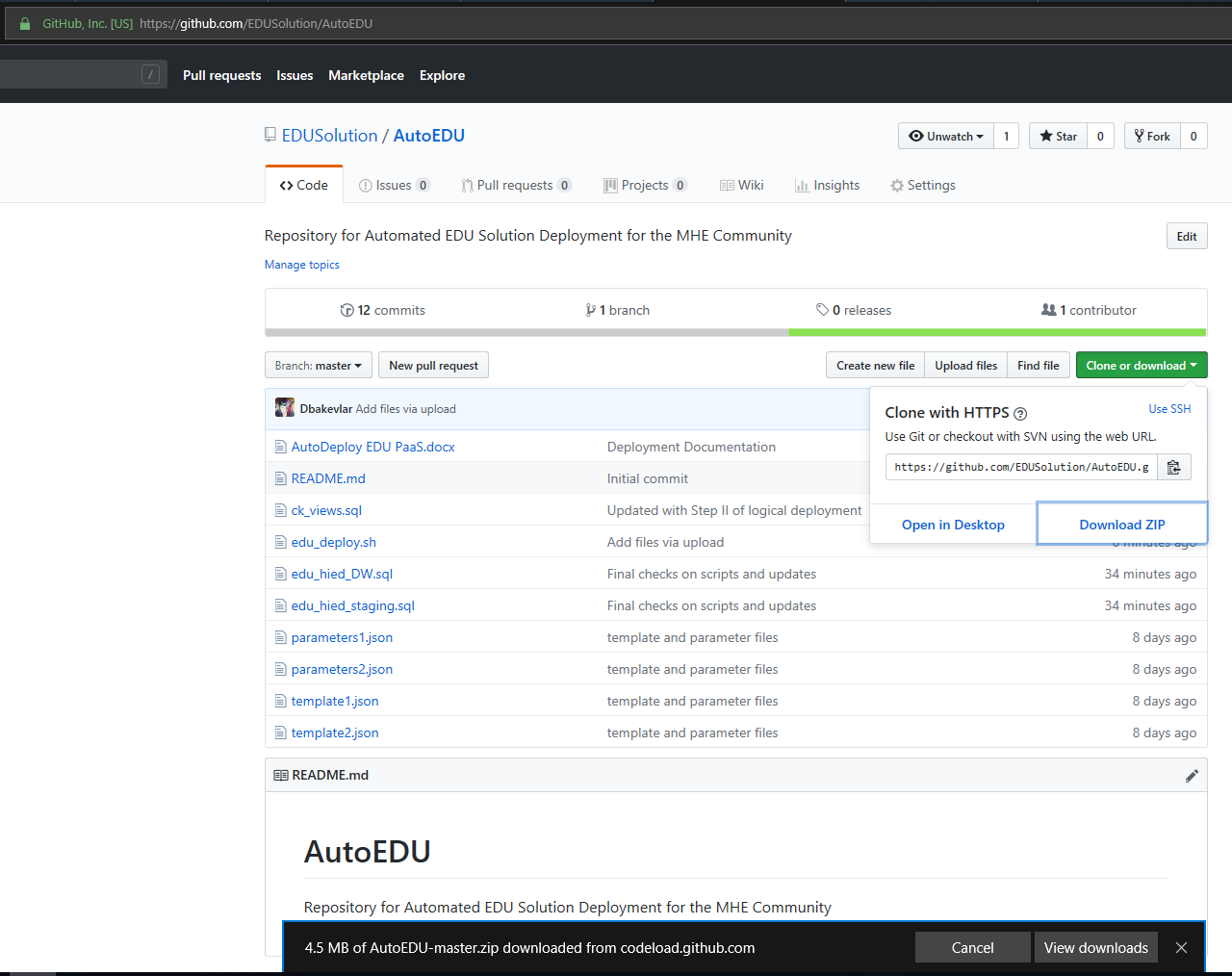
**Set up Azure account on your own as provided by your IT department-**

This auto deployment assumes you’re using the (Azure Active Directory) configured for an account and if not already using Azure, there are [credits offered to use with Azure](https://azure.microsoft.com/en-us/free/) that will support most of this exercise, (as long as you are conservative with data/resources).

**Have installed on your local workstation the LATEST SQL Server Management Studio, (SSMS) and SQL Server Developer Tools, (SSDT).**

## Deploy the Environment to Azure

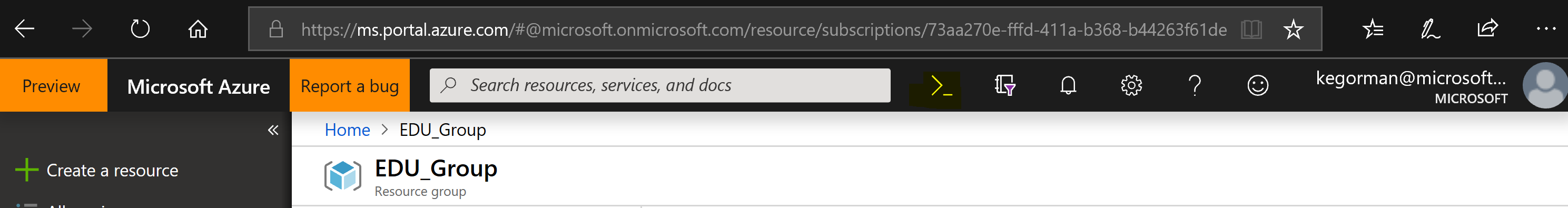
1. Log into [Github](https://github.com/EDUSolution/AutoEDU) and download the EDU Solution Scripts



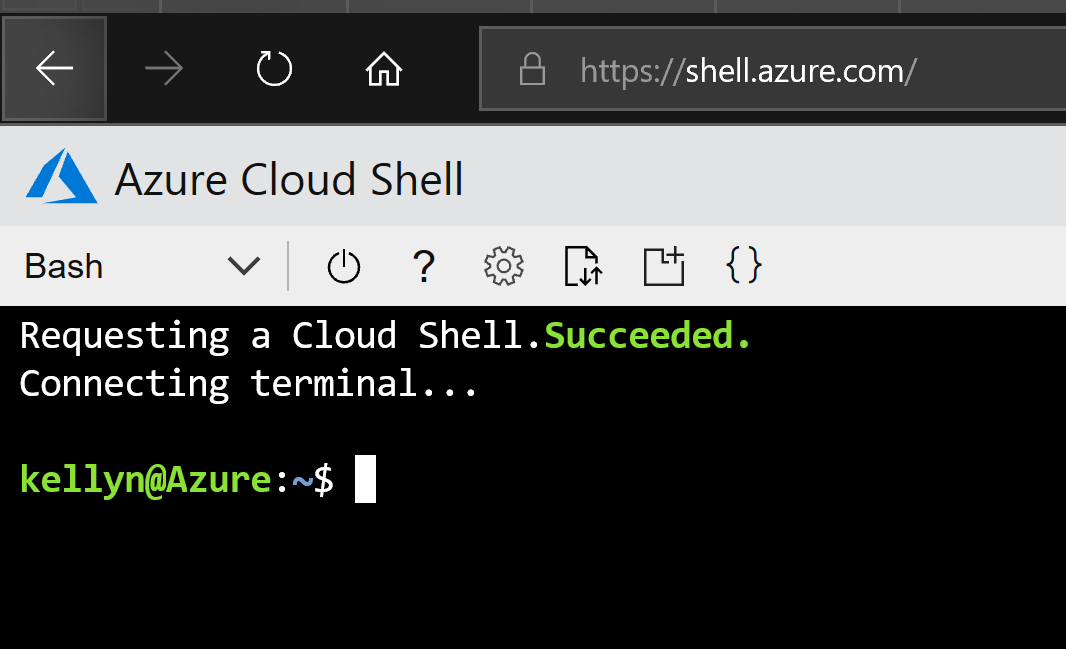
1. Unzip the files in a separate directory on your workstation.

## Log Into the Azure Cloud Shell

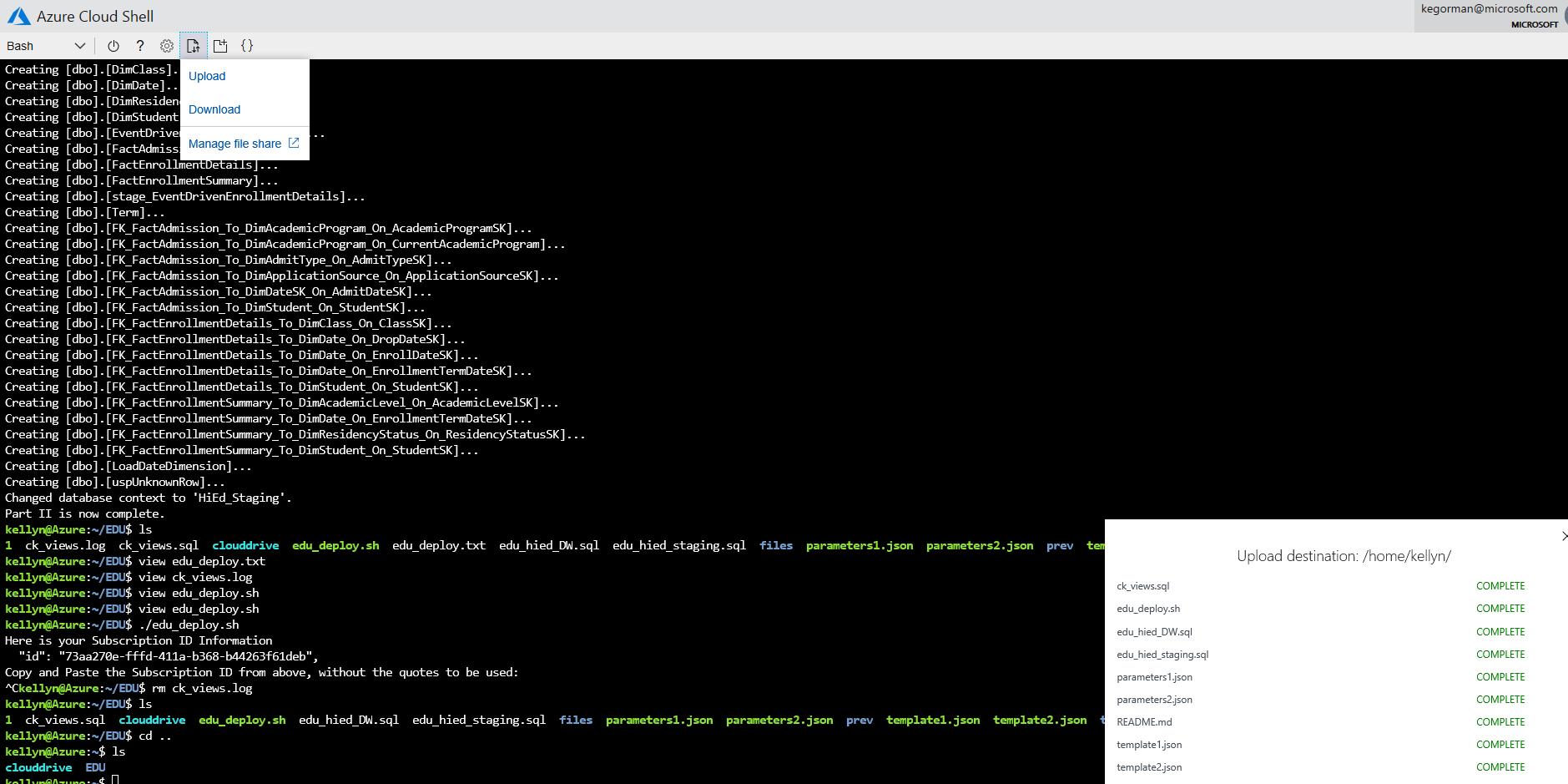
1. Log into the [Azure Portal](https://ms.portal.azure.com) and access the Azure Cloud Shell from the title bar:



1. Or Log into the Azure Cloud Shell directly from [here](https://shell.azure.com/).
2. Choose the Azure Active Directory Login to use for the deployment and click on it to then access the shell if you aren’t already logged in.



1. Upload the files from Github to your shell home directory, highlighted in the square above.
   1. Choose Upload from the drop down.
   2. Browse to the location you extracted the files to.
   3. Extract all the files from the extract except the instructions, (docx) to the cloud shell:
      1. edu\_deploy.sh
      2. template1.json
      3. template2.json
      4. parameters1.json
      5. parameters2.json
      6. ck\_views.sql
      7. hied\_data.zip, WHICH CONTAINS:
         1. hied\_dw.sql
         2. hied\_staging\_enroll.sql
         3. hied\_staging\_data.sql



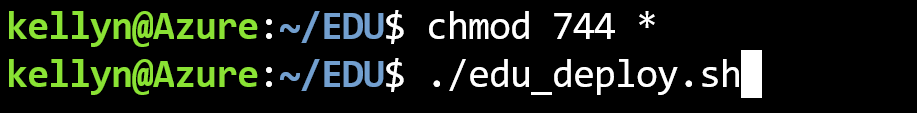
* 1. Type in the following command in the shell console to ensure files are executable:

unzip hied\_data.zip

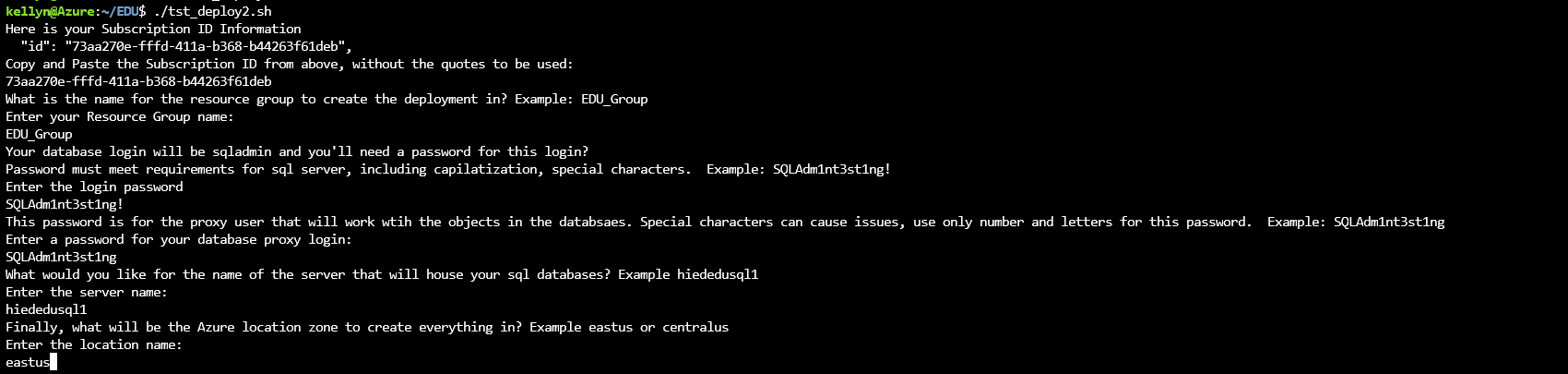
chmod 744 \*

## Execute the Script

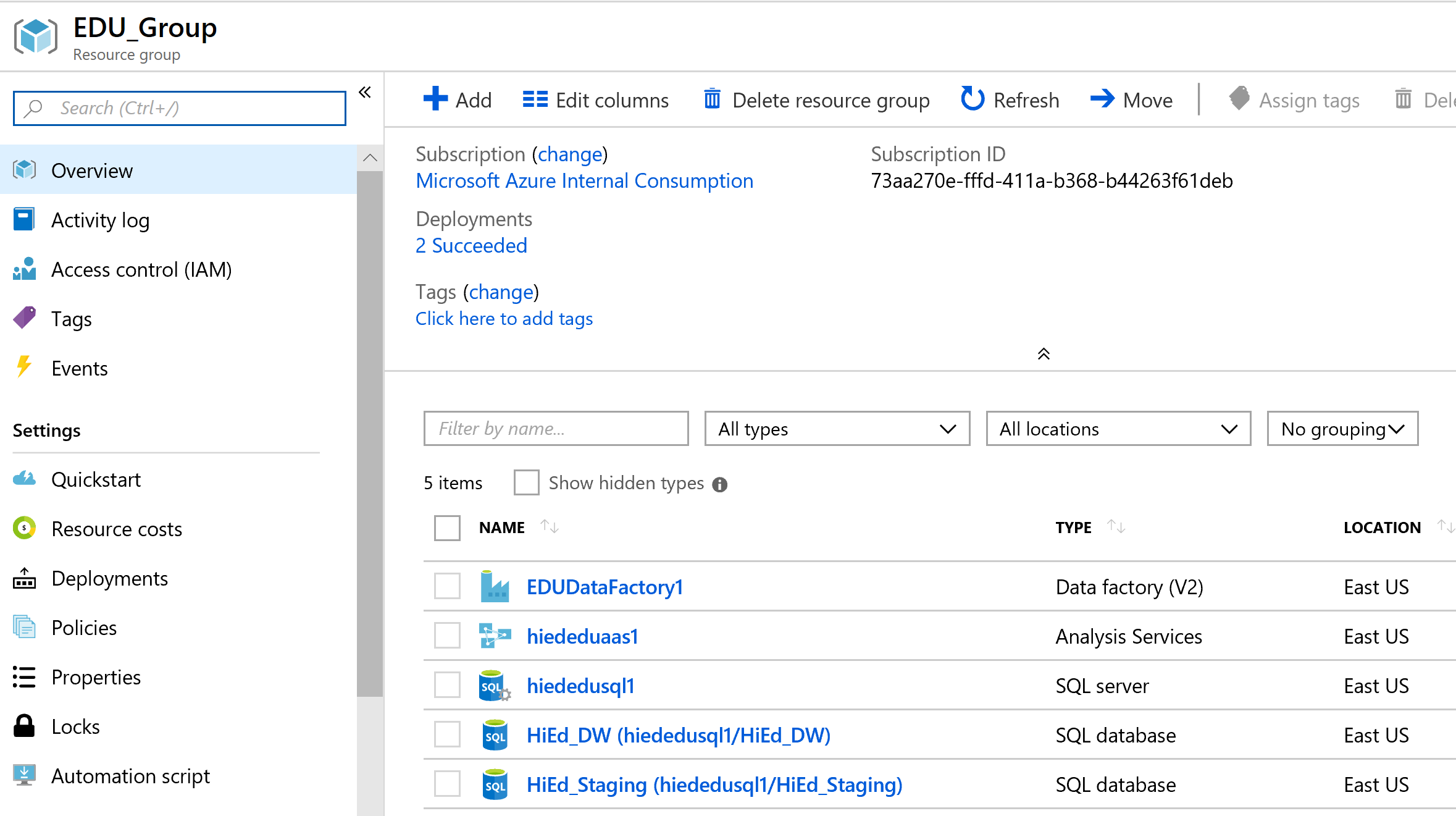
1. Run the script by typing in the following command:
   1. ./edu\_deploy.sh



1. The script will take you through the following:
   1. Display subscription IDs connected to your account and you’ll choose one to use as the default for the deployment. This can be copies and pasted by highlighting the text, right click, left click copy, right click, left click past.
   2. It will ask you questions about what you wish to name your Resource Group, SQL Server, the password for the administrator account for the database server, along with password for the proxy database user, the zone, (eastus is the default for the Azure Data Factory and Analysis Server, so I would recommend using eastus for better performance.)



* 1. It will then deploy until it begins the Analysis Services step. It will then request an administrator for this resource. The most logical choice is the same Azure account that you used to log into the shell. An example is displayed to offer assistance.
  2. The final two resources are deployed and no log results should be returned in red, (signals an error) and you will see the following when completed.



**Important -** If there are any failures in the execution, the script will exit and a clean build is recommended. Its simple to clean up and redeploy, please see the end of this document for the Troubleshooting section and “If a Failure Occurs”

Once the physical resources are deployed, the logical objects inside the database are deployed

1. Creates the HigherEDProxyUser login in the SQL Server with the password you entered in the questions section of the script deploy
2. Creates the Data Warehouse objects, support structures and HigherEdProxyUser in the Warehouse database.
3. Creates the Staging database objects, views and HigherEdProxyUser in the Staging database.
4. Does a check and output of the view objects to verify they were all created.
5. Ouputs a log file, named edu\_deploy.txt that shows the information and passwords for the environment for you to retain.

You will know the process is complete when the prompt returns, “Part II is completed.”

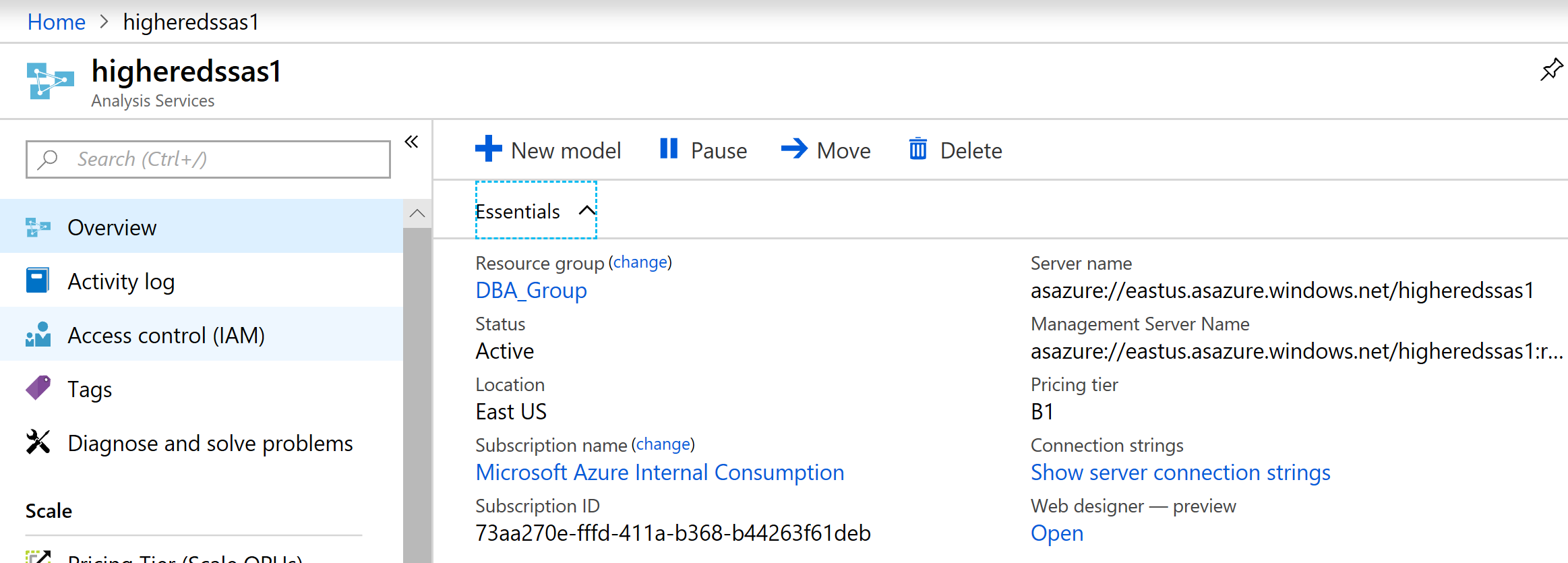
You can review the information log file by typing in the following at the prompt:

view edu\_deploy.txt

To exit from the log file, click the [Esc] key, then “:q” [enter] key

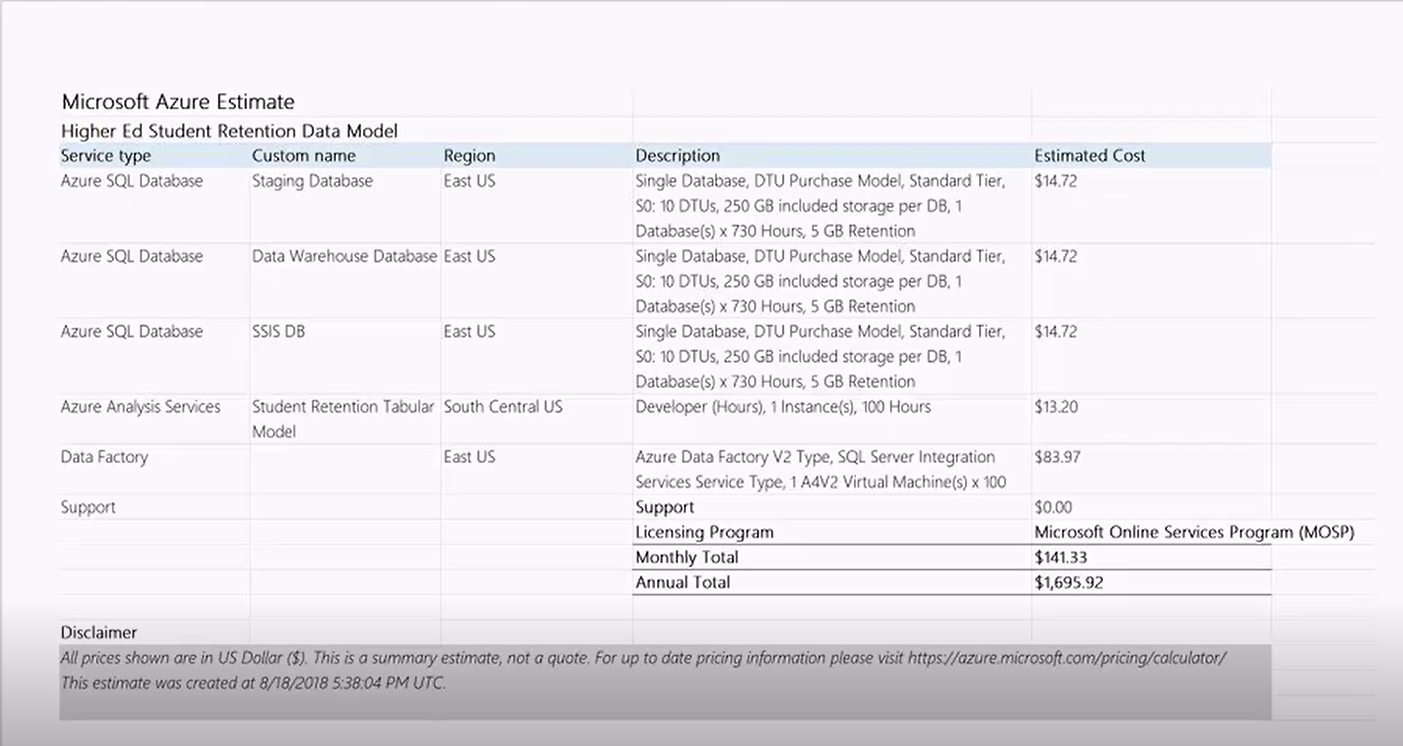
# IMPORTANT - If the Analysis Services server is not going to be actively utilized, make sure to select Pause to pause the compute resources allocated to the server and minimize charge.

1. In **Azure Portal**, click on **All Resources**
2. Double click on the link to your Analysis Server in the list of resources
3. At the top, you’ll see **Pause**. Click on it and confirm you wish to pause the SSAS chosen:



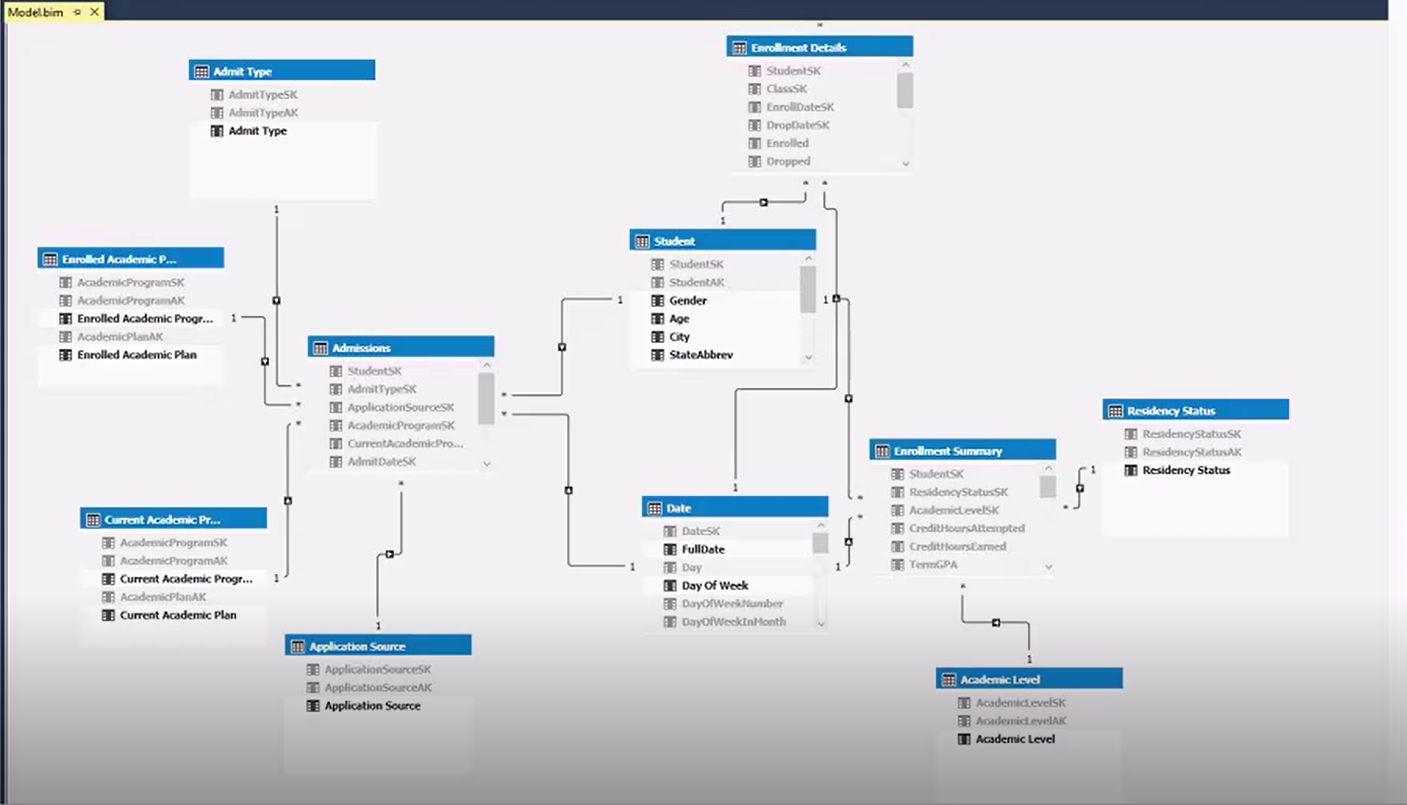
Once complete, refresh the portal and verify that SSAS displays **Paused**.

Understanding the cost of the deployment model in Azure:



Consider the usage of the solution to make your decision. A lower tier will provide you with a deployment example that may not be acceptable for user deployment. The estimated time to develop or test the environment is 100 hours.

This doesn’t include any Power BI Pro cost. A requirement for Power BI Pro is expected as part of this solution for each power user that will be creating and working with the solution.



**IMPORTANT:** Select the Analysis Services server to open the overview blade. If the Analysis Services server is not going to be actively utilized, make sure to select **Pause** to pause the compute resources allocated to the server and minimize charges:

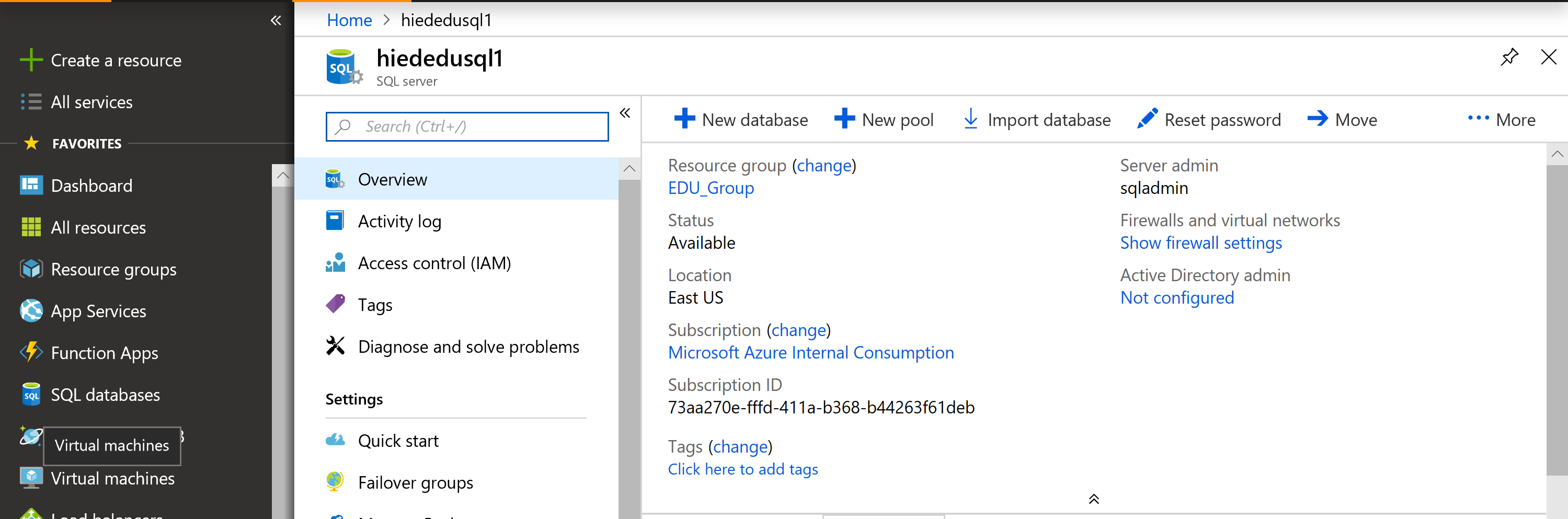
# Part II- Prepare Azure for Access

To access your Azure SQL Database via the SQL Server Management Studio, you’ll need to perform a few steps. We’ll learn how to update the firewall rules that were created by the script. The roadmap for the automation will dynamically configure this step, but doesn’t do it currently and knowing how to configure access is important to know. As you access the Azure resources from different locations or different PCs, a rule will have to be configured or updated. The following section will explain how to do this.

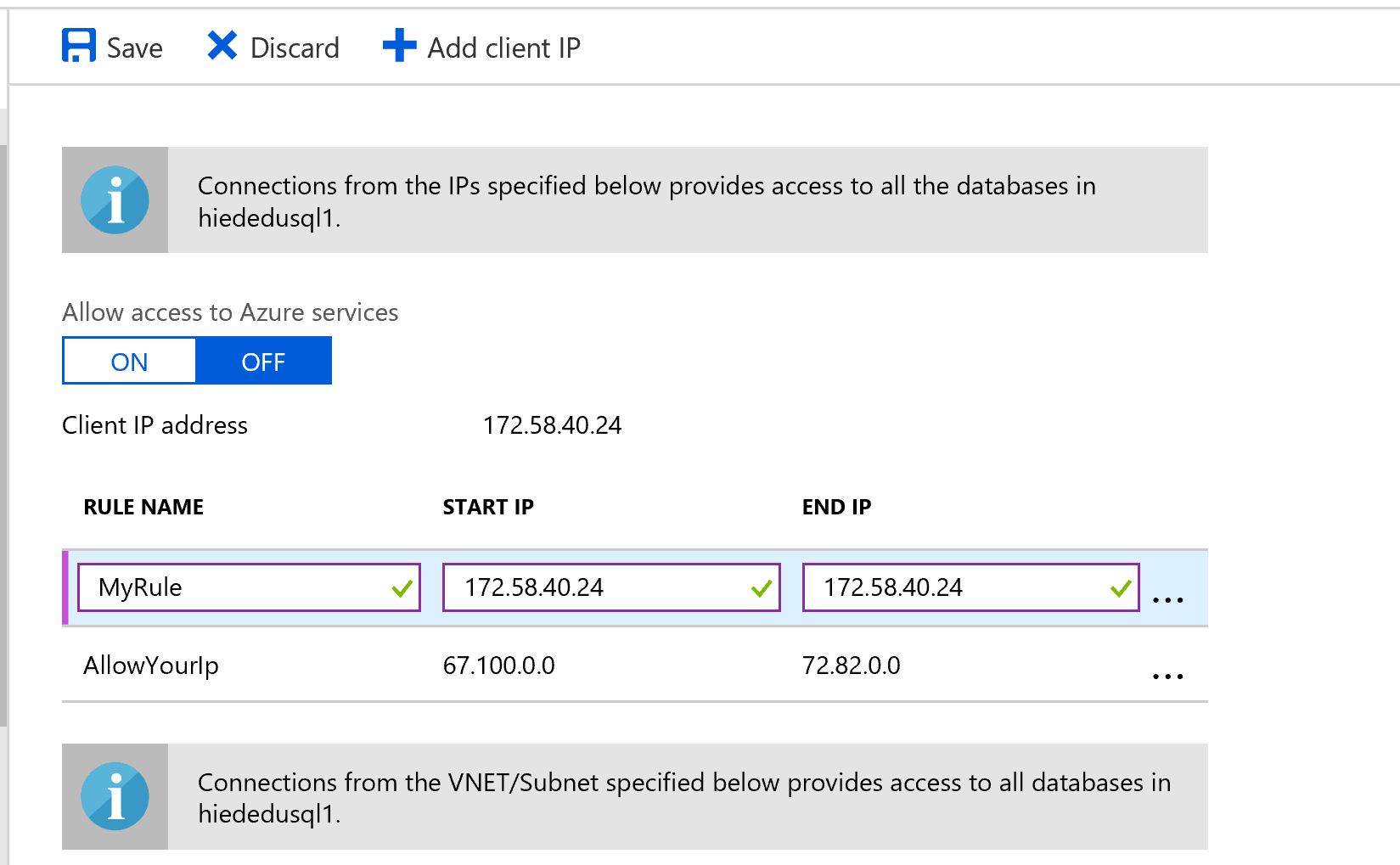
The next step is to collect the information about the Azure host name and how to connect with Management Studio. Although it can feel quite alien to connect to the cloud, its rather simple once you get the hang of it.

### Configuring the Firewall Rule

1. Go to the Azure Portal.
2. Click on Resource Groups and choose the EDU\_Group.
3. Click on your SQL Server, hiededusql1.
4. Click on Show Firewall Settings



1. You’ll note the auto-deployed rule **AllowYourIp**, which displays the example. Using the “**Client IP Address**”, which displays your IP address that is accessing the portal at that moment, create a new rule, filling in the name and then the **Start IP** and **End IP** as your **Client IP Address** displayed.



Once filled in, click **Save.**

### Your Host Information in Azure

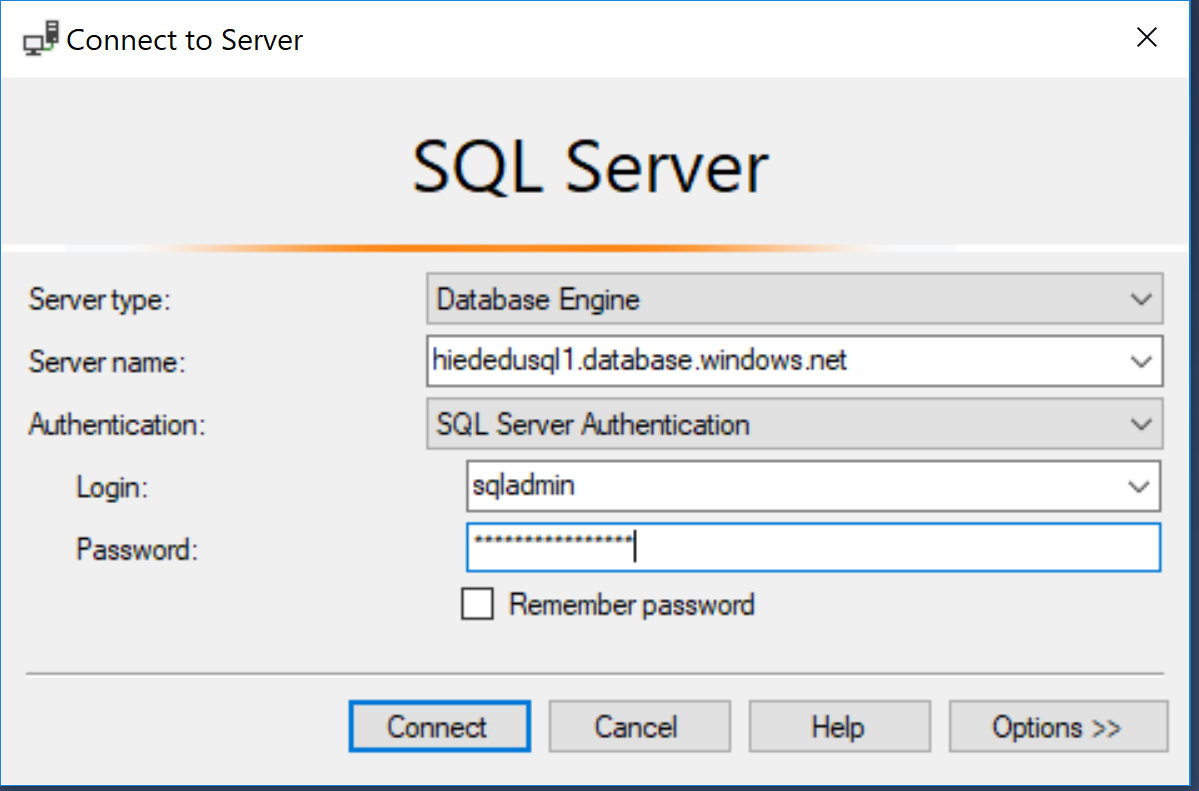
Your database server name for SSMS access is <sql server name>.database.windows.net

You can locate this in the Azure portal by:

1. Click on SQL Databases
2. Click on the database you wish to connect to:



The server name is clearly displayed in the Overview.

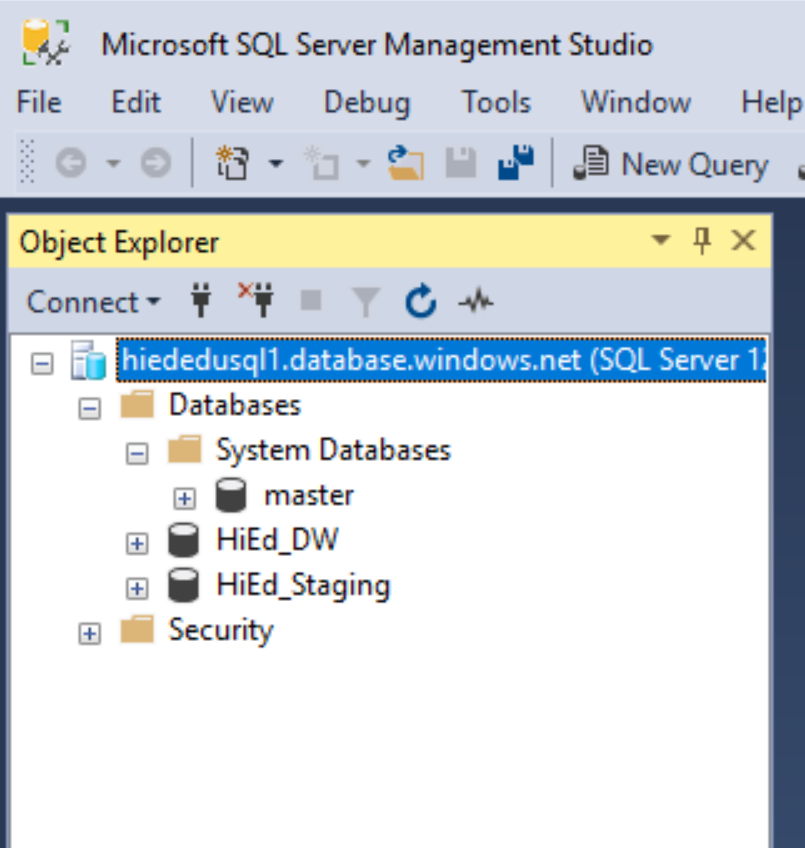


If, for some reason, your firewall rule hasn’t been set up, it will prompt you to set it up dynamically.



Choose to Sign in with your AAD and click OK once it prompts for the firewall rule.

You should then be logged in successfully with SQL Server Management Studio, (SSMS).



As Visual Studio 2017 becomes more mainstream, there are templates for Power BI that have been left out from Visual Studio 2015. Upon each upgrade, these could be missing and impact the ability to open up projects in the solution explorer. These packages take no time to install and can be found here:

<https://marketplace.visualstudio.com/items?itemName=ProBITools.MicrosoftReportProjectsforVisualStudio>

<https://marketplace.visualstudio.com/items?itemName=ProBITools.MicrosoftAnalysisServicesModelingProjects>

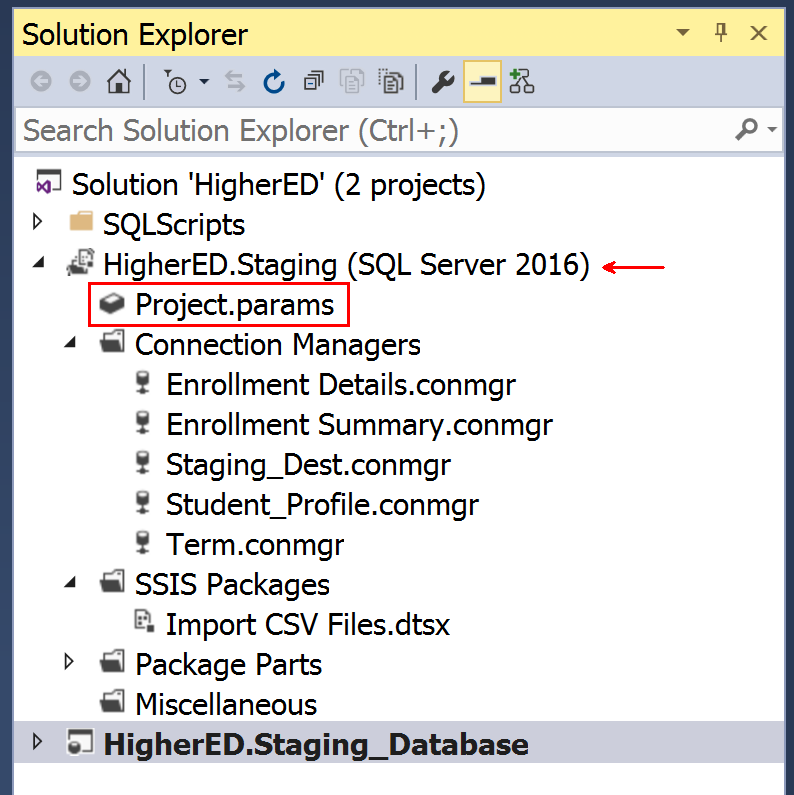
If you receive an error about compatibility when opening in VS or SSMS 2017, download and and install these packages to complete the missing pieces of the newer installation. They do work.

## Part III- Deploy the SSIS Packages

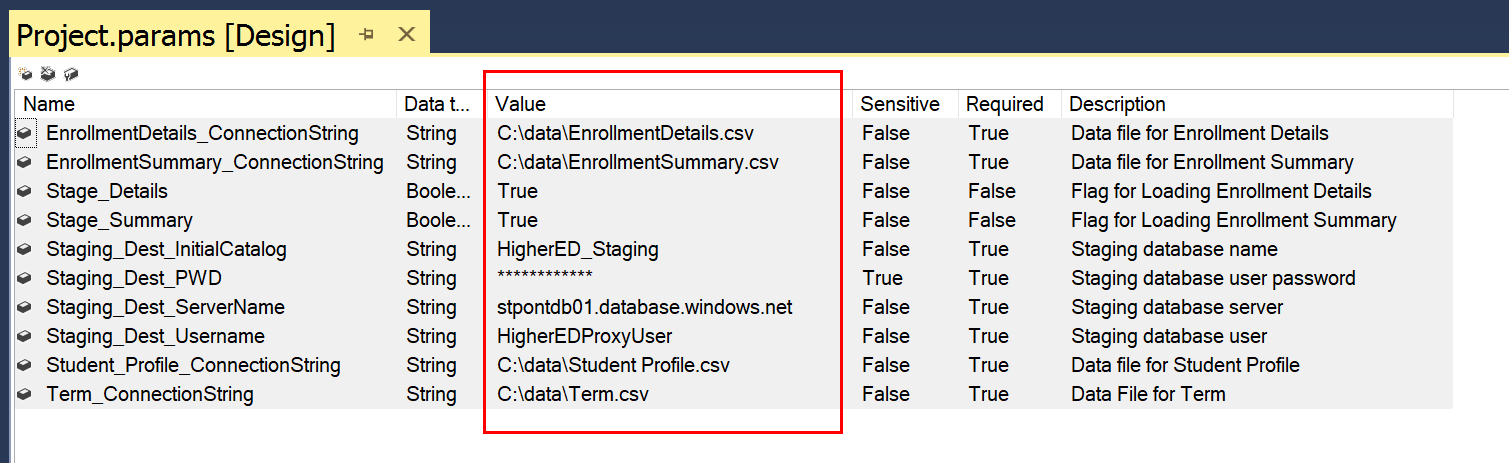
In the Solution Explorer or Go to Visual Studio, (VS) and open up the staging project and navigate to the data folder from the Data Validation folder. There is only one solution folder but four CSV files will be located in the directory. These are our data files we’ll load into the staging database.

This is the location you would place your own data if you wanted to use that instead of the example files. We need to update the parameter files first to work with your environment source files:

1. Expand the **HigherED.Staging** Solution
2. Double-click **Projects.params** in the HigherED.Staging Project.



1. Update the file locations for: **EnrollmentDetails\_ConnectionString**, **EnrollmentSummary \_ConnectionString**, **Student\_Profile \_ConnectionString** and **Term \_ConnectionString**
2. Change the **Staging\_Dest\_ServerName** to your server name.
3. The **Staging\_Dest\_Username** and **Staging\_Dest\_PWD** are pre-populated with the correct values for the proxy account created when provisioning the staging database. The values for the Proxy user are:
   1. **Username:** HigherEDProxyUser
   2. **Password:** MakeThisStrong123!



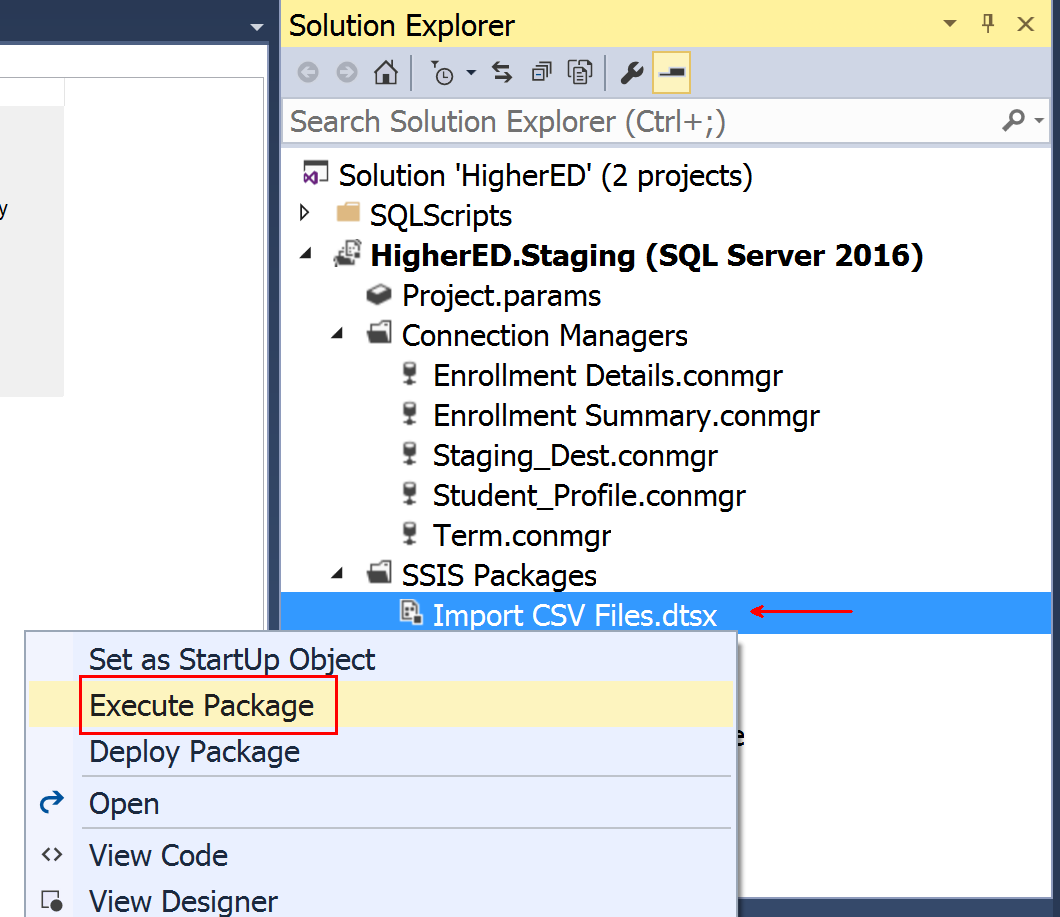
Once finished, click on SAVE to save the changes to the parameters.

Note: If you’re doing this from Visual Studio instead of SSMS, the steps to deploy are very similar:

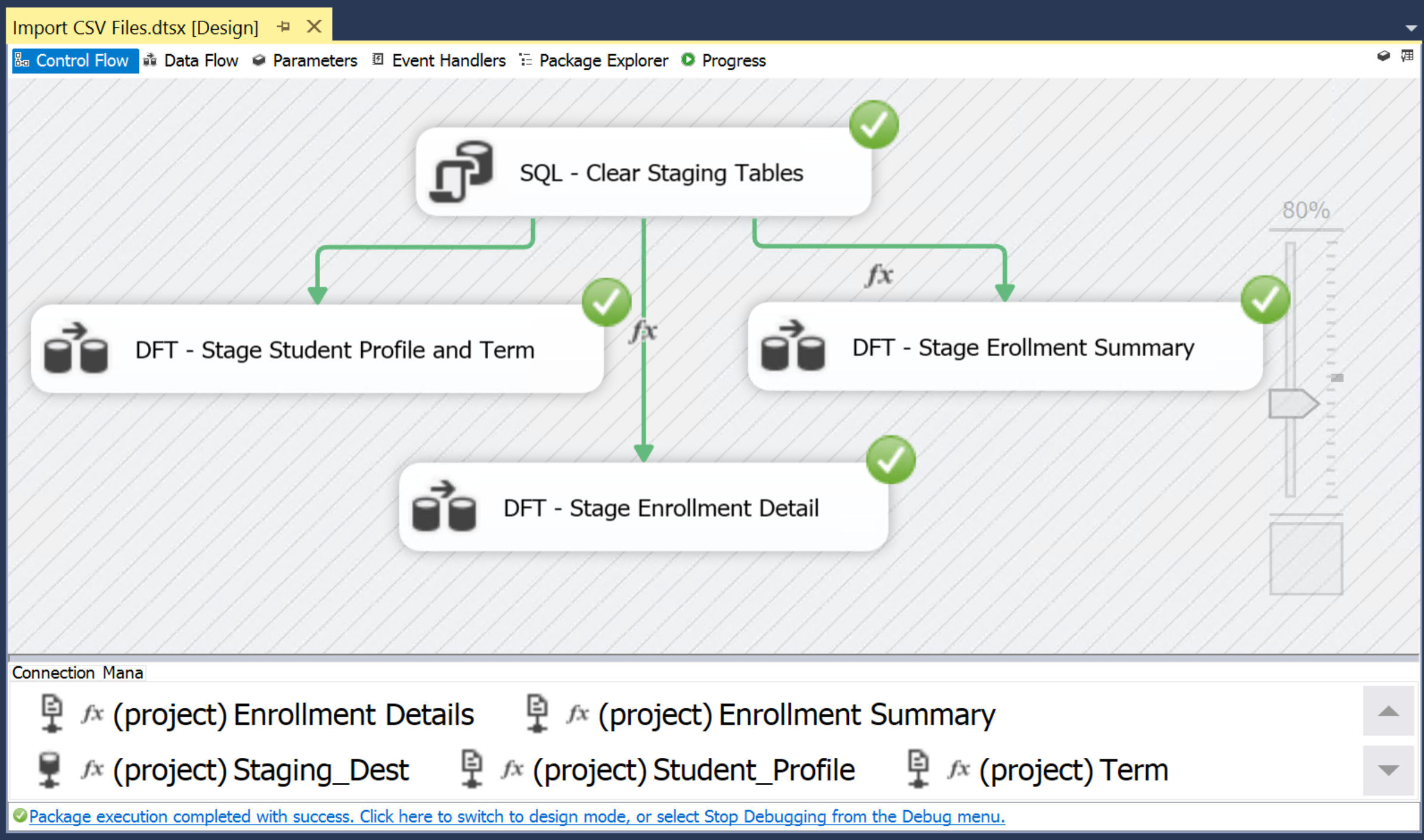
A screenshot of a cell phone

Description generated with very high confidence

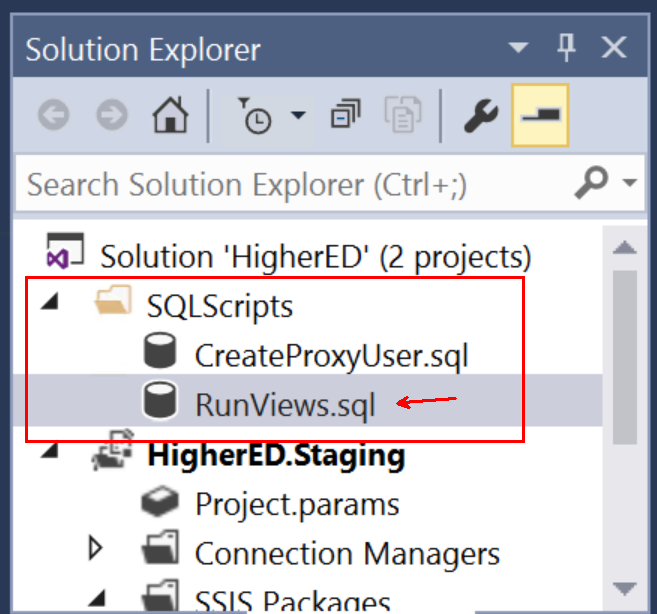
1. If using Solution Explorer in Management Studio
   * Right-click the **Import CSV Files.dtsx** SSIS Package and select **Execute Package** from the context menu.
2. If using Visual Studio:
   * The Import CSV Files will already be displayed, Right on it and left click on Execute Package.



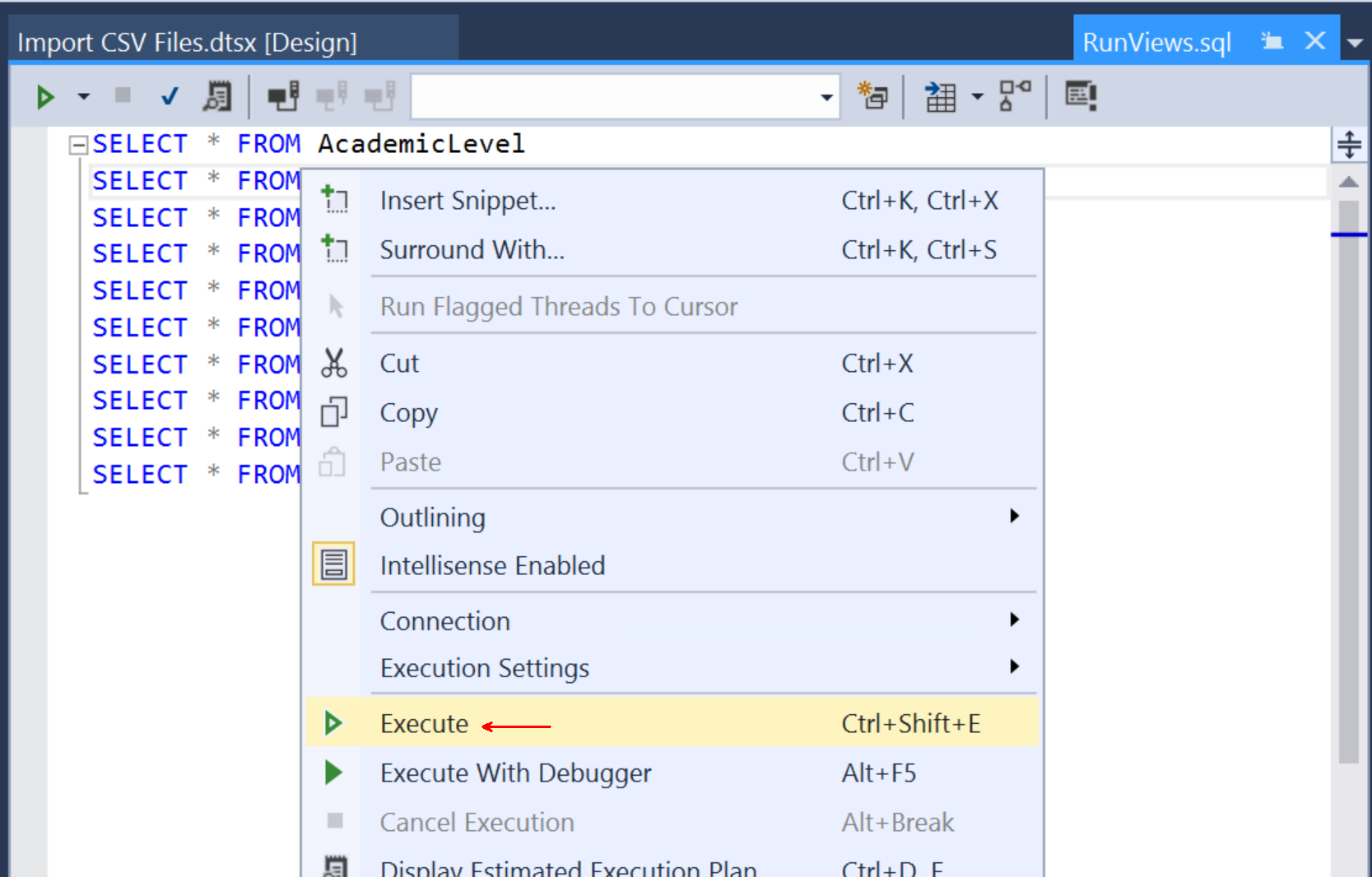
* The DTSX file you’re executing now is a data flow, which is a simple task to move data from one place to another. In this case, moves the data from the CSV files to the staging database.
* The package should run successfully and load all of the staging tables. If the package does not run successfully, debugging may be required.
  + First thing to check, are there any red “X’s” at the corner of any of the steps in the data flow? Check the parameters for any incorrect settings.



1. Expand the **SQLScripts** folder in the Solution Explorer. Double click on the **RunViews.sql** query to open the SQL Editor.



1. Right click in the SQL Editor and select **Execute.**

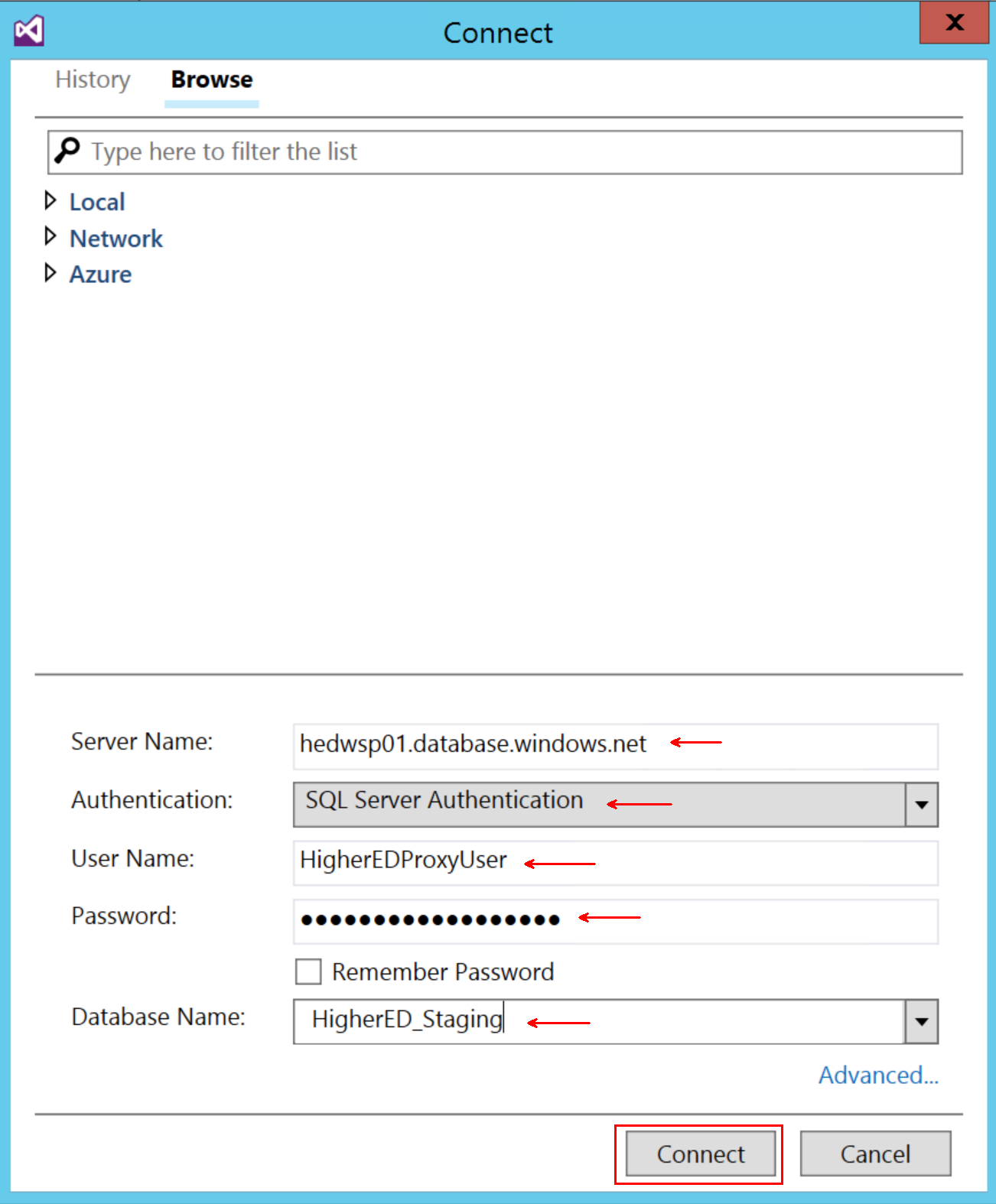
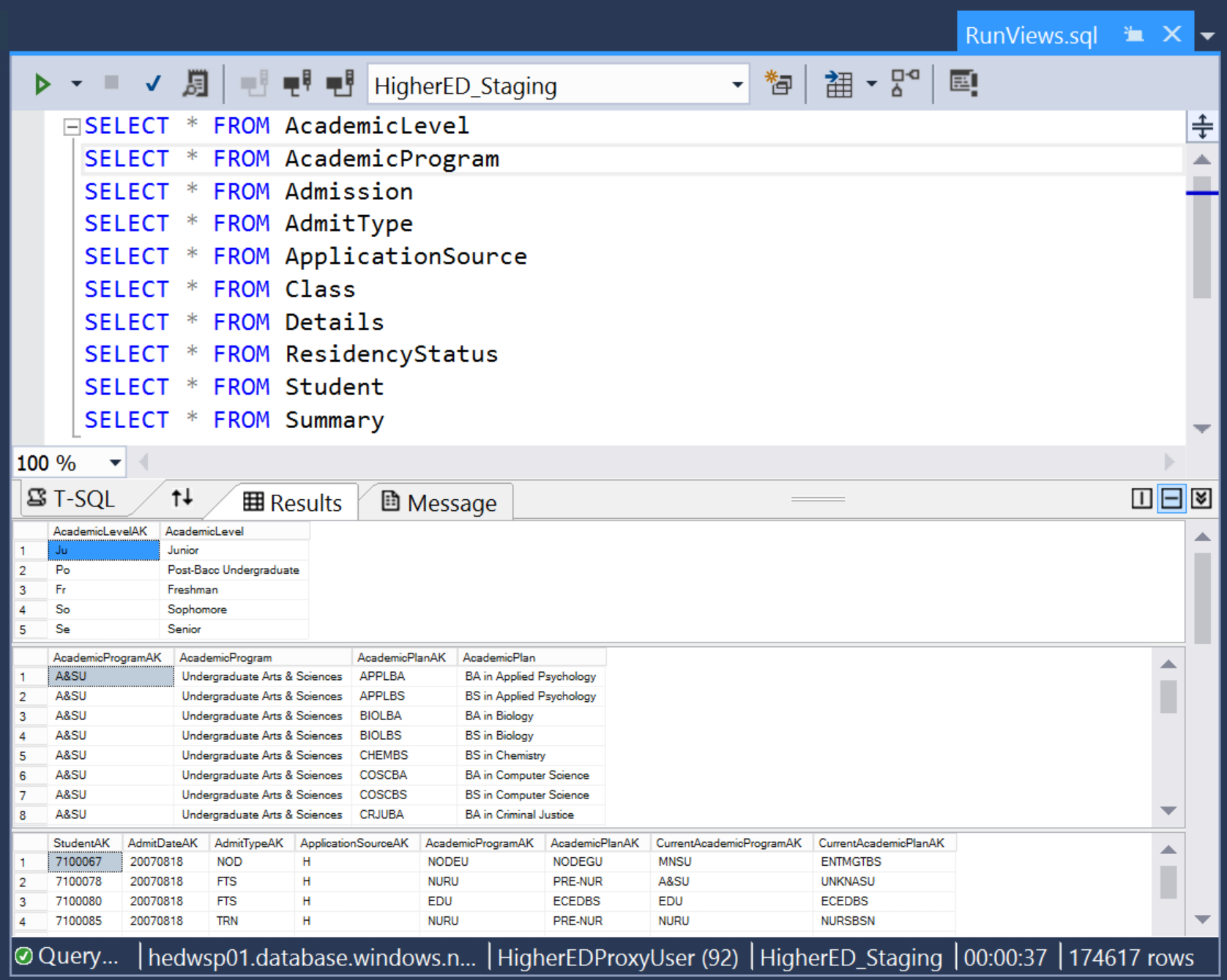


1. **Visual Studio:** 
   * Click File -> Open -> File and browse to Data Validations/scripts/RunViews.sql and choose to open it.
   * Click on **Connect**
   * Click on **Local** and use the drop down to locate your SQL Server used for the project and click on the server name.
   * Verify the **authentication** has rights to the tables, (using Windows authentication in the example, but you will connect to Azure.)
   * Choose the **HigherEd\_Staging** database from the dropdown and click **Connect**.

A screenshot of a social media post

Description generated with very high confidence

1. Enter the name of the Azure SQL Database server, select **SQL Server Authentication**, enter **HigherEDProxyUser** as the user name, **MakeThisStrong123**! for the password and **HigherED\_Staging** for the Database Name. Click **Connect** to run the script which will query the staging database tables.
2. Execute the script. If any tables are empty are errors appear you should reach out to you Microsoft Contact to help resolve.

You have now successfully deployed the SSIS Packages to the Staging Database.

# Part IV- Deploy the Datawarehouse to Azure

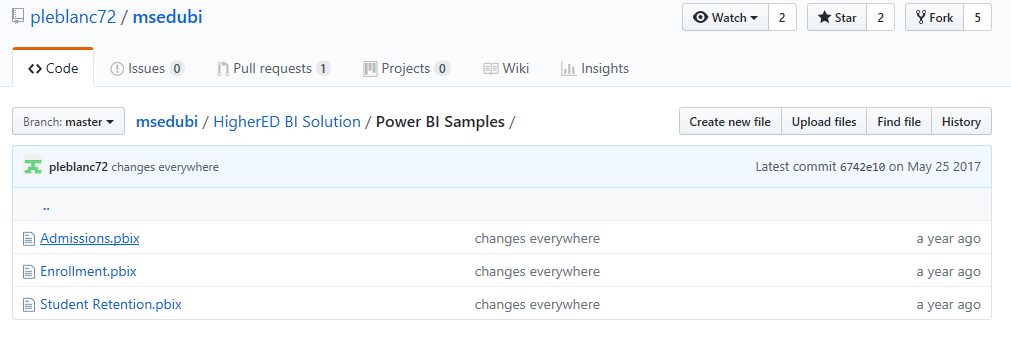
1. Sign into the Azure Portal:

<http://portal.azure.com/>

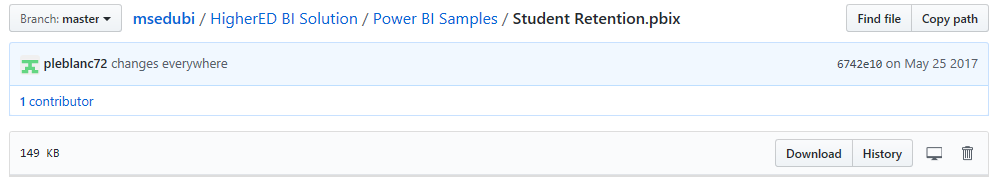
# Part V- Deploying and Connecting the Power BI Desktop Reports

In the Github repository, under msedubi/HigherEd BI Solution/Power BI Samples, there are three PBIX files:

* Admissions.pbix
* Enrollment.pbix
* Student Retention.pbix



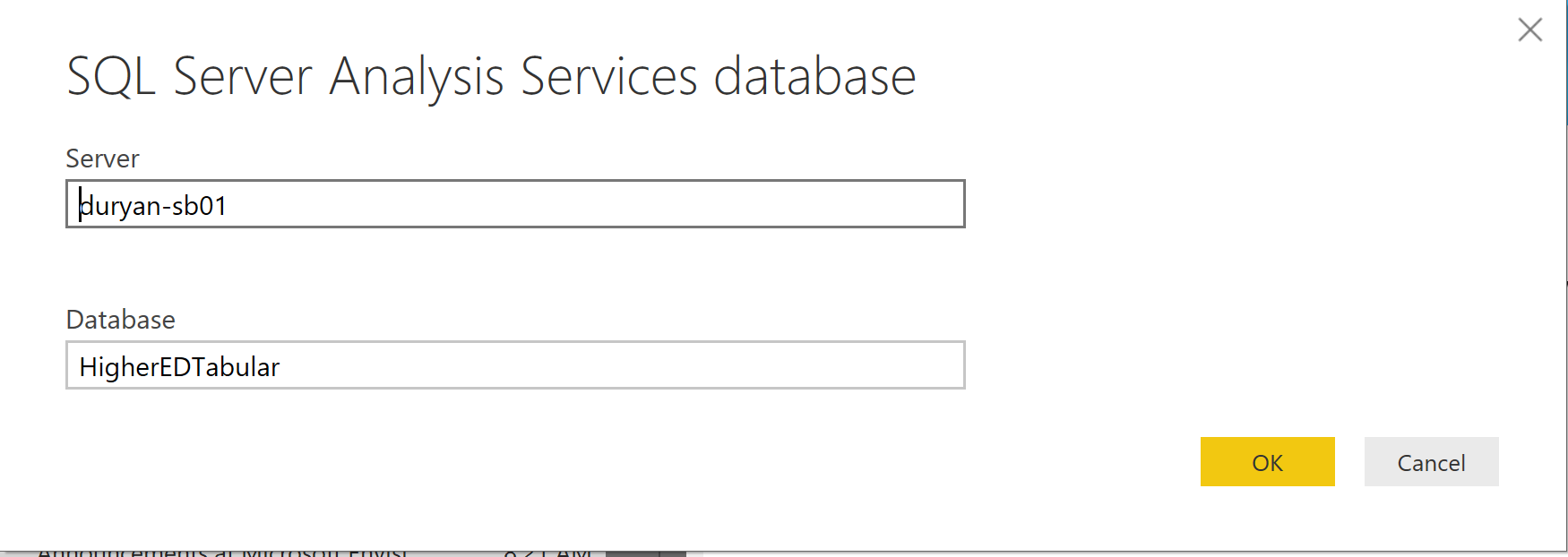
Download all three of these files from the repository to your local workstation by double-clicking on each of them and then clicking the Download button in the next window.



Once you’ve added each of these to your desktop, open one of them with PowerBI Desktop.

As an example, we’ll use the Student Retention.pbix file.

An error should display about the Analysis Server connection, click on Edit and the following screen will be shown:

Update the Server to your SSAS server and the database to the Higher Ed Data Warehouse you deployed.

Before you proceed forward, ensure that your VMs, your ADF are turned on/unpaused.

For the next steps, you’ll need the following:

* Workstation download of github files.
* Visual Studio 2015, (or higher)

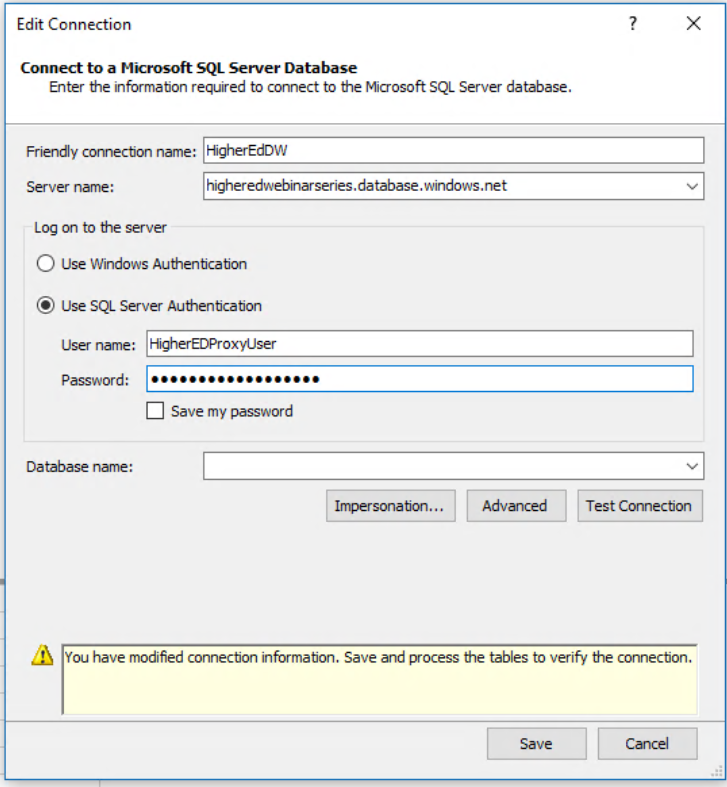
1. In the SSAS Tabular Folder, open the HigherEd.sln file in Visual Studio
2. When asked what workspace, SSAS will need to know where you are working on the file. Unlike PBI, SSAS can have Integrated Workspace Mode True or False.

\*\*If the data model is extensively large, choose FALSE)

1. Double click on the .bim file that is part of the SLN

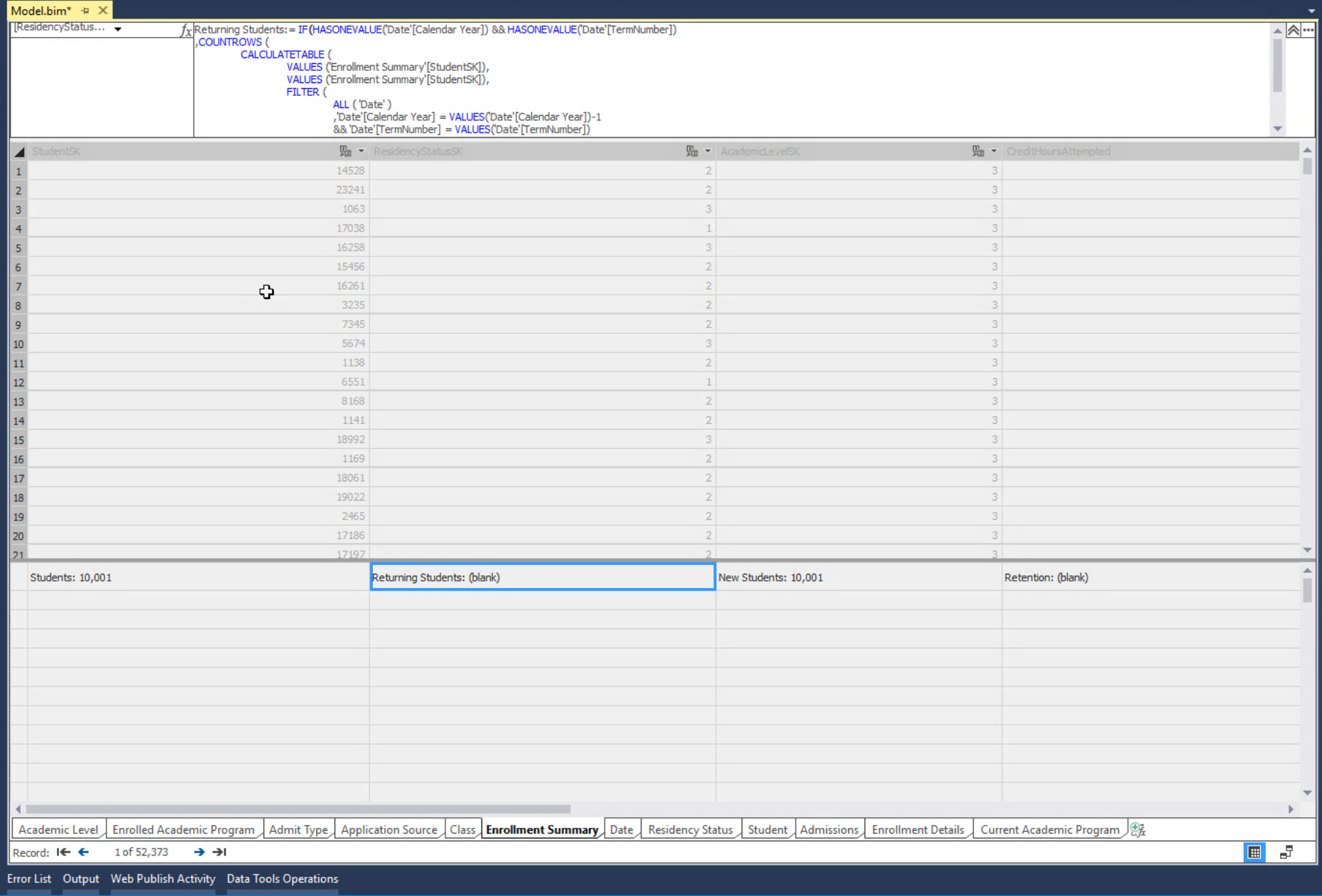
## Connect and Load Data

1. Click on left side Object Explorer and to the HigherEdDW Data Source.

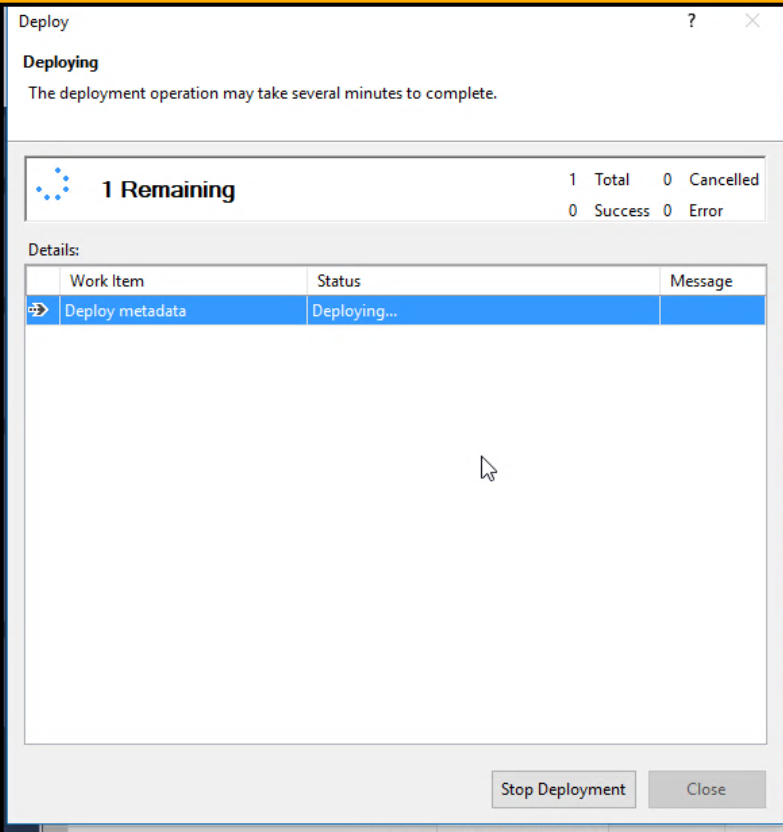


1. Update the information to your database, including database name, user name and password.
2. Model 🡪 Process 🡪 Load Data
3. If you’re loaded your own data, it may take some time. If you’re used the same data, it will load data rather quickly.
4. Parse through each table and review the data that’s been loaded into the data model.

Although the interface isn’t as user friendly as Power BI, there is a relationship and grid view of the data:



1. Right click on projext in the Solution Explorer and click on Properties.
2. Update the Server name configuration to the SSAS server you created as part of your project in the Azure portal.
3. If you would like a different name for the database than HigherEdTabular, rename it. If you deploy a second time with a new name, it will create a second database.
4. Click OK.
5. Right click on the project and click on Deploy.
6. Log into your SSAS Server with the logins you created as part of that environment and the model will deploy:

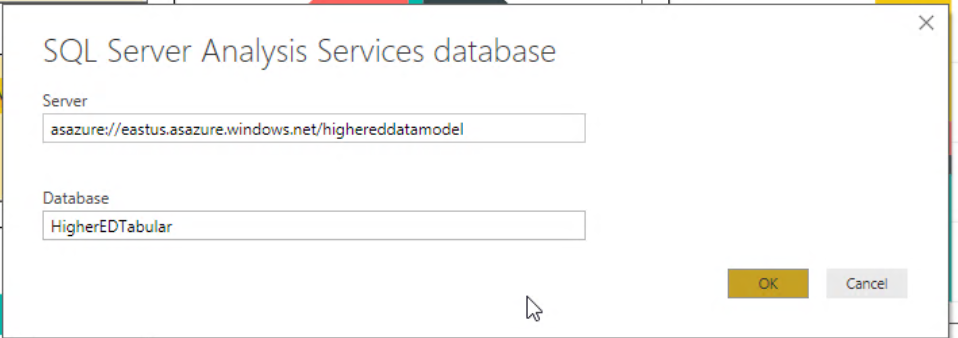


## View and Load the Tabular Service Database in AS

1. Log into Analysis serves with SSMS and use SQL Authentication of the sqladmin and password.
2. Once logged in, the database and objects are readily available, but no data is present. The model hasn’t been processed yet.
3. Right click on HigherEdTabular, (or what you’ve named your database) database and click on Process and choose Process Full, not Default.
4. Once successful, the process will complete and you can view data present.

## Load Power BI Reports

1. Go to Power BI Sample reports.
2. Open the Admissions.pbix file from the examples present.
3. Expect that failures will happen, as data sources will need to be updated and a popup will occur to correct the sources.
4. Update the server to the Azure Server instance for SSAS. (Servername/data model)
5. Enter the name of the SSAS database name for the database name. Click OK



Deploy Power BI Reports

1. In the Power BI Portal
2. Add a new app workspace, (use newest workspace for new features!)
3. Name the Workspace and click Save unless you want to change to premium capacity)

Go back to Desktop

1. Click on Publish for the report, choose the new workspace you just created and click on Deploy.
2. Perform the same steps for the Enrollment.pbix and Student Registration.pbix

Congratulations, you’ve now published the report to the web with the reports and the data model.

# Troubleshooting

## If A Failure Occurs in the Script or to Remove the Installation

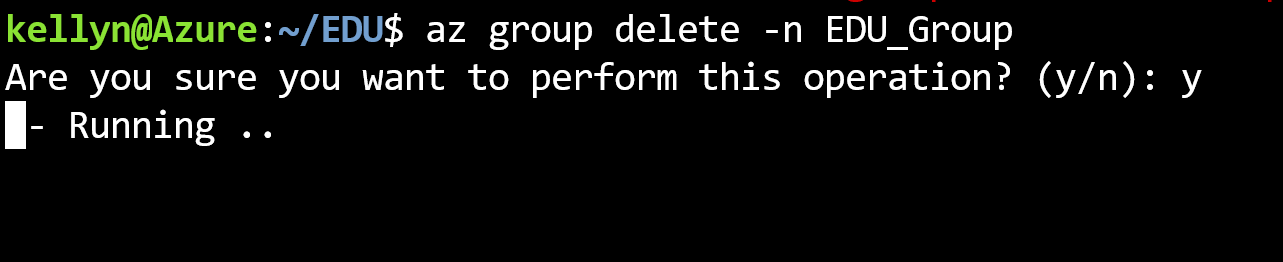
1. Check the log and request assistance from the TSP team at Microsoft if there is a failure.
2. Clean up to restart the process is easy.
   1. From the **Azure Portal**.
      1. Click on Resource Groups.
      2. Double click on the Resource Group you attempted to deploy your EDU Solution to, (in the example, it’s EDU\_Group.)
      3. Inside the blade for the group, choose Delete Resource Group from the top.
      4. Confirm you wish to delete the resource group by typing in the name of the group and click OK.
   2. From the **Cloud Shell**
      1. Type in the following:

az group delete -n <group name>

* + 1. Example:

az group delete -n EDU\_Group

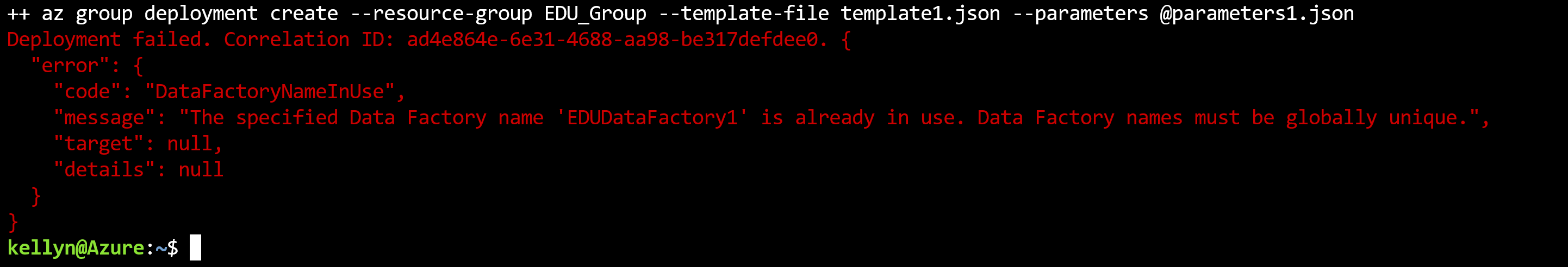
* + 1. Confirm that you wish to delete the group and hit enter.



Important: Depending on the number of resources that were deployed, this process can take upwards to 15 minutes to complete.

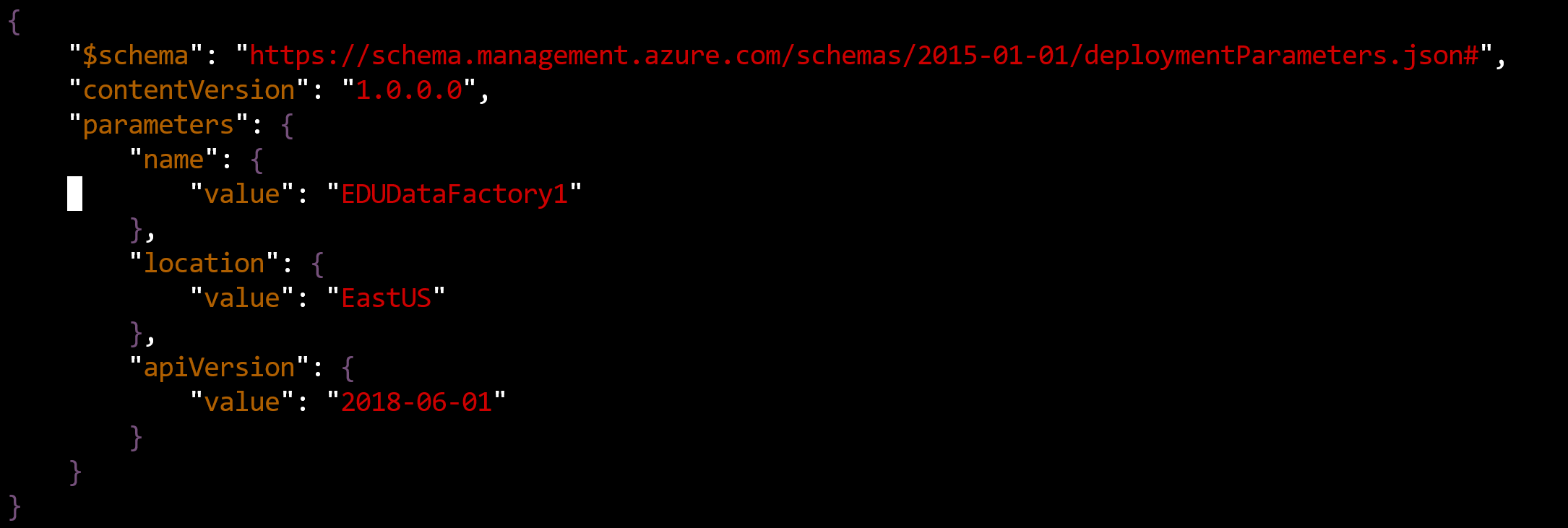
## Data Factory Creation Fails Due to Name

Upon drop of Resource Group on previous creation, a new run of the EDU Solution auto deploy fails on the ADF step with the following error:



There is a “shadow” entry for the previous data factory name, although its actually been deleted. To address this, update the json parameters file for this resource and update the name to a unique, new name.

>vi parameters1.json



1. Use the “J” key to go down to the line that says “EDUDataFactory1”
2. Use the “L” key to move over to the number 1 in the Data Factory name.
3. Click on the “R” for replace and type “2”- this will replace the 1 with a 2, updating the name of the Data Factory that will be create.
4. Save the file by typing [Esc] :wq [Enter]
5. Drop the Resource Group from the Azure Portal for EDU\_Group for the failed run.
6. Re-execute the script from to build the solution.

This should resolve the issue until Azure addresses this intermittent bug.