**Uploading On-Premise data to Azure Blob Storage using SSIS**

**Author: Supriya Pande**

**Introduction**

In this article, I am going to describe how we can upload the on-premise data to Azure Blob Storage using SQL Server Integration Services (SSIS). SSIS is a great ETL tool known for moving the tasks from various data sources and for applying transformations. On the other hand, Azure Storage is a Cloud based storage solution which can store data of varied types. This article will give you a brief overview of Azure Storage account and how-to setup Azure Storage account. Further, I will walk you through the process of uploading the data successfully to the Azure Blob Storage.

**What is Azure Cloud Storage?**

Many time in your work you might come across a scenario where you need the data to be available across the globe, you might want your data to be replicated at various locations so that there is no chance of data loss and it will be easily accessible in any part of the world. This durability and scalability is provided by Microsoft’s Cloud Solution called Azure. For Storage purposes, Azure provides its very efficient service called Azure Storage. As defined in Microsoft’s documentation “Azure Storage is Microsoft's cloud storage solution for modern data storage scenarios.” It really made of 4 different types of storage such as Table Storage, Blob Storage, Queues Storage and Files Storage. All of these storages are the main components of Azure Storage account. Out of these storages we are going to use upload our data into blob storage. In the next section, I am going to provide you the list of required tools and packages we will need to successfully load data into Azure.

**What is Azure Blob storage?**

Blob Storage is a highly efficient type of Storage and it has capability to store huge unstructured data. It can store any type of unstructured data such as MP3 audio files, images, pdfs and video files, audio files, pdfs and larger documents. Unstructured data may contain Dates, numbers and Blob storage provides great solution to store them all. Additionally, it provides scalability and geo-redundancy. In this article, I am going to use the capabilities of Azure SSIS pack to upload a file from my local environment over to the Blob storage.

**What are all the requirements:**

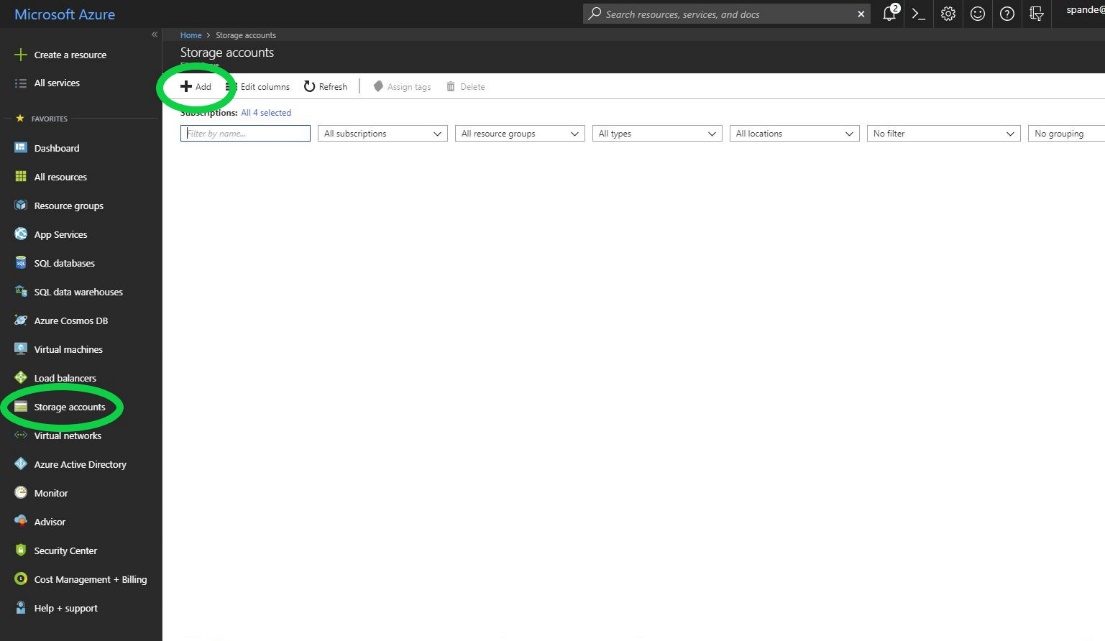
1. Azure Storage Account
2. Azure Storage Explorer
3. Azure SSIS Feature pack

**Environment Setup:**

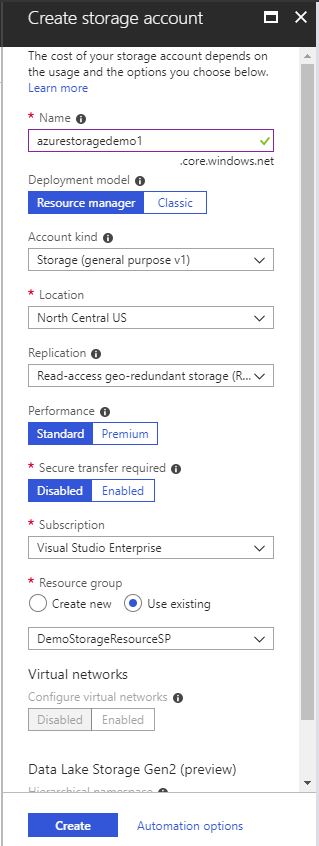
**Azure Storage Account:**

Let’s just get our environment setup. In the first place, I am going to setup an Azure Store account. To set this up, you will need a valid Azure Storage Subscription. Let me show you how to create a Storage account under your Azure Subscription.

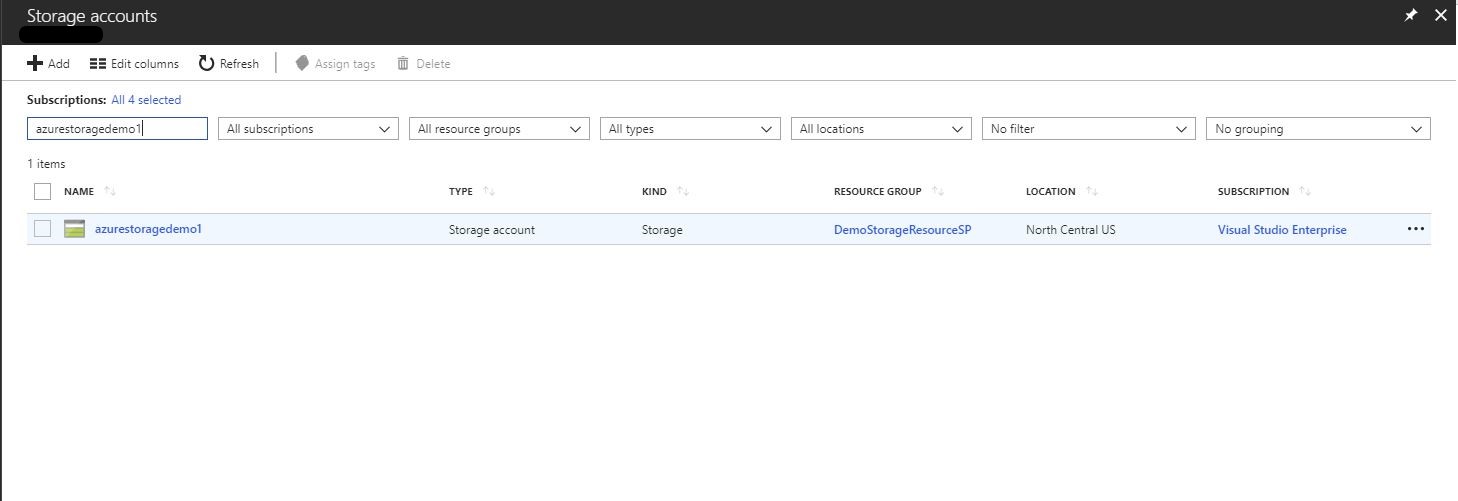
Once the login to [Azure Portal](https://portal.azure.com/), you will see all these services that Azure provides on the left panel of the page. You will the notice that there is a service named “Storage Account”.



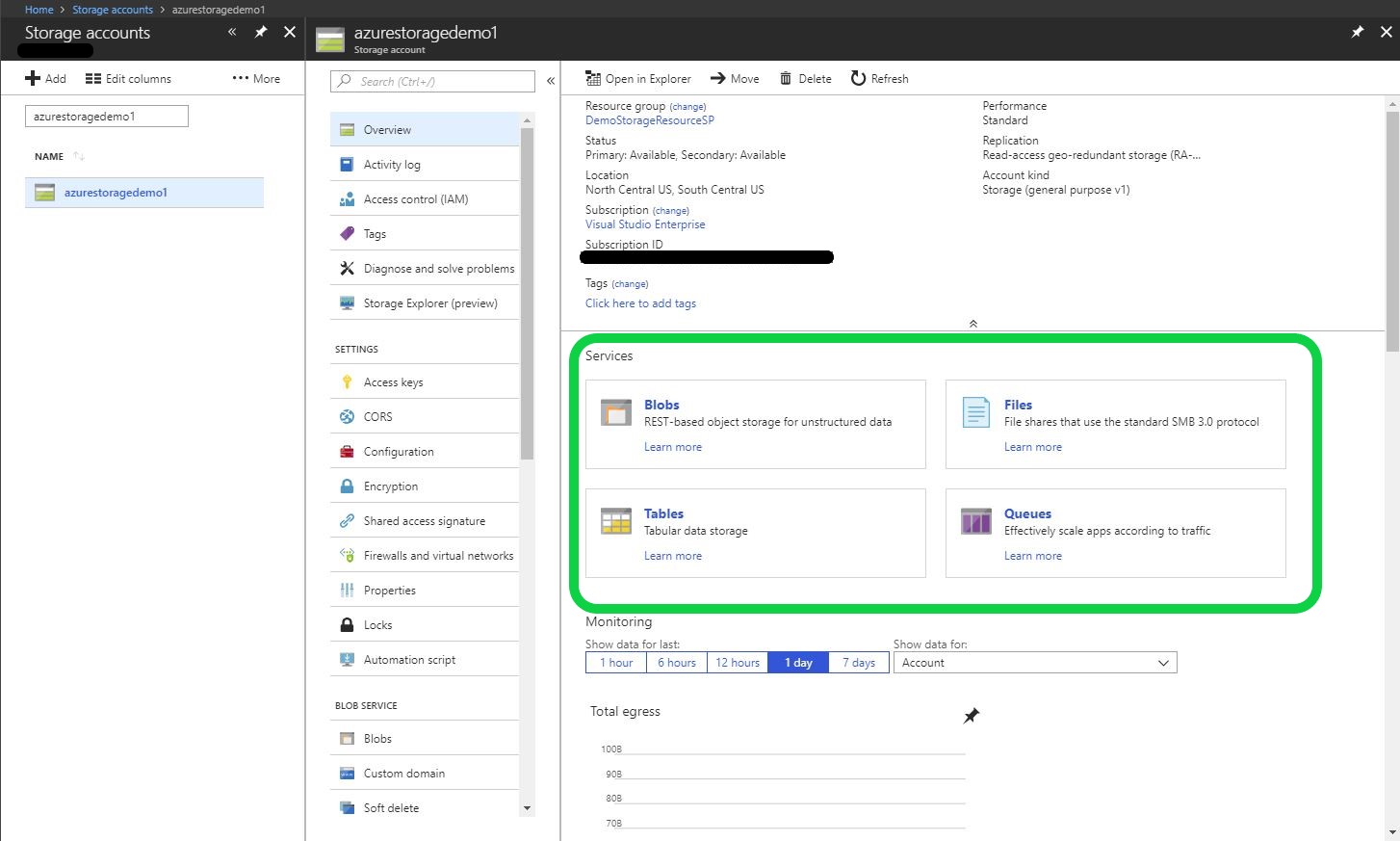
Once you click the Storage Account you will see all the list of storage accounts present in your subscription. You can further add an account by clicking the “Add” button and provide the storage account name. I have provided the storage account name “azurestoragedemo1” for this demo and I have selected the existing resource group for my account. You can also create a new Resource group if you want. For now, I have just selected default options to create Azure Storage. We can go over to the specifics of all the fields sometime later.



After you click Create. You will see that Storage Account named “azurestoragedemo1” is successfully created.

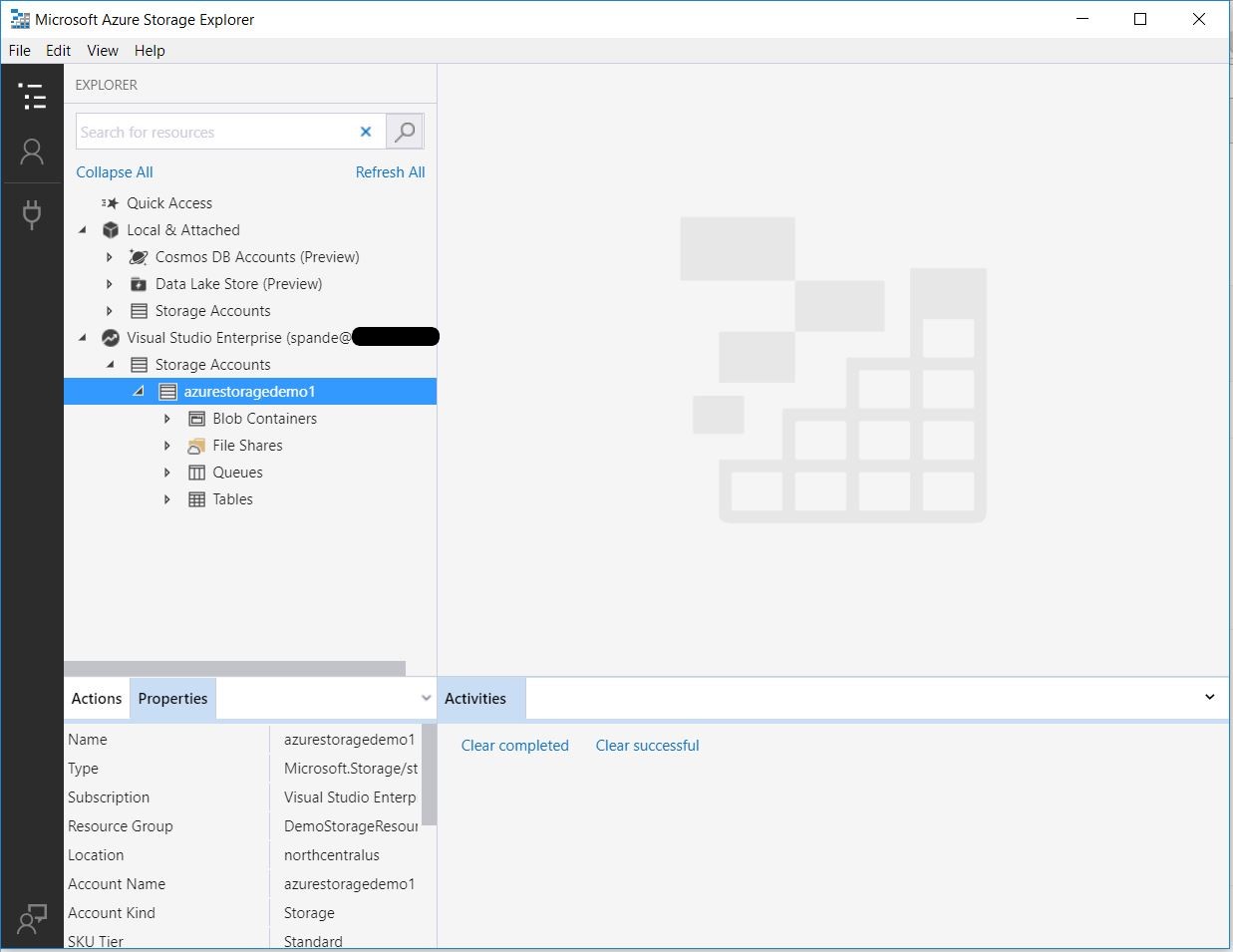


Once you double click the storage account, you will notice that all the related services such as Blob, Files, Tables and Queues under your storage account.

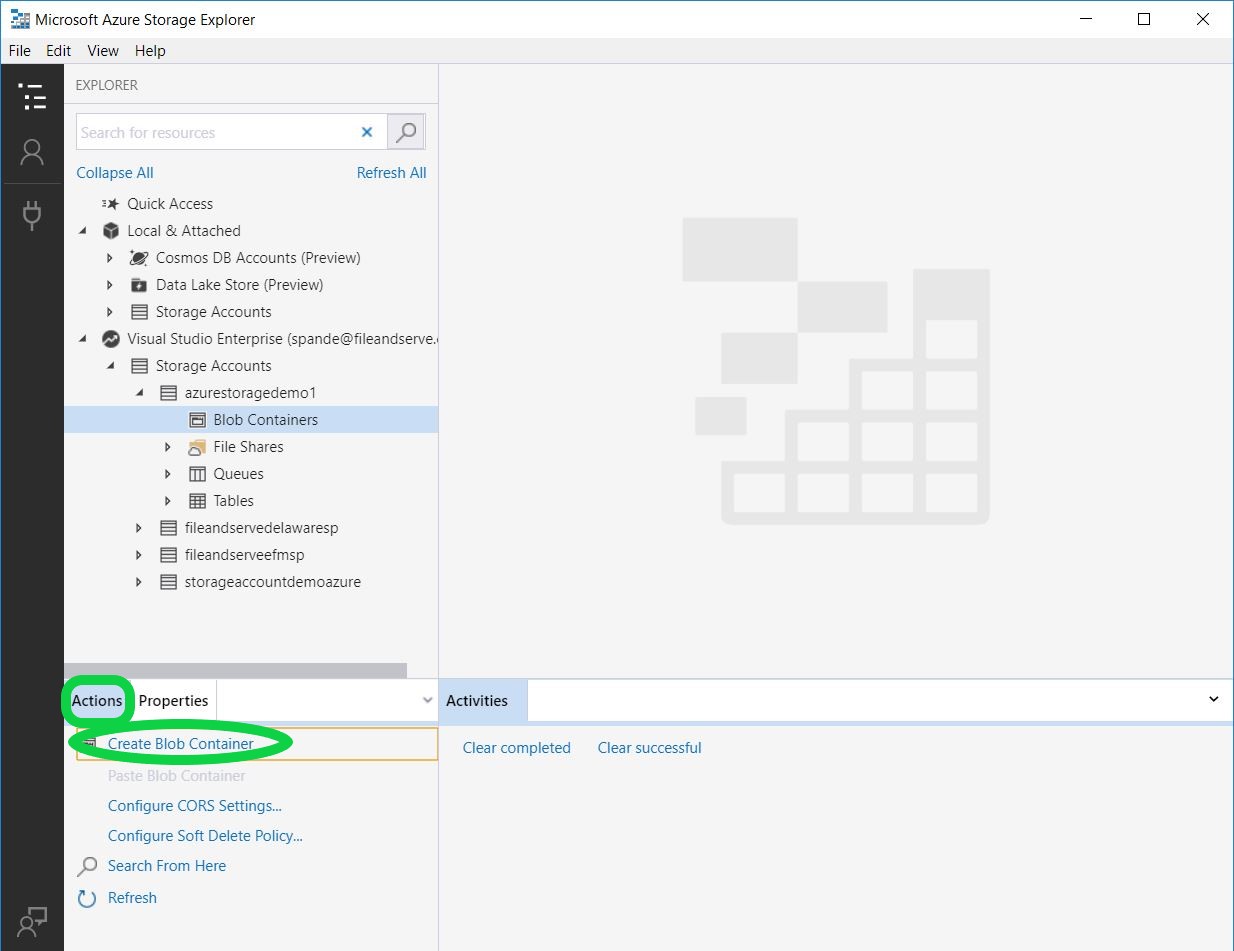


**Azure Storage Explorer:**

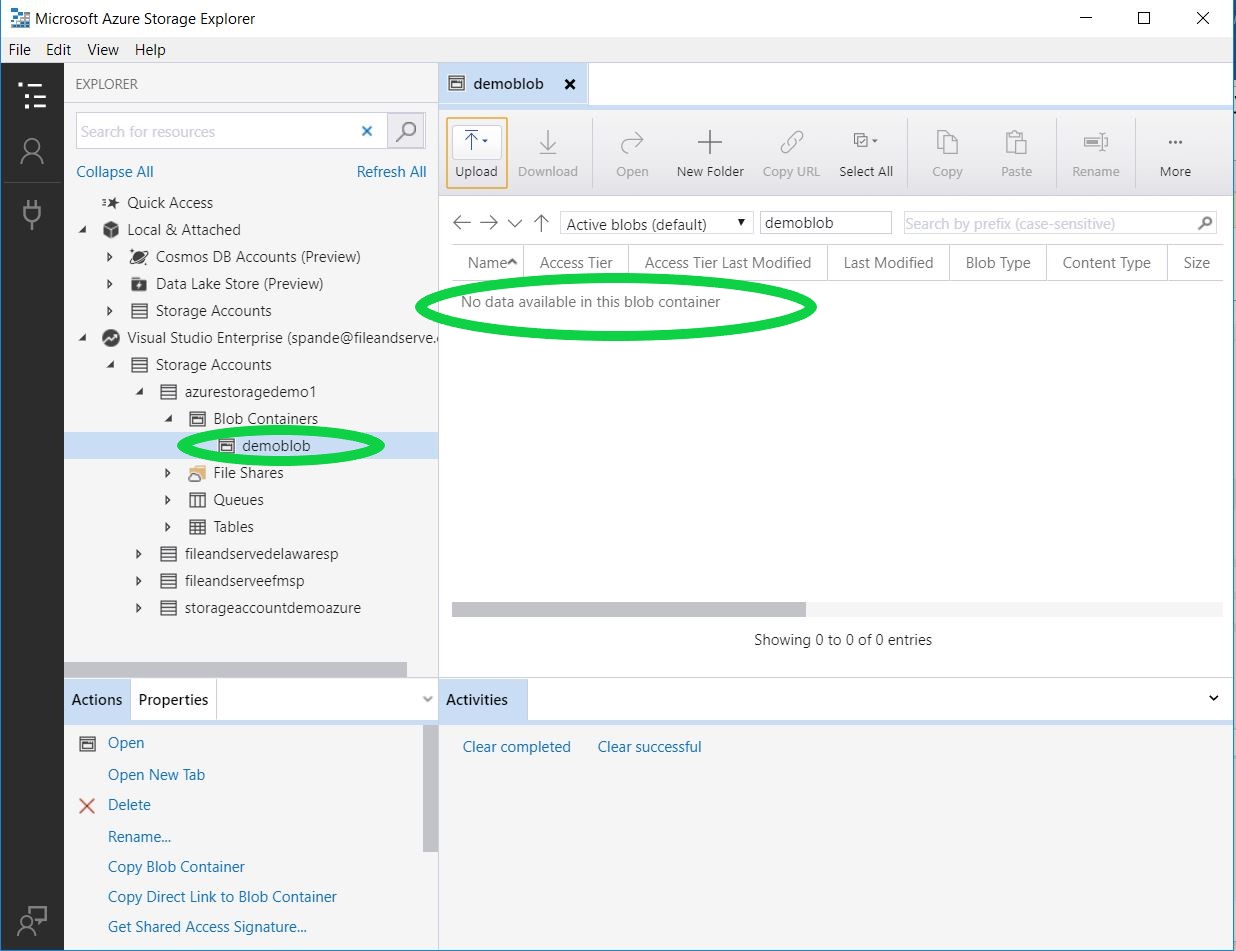
Now, to look deeper on the data level for each storage type from your Storage Account you need to have your Azure Storage Explorer Setup. Let’s move our focus to Azure storage Explorer. After you have installed Azure Storage Explorer from [here](https://azure.microsoft.com/en-us/features/storage-explorer/) you will have to provide your Azure Account details. Once the setup is complete you will notice that your Storage explorer shows up your Storage account details as below.



As we are doing to upload our data to Blob storage, let’s add a new Blob container to our Blob storage. Let’s navigate to *Actions* tab ->*Create a Blob Container* andcreate our first blob container for this storage account called “demoblob”.

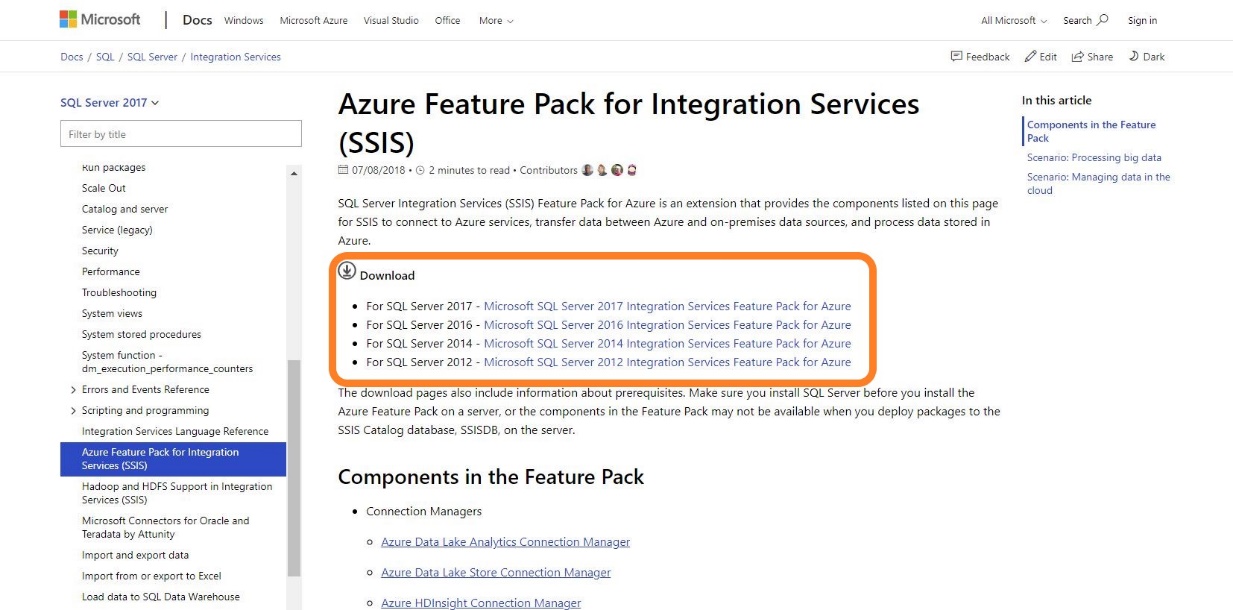


You will now notice that blob container named “demoblob” is created and there is no data currently present in our demoblob.



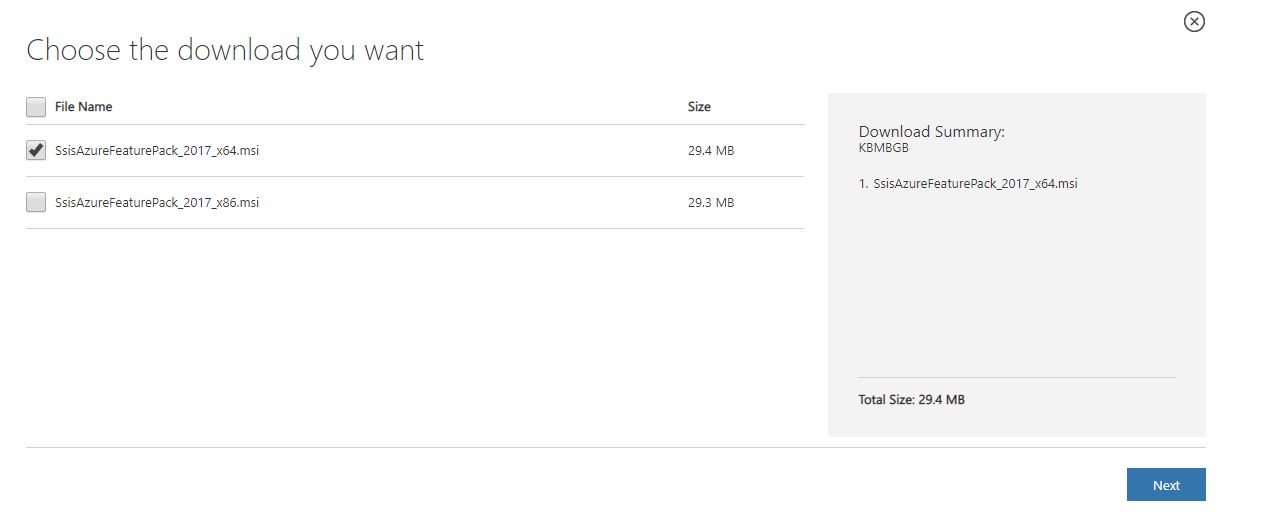
**Azure SSIS Feature pack**

We are now going to get the Azure Server pack for the Integration Services. Azure Service Pack is an extension which enables the user to upload the On-Premise data over to Azure. You will need to install the SSIS feature pack for the Visual Studio and SQL Server version you are going to use. You can download the pack from [here](https://docs.microsoft.com/en-us/sql/integration-services/azure-feature-pack-for-integration-services-ssis?view=sql-server-2017).

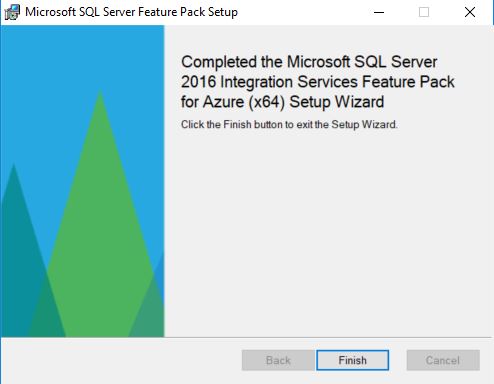


I am going to download SQL Server 2017 pack and specifically I have chosen   
*SsisAzureFeaturePack\_2017\_x64.msi* for my x-64bit processor. You will have to choose the type of msi

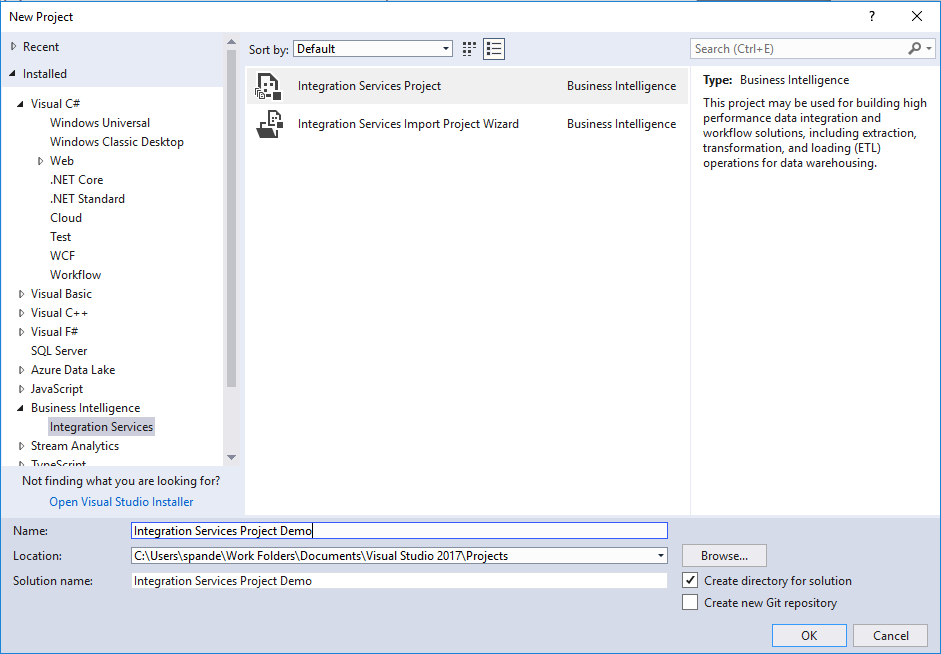
based on your machine requirements.



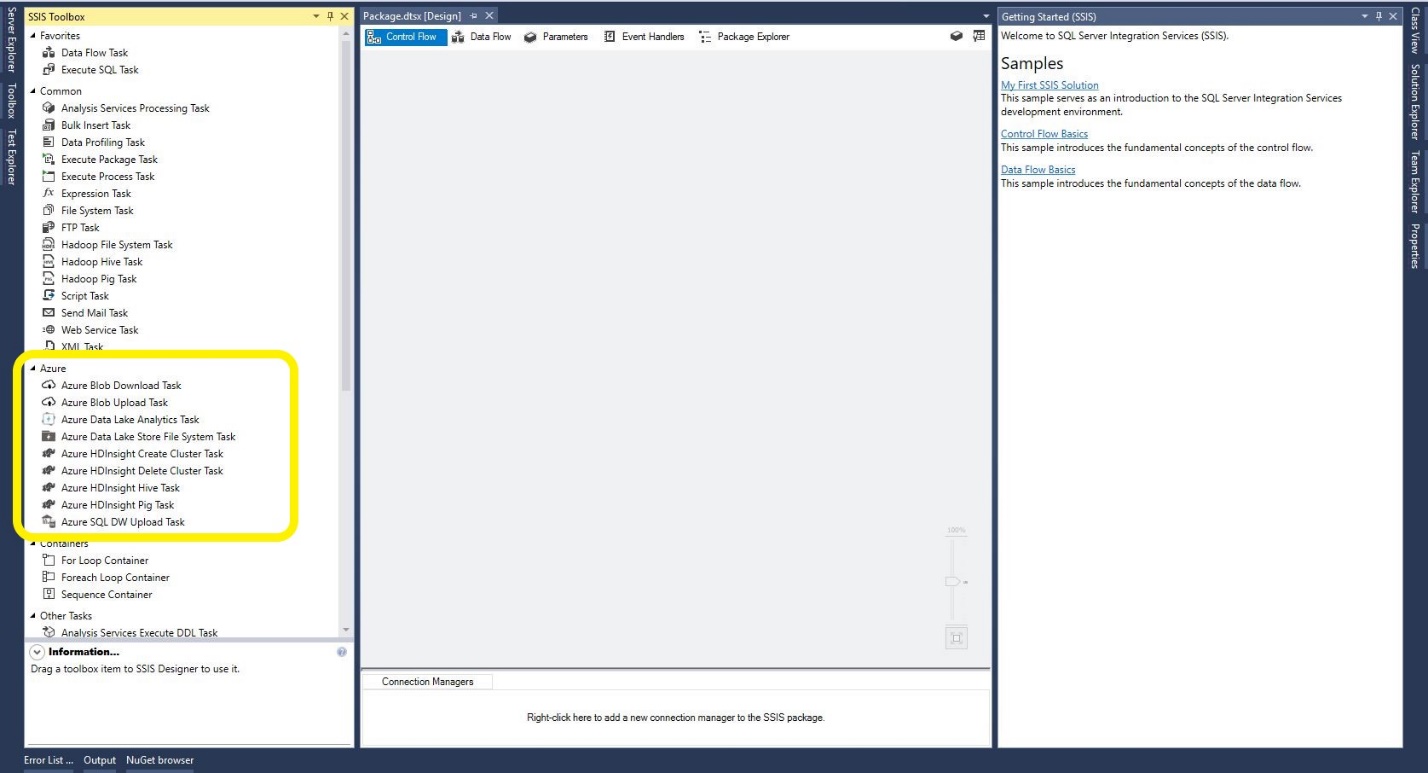
After the download has finished, I am going to install the .msi with just default setting. Once the installation is successfully done you will see the below screen.



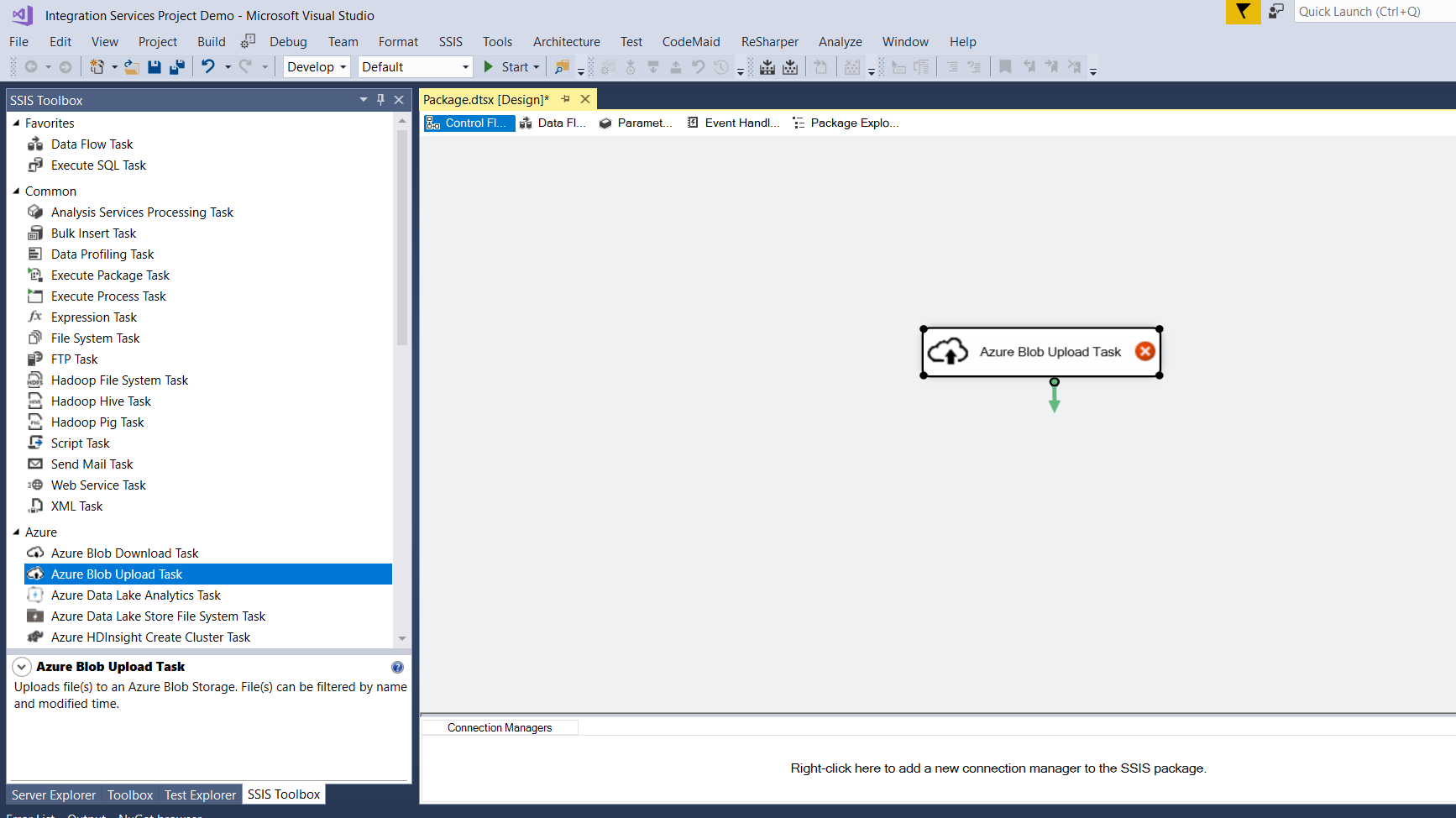
Now click Finish. After the installation has finished let’s go to Visual studio and create our empty package for demo.



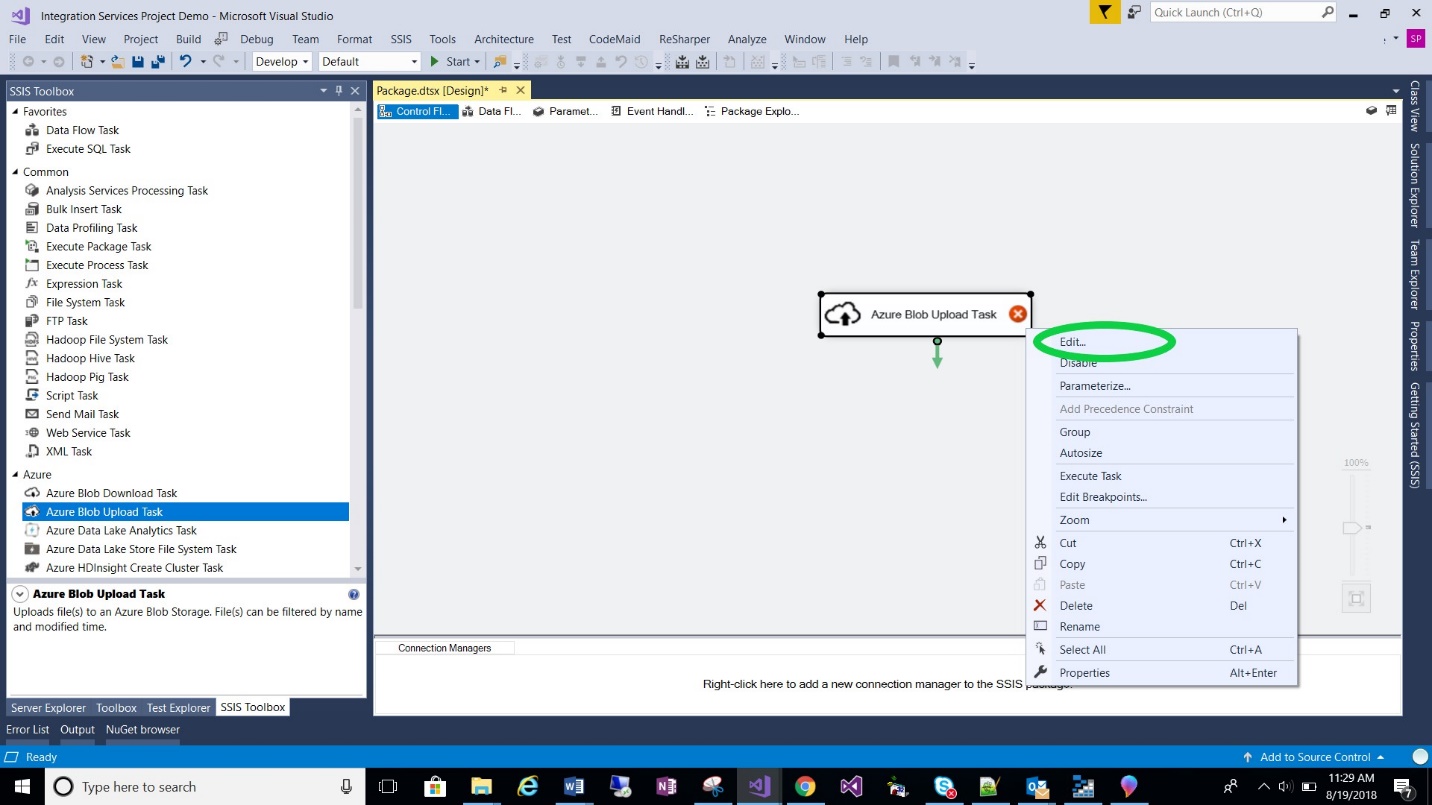
Once the project is successfully created, you will notice that the features which just got installed with service pack will show up under Azure section in SSIS toolbox. The Azure service pack provides an ability to connect to varied options such as Azure Blob Storage, Azure Data Lake, Azure HDinsights for processing massive amounts of data his includes PIG, Hive tasks and Azure SQL DW for data warehousing task.



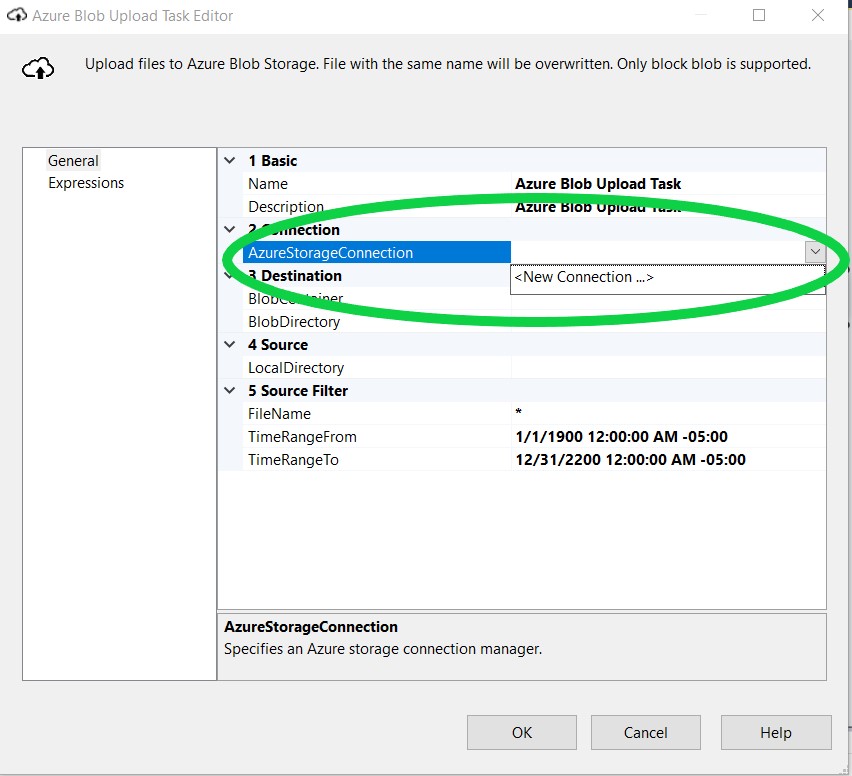
Now, as we have to upload the data to Azure Blob lets select/drag and drop the *Azure Blob Upload Task.*



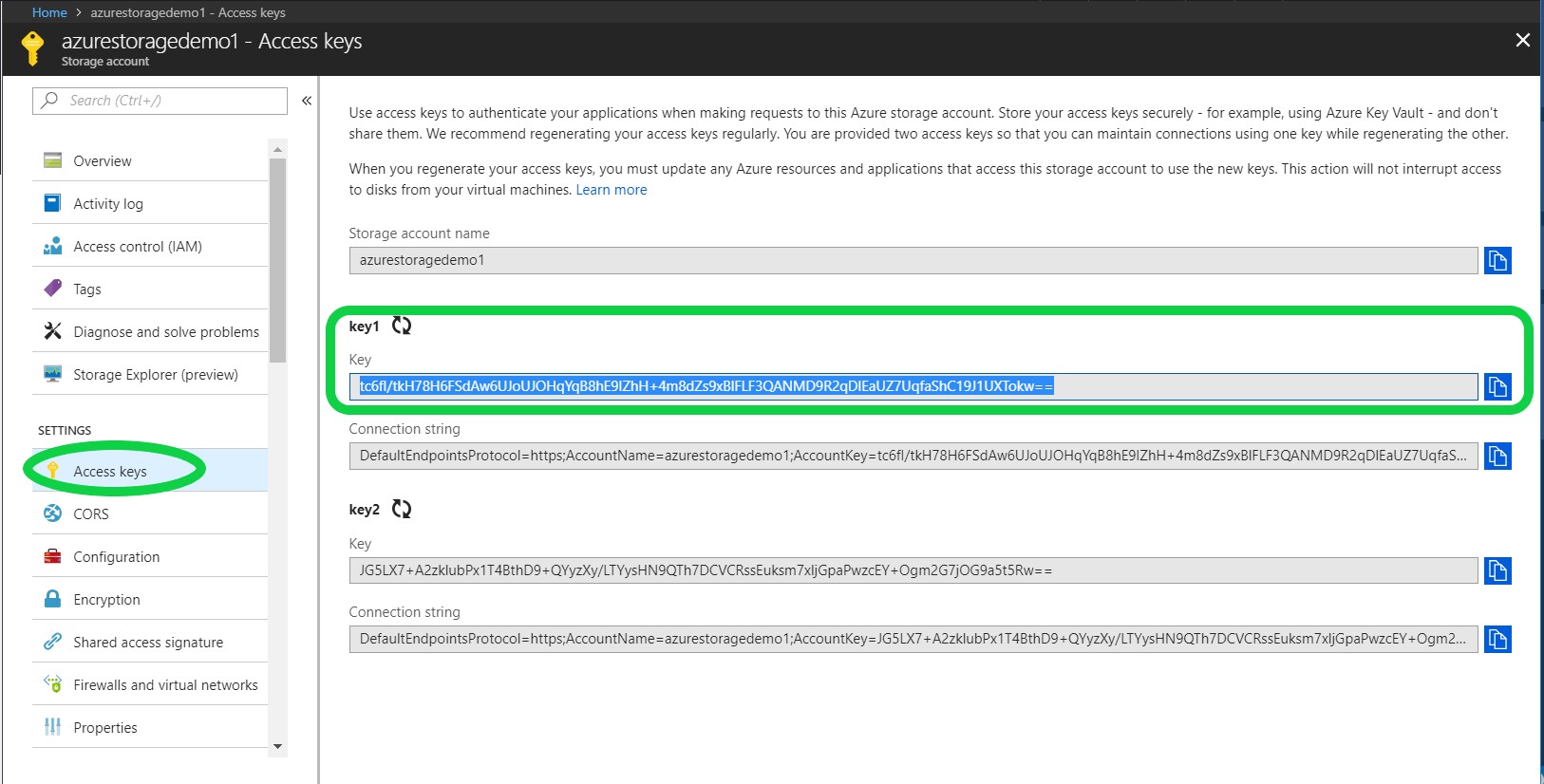
Let’s configure our task to upload the data to Azure. Right click the task and click on *Edit.*



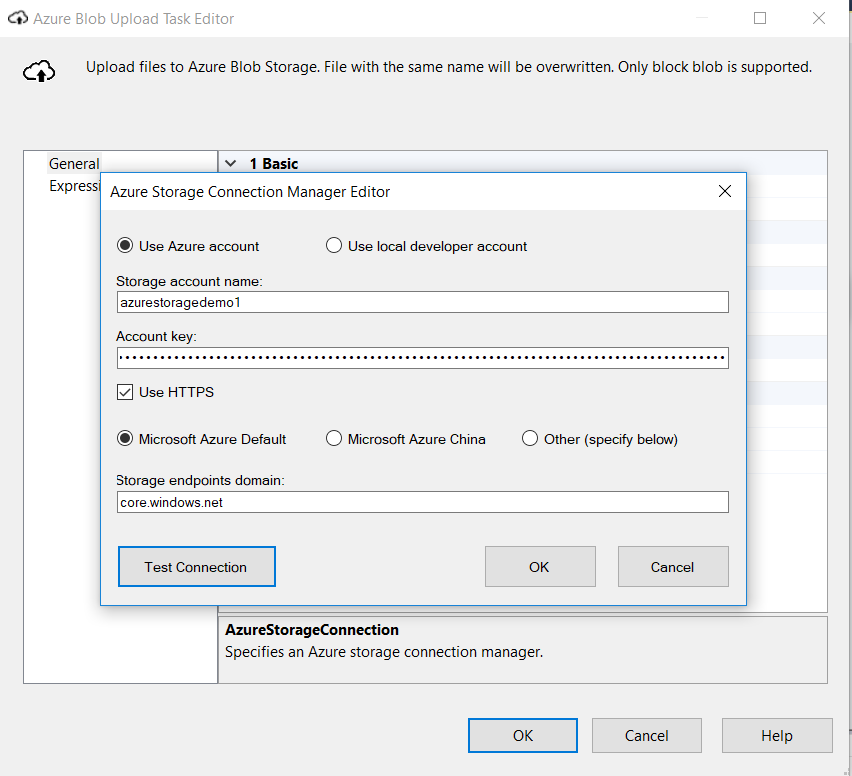
Once you click Edit, you will see the Azure Blob Upload Task Editor. We will have to now add a new Azure Storage Account Connection details to our task, so that it knows what Azure Account you want to upload the data.



You will be asked to provide the Storage account name and the Key. You can grab the key for your Storage account from Azure portal. Go to the *Access Key*s and for now I am picking Key1. You can also pick Key2.



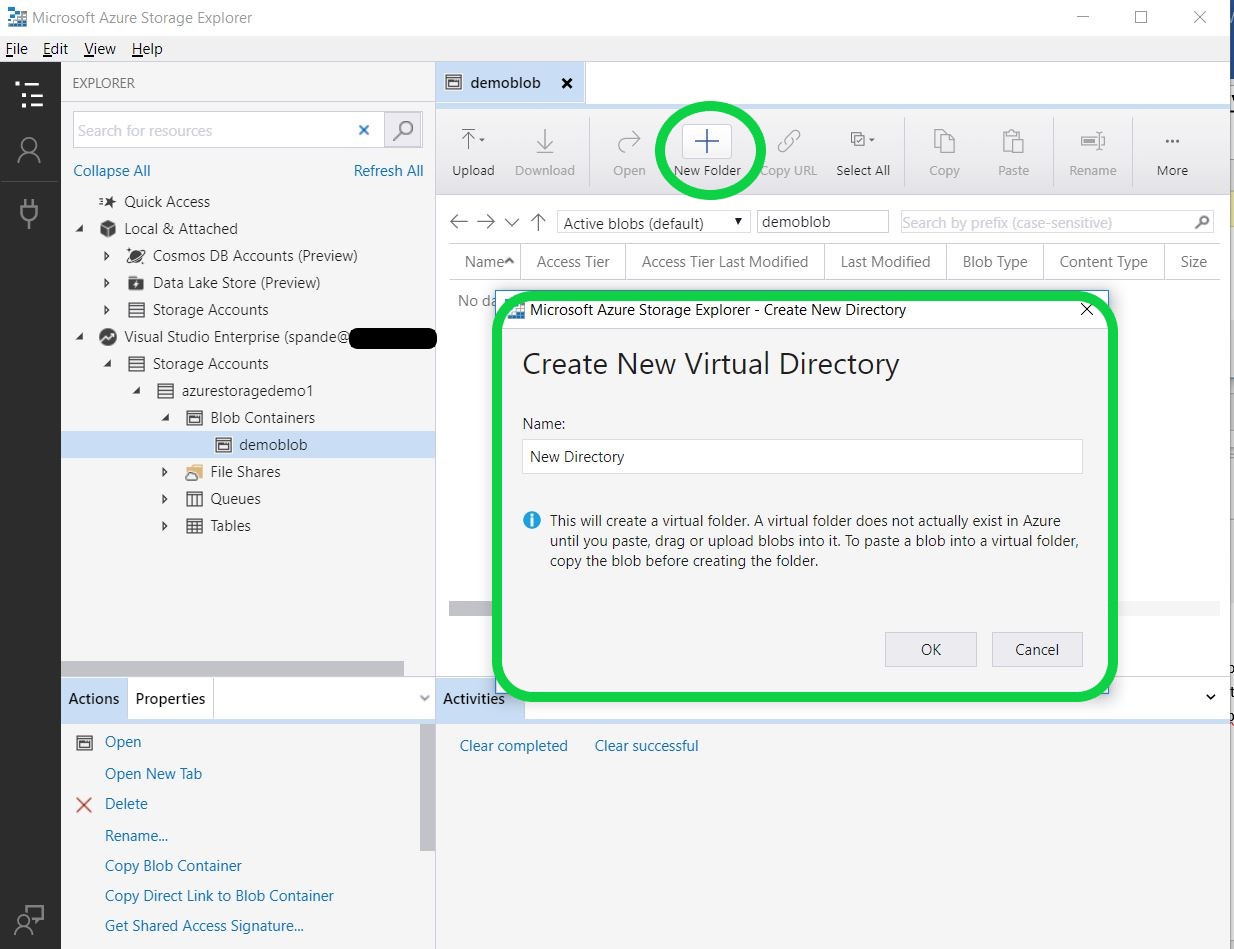
And as you know name of our Storage Account is “azurestoragedemo1”.



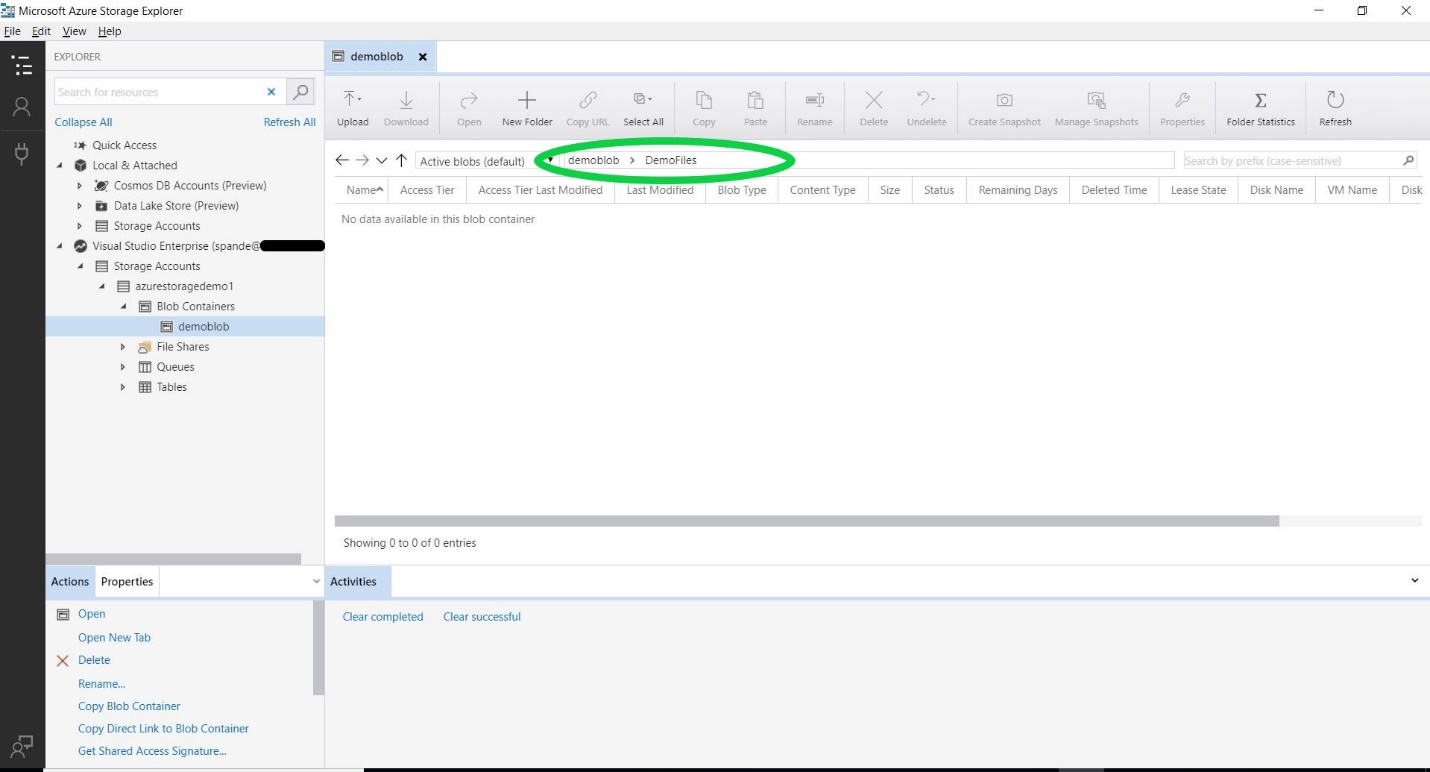
You can test your connection by clicking Test Connection button.



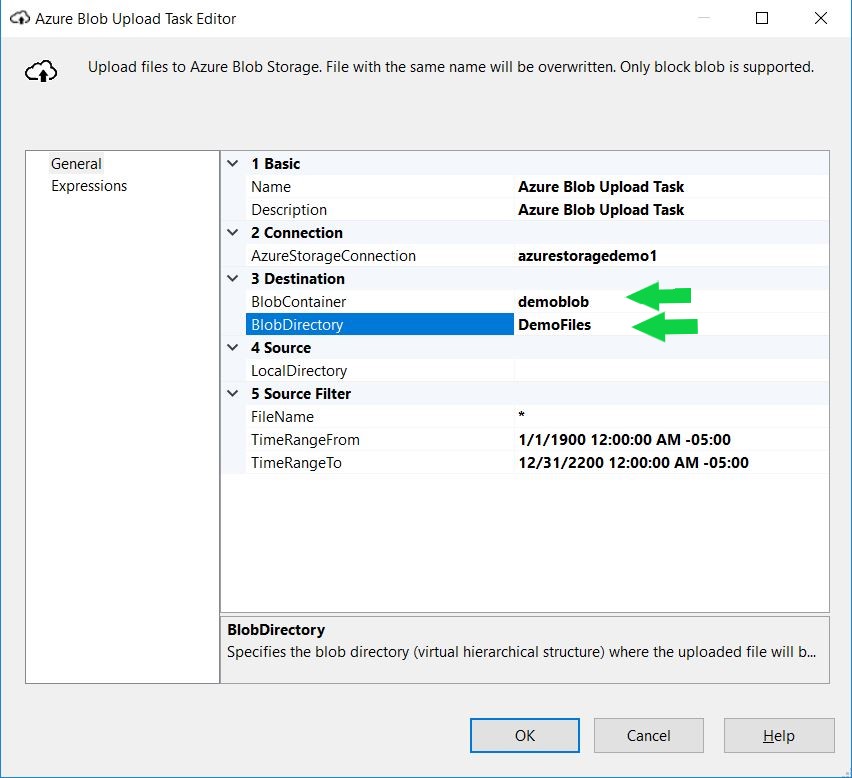
After you see that Test Connection Succeeded, you can click OK. Now for option 3 we have to provide Blob details, let’s go back to our Azure Storage Explorer to get these details. There are 2 things that we need to provide here they are Blob container name which is “demoblob” that we created some time back. Additionally, we need a Blob Directory. Let’s create a Blob directory now. After you go back to Azure Storage Explorer and your blob is selected click on *New Folder* and you will see below Create New Directory Window.



I have provided my new Directory name as “DemoFiles”. Once you click OK you will notice that the new directory is created in your Blob.

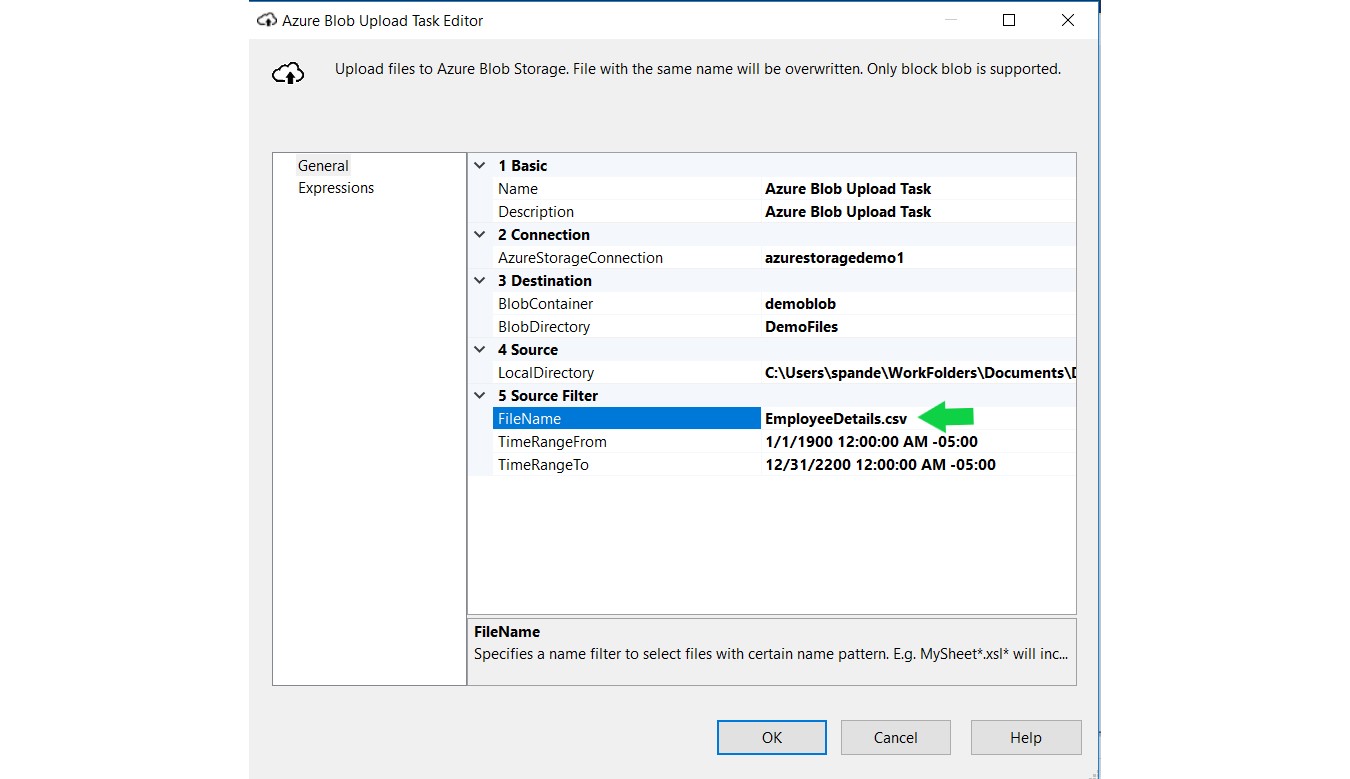


Now you are all set to fill in the value on Section 3 of our Azure Upload Task Explorer. So why wait? Let’s just finish our form.



In Section 4 you just need to specify the details of your Local Directory i.e. the location of the data you want to upload. I have my .csv file stored at location “C:\Users\spande\WorkFolders\Documents\DemoFiles”. Let’s specify that in our LocalDirectory.

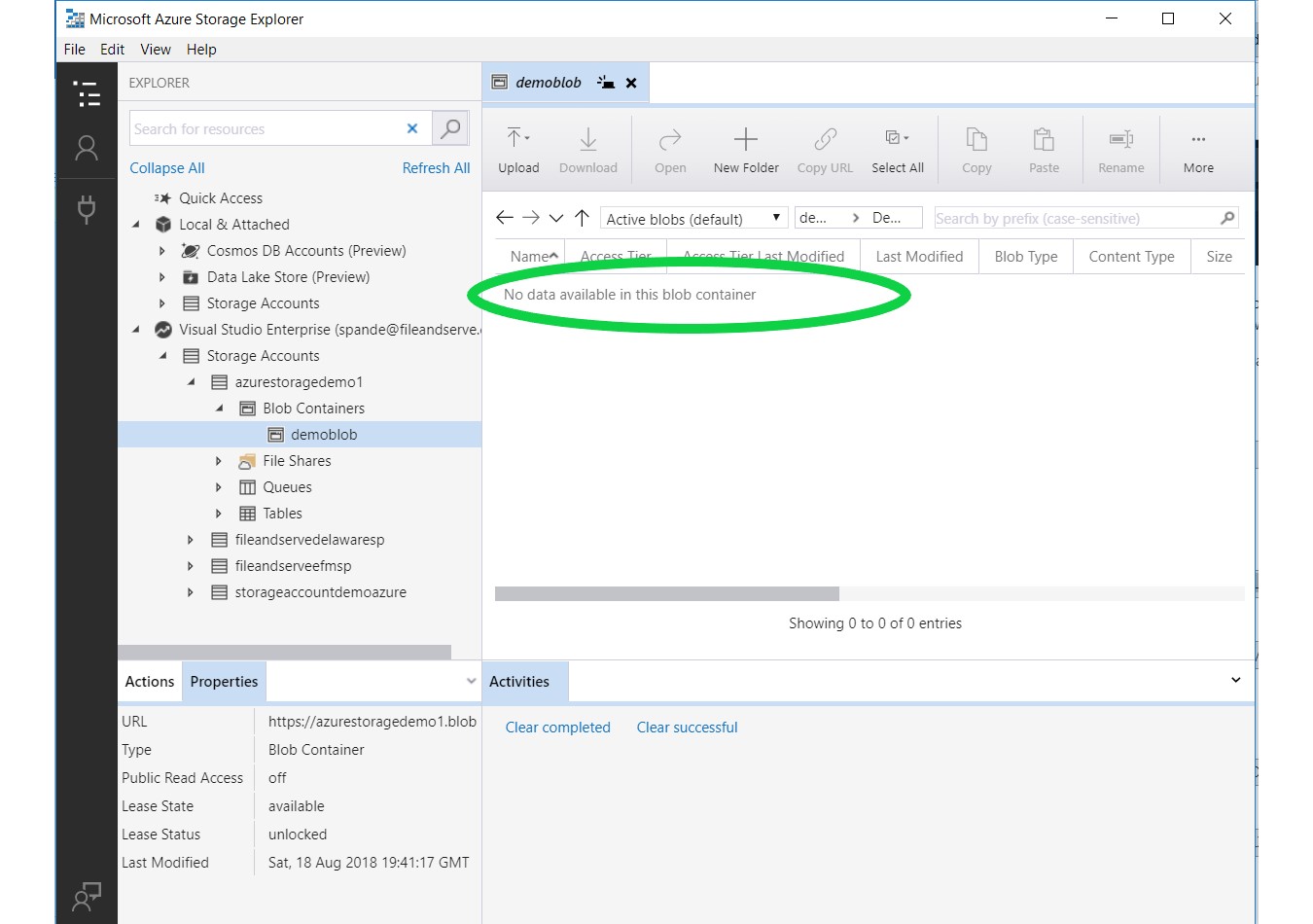
In section 5, for the Filename I have mentioned specific file name instead of “\*” As I want to upload just one file called EmployeeDetails.csv. If you want to upload everything from your directory just keep “\*”.



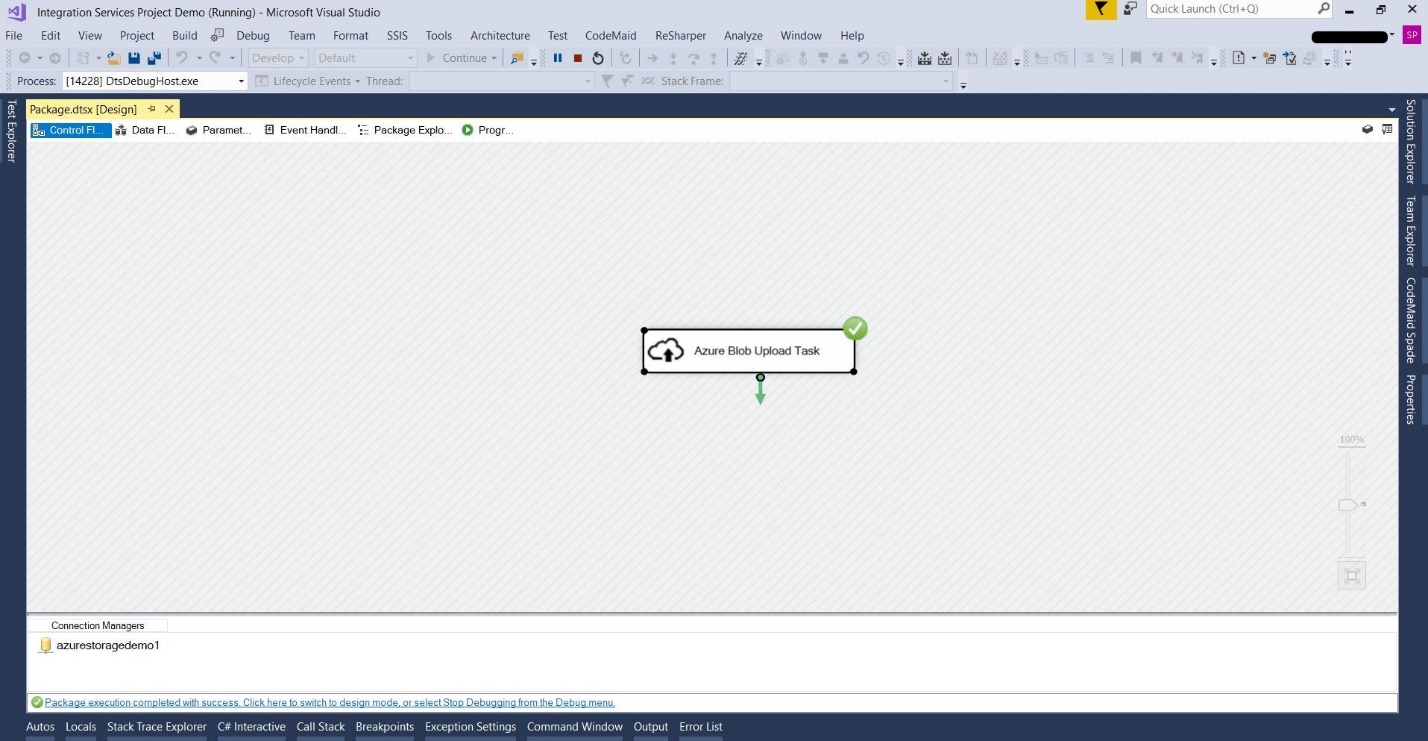
Finally, you can just click OK and you are all set.

**Uploading the data to Azure**

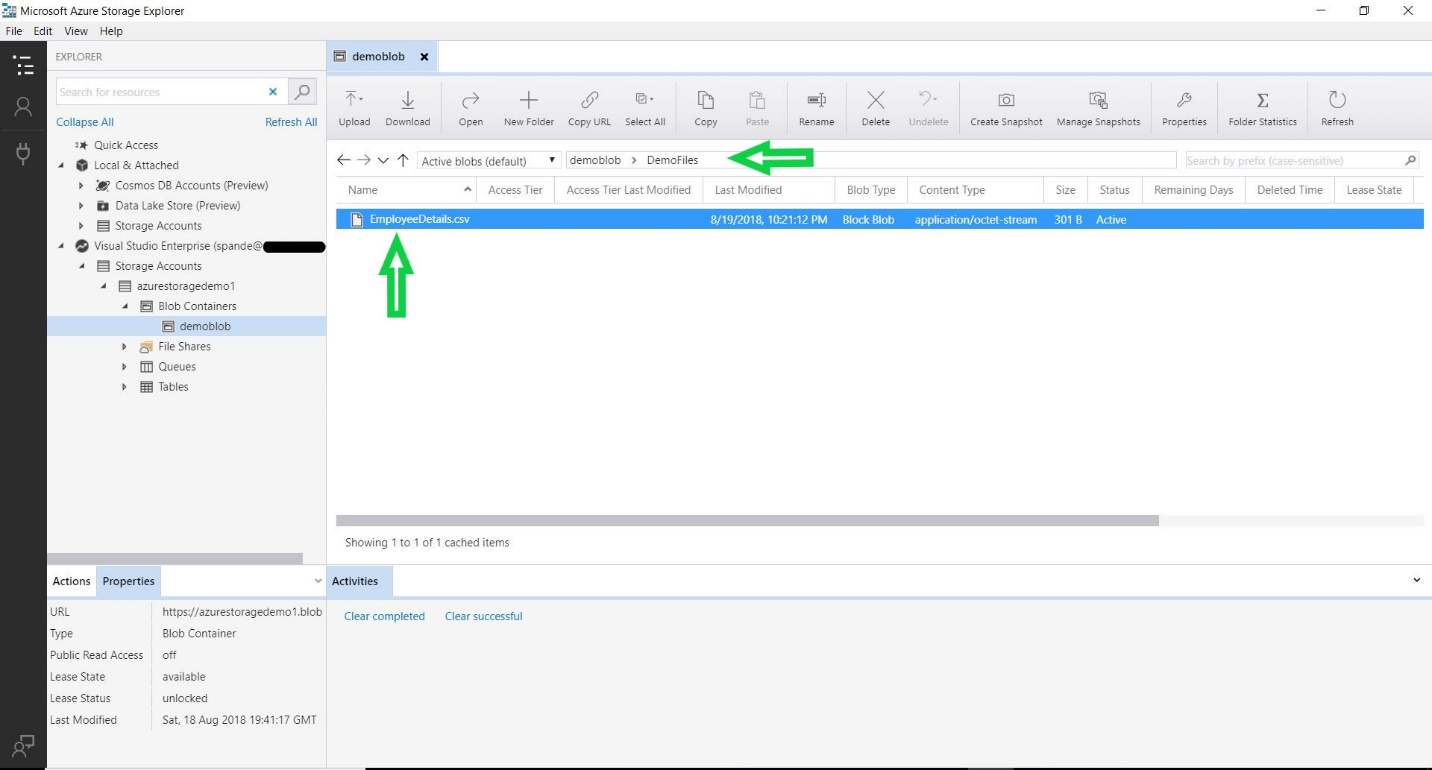
As you might see there is no data present in our DemoFiles directory for demoblob.



To upload the data over to Azure Blob storage, the last step that you just need to do is Start the task. Once the task is successfully run you will see the file being uploaded to the destination directory ‘DemoBlob’. You will see a green successful execution sign on the right top corner of your task as below.



After the task has been successfully executed, to check the uploaded file you can go back to Azure Storage Explorer and navigate to the DemoFiles Directory. You will notice that the *EmployeeDetails.csv* file has been successfully uploaded to Azure Blob as shown in the below screenshot.



**References:**

<https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/move-data-to-azure-blob-using-ssis>

**Conclusion**

Azure Storage has capabilities to store different types of data and it enables user to access the data using various operations. We can use Azure SSIS feature pack to upload the On-Premise data to Azure Blob Cloud based storage. Azure SSIS task can be configured to upload the data to Azure Storage Destination Folder and by running this task you will be able to efficiently upload the data to Blob Storage.