In this demo clip, let’s look at creating a Data Factory. A little tour of Data Factory so that we familiarize ourselves with its features. And we will also create an Azure SQL database that will host our integration services catalog.

Login to your Azure account, and create Azure data factory service. Specify a name for the service and note that The name of the Azure data factory must be *globally unique.*

You can also create a Data Factory using either PowerShell, .NET, Python, REST API or using Azure Resource Manager Templates. Please check the Microsoft documentation for other ways to create a Data Factory.

And let me put the Data Factory into a new resource group. A Resource Group, if you are not aware, is a logical grouping of resources for an application. So we are going to place all the resources that we create in this resource group. For running SSIS packages, you need to choose Azure data factory V2. At the time of recording, V2 with data flow is in preview and is only available if you whitelist your Subscription.

And select the location for the data factory. This is where the metadata of the data factory is stored. As you learnt in the slides, this location can be different from the data stores and the Compute services that the data factory connects to.

Once done, Click Create.

After the creation is complete, go to Data Factory. Select ‘Author & Monitor’ tile which opens the Data Factory application.

So this is the overview section.

And, if you go to the Author section, you can view Data Factory assets like the Pipelines, datasets, and within the connections tab, you can view Linked Services and the integration runtime

By default, when you create a Data Factory, an Azure Integration Runtime is created that supports operations on data stores and Compute services in public network. The location of this Runtime is auto-resolve, which means that the location is decided dynamically at runtime. Generally, Data Factory will try to use the Azure Integration Runtime that is closest to the data sources. You can also manually create an Azure IR if you would like to explicitly define the location of the IR.

You can also define triggers but we will discuss this in later modules.

In the Monitoring section, you can monitor the Pipeline runs. We will talk about Monitoring in the last module.

Let’s create Azure SQL Database. Azure Data Factory creates the SSIS Catalog and SSISDB database on this database server.

So, first create the database, and put that within the same resource group, and Let’s just create a blank database.

And configure the Logical Server for the database. Server admin is used to connect to the server for the first time before you can create additional users. And I will have the server in East US 2 Location. Make sure that you check ‘Allow Azure Services to access server.’ This is required to allow the integration runtime to access SQL Database to create SSIS Catalog. This is not a security best practice as it configures the firewall to allow all connections from Azure, including the connections from subscriptions of other customers. Nevertheless, leave this one checked & you will learn later how to workaround this. But for now, to ensure security, just make sure you grant access to only the authorized users with username and password.

Let’s leave other default options and create.

Let’s check out the resource. This is the database. From here, you can navigate to the server as well.

You can also access the resource from the Resource Group. So we have a data factory, SQL Server and a SQL Database.

And if I need to connect to this server from SSMS, I need to add a Server-level firewall rule to allow my client IP to access the server. You can also add a database-level firewall rule, after logging into the server by running a T-SQL Script. So you can configure both firewall rules and virtual network rules in this section. We will talk about virtual network rules later.

Open your SSMS and let's try connecting to our server with admin user and password. To access Azure SQL Database from your local computer, ensure your firewall allows outgoing communication on TCP port 1433.

Within databases, you should be able to see our empty database.