**Introduction**

Extract, Transform, and Load (ETL) jobs are very common in Enterprise projects. They allow businesses to gather data from multiple sources, transform it and consolidate it to a target system. Performing ETL on cloud can be much more effective.

So, let's understand how we can implement the ETL concepts using the Microsoft Cloud's Azure Data Factory service, which can help you to integrate data from multiple data sources and implement hybrid ETL pipelines.

**What is Azure Data Factory?**

It is a fully managed, serverless data integration service using which we can visually integrate data sources with more than 90 built-in, maintenance-free connectors at no added cost. Using Azure Data Factory we can easily construct ETL and ELT processes code-free in an intuitive environment. Or you can write your own code and then deliver the integrated data to Azure Synapse Analytics to unlock business insights

**Azure Data Factory Architecture**

**Benefits of Azure Data Factory**

* **Easy to use**: Build ETL and ELT pipelines code-free, with built-in Git and CI/CD support.
* **Cost Effective**: We can enjoy this service as pay-as-you-go. It is a fully managed, serverless cloud service that can scale on demand.
* **Powerful**: We can ingest all the data from on-premises servers or software-as-a-service (SaaS) data with the help of more than 90 built-in connectors. We can even orchestrate and monitor the pipelines at scale.
* **Intelligent**: We can use autonomous ETL features to leverage the operational efficiencies and to enable citizen integrators.

**Hands-On: Implement ETL using Azure Data Factory**

First of all, Sign In to Azure Portal and inside Azure Portal search for **Data Factory**.

Click on the **"+ Create"** button for creating the Data Factory.

Under the **Basics** tab, select your created **Resource Group**. Then enter the Name of the Data Factory (in our case it's **dcadfdemo**).

Under **Git Configure** Tab select the **Configure Git Later** Option.

Now go to the **Review + Create** tab and wait for passing the validation. Once you have passed the validation, click on the **Create** button.

It will take a couple of minutes to create the Data Factory.

While the Data factory is being created, let's now create a SQL Database in our Resource Group.

For doing so, first of all search for **SQL Database** in the search bar and select the Azure **SQL Databases**.

Now click on the **"+ Create"** button present on the top-left section of the page.

Under the **Basic** tab, select your **resource group**. Then give a name to your database (in our case it's **adfdemo**). Finally, click on the **Create new** button in front of the Server option.

Now, give your server a name (in our case it's **deadfserver**), enter admin as your username and create a password as well.

Once this is done, click on the **Create** button. After this click on the **Review + Create** button.

Then finally, click on the **Create** button. It will take 2-3 minutes for creating the SQL Database.

While the SQL Database is being created, we can start creating the Azure **Storage Account.**

For doing so, search for a **Storage Account.**

Now click on the **"+ Create"** button on the top-left section of the page.

Then, under the **Basic** tab select your **resource group** and give name to your storage account (in our case it's **dcadfstorageaccount**). Finally click on the **Review + Create** button.

Once validated, click on the **Create** button.

It will also take a couple of minutes for creating the resource.

Once all the resources have been created, go to your resource group. Here we would be see the resources we just created - SQL Database, Storage Account, Server, and Data Factory.

So, now let's build our ETL pipeline. But before that let's see what would be our plan of action. So here it is:

*So:*

1. We would be creating a blob storage using the Storage Account that we have created.

2. Then, we would create a table in SQL Database.

3. After that, inside the ADF(Azure Data Factory), we would be creating a pipeline in which we would be having Integration Runtime. Although, we would be automating this so as to reduce manual work.

4. Then, we would be creating two Linked Services, one for sending data from the CSV file (from Blob Storage) to the database and another for sending data from the database to the table in the SQL Database.

5. We would be using the **Copy Activity** for doing so.

Now we have a plan, so let's go back to Azure Portal and start creating our stuff.

**Step 1: Creating Blob Storage**

First of all, we would be creating a Blob Storage and we would upload NYC\_Bus\_Data.csv to it.

NYC\_Bus\_Data.csv contains the data related to the public Buses of New York City. So in your resource group click on the **Storage Account** that you have created.

Now, under the **Properties** tab click on **Blob Service**.

Now click on the **"+ Container"** to create a new container for our csv file.

Give it a name such as **input** and click on the **Create** Button.

Now, go inside your container and click on the **Upload** button.

Browse for Nyc Bus Data.csv file from your system and Click on the **Upload** button.

With this, we have completed the first part of our plan.

**Step 2: Creating SQL Database Table**

Now let's move to the second step, i.e., creating a table in the SQL Database. For doing so, go to the Resource Group and click on the **SQL Database** that you have created.

From the leftmost section, click on the **Query Editor (Preview)** button.

It would ask for the username and password that you have created while creating the server. Enter the user id & the password and click on **OK**.

After clicking **ok**, you would be getting this error message:

To resolve this error, we must add our IP to the allowed-list of IPs of the firewall of the server that we have created. For doing so, open the resource group from home in a separate tab and click on the server that you have created.

Now, from the leftmost section, scroll down to **Security** and Click on **Firewall and Virtual Networks.**

After this click on the **"+ Add Client IP"** button and then click on **save**.

Your IP gets added into the firewall automatically.

Now let's head back to the login page for SQL Database and try again. This time we would be able to login to the system successfully.

Now, run the following query for creating a table named as buses:

CREATE TABLE buses (

recordedattime varchar (30),

directionref varchar (1),

vehicleref varchar (10),

vehiclelocation\_latitude decimal (8,5),

vehiclelocation\_longitude decimal (8,5),

expectedarrival time varchar (30),

scheduledarrival time varchar (30)

);

Copy above query and paste it in the query editor of the SQL Database. Click on the **Run** button to run the query.

It would create a table and we can verify it by exploring the **Tables** on the left side of the page.

**Step 3: Creating Pipeline in Azure Data Factory**

After this, let's move to the 3rd step, i.e., creating a pipeline in Azure Data Factory. For doing so, go to **resource group > Data Factory**. After that, click on Open **Azure Data Factory Studio.**

It may ask you to authenticate, so click on Authenticate

From the leftmost section of the page, click on the **Authon** button

Click on the three dots in front of Pipeline and click on **New Pipeline.**

Give the new pipeline a name (in our case it's **demopipeline**).

From the left side of the page, click on three dots in front of **Dataset** and click on **New Dataset**.

From the **Azure** tab select the **Azure Blob Storage**. Then click on the **Continue** button.

Now, select the format type of Data as **CSV** and click on **Continue**.

Now, the **Set Properties** page would appear. Give your data source a name, like **InputSource**, and click on the "**+ new**" option within the list of options in **Linked Service** dropdown.

Now, **New Linked Service (Azure Blob Storage)** screen would appear. In this, give the service a name as **AzureBlobStorageSource** and under **Azure Subscription**, select the **Free Trial** for now. Then, select your **Storage Account Name** and then finally click on the **Create** button.

Now, enter the **file path**. You can enter the file path manually or click on the file icon beside it for browsing the file in Azure.

Now, select the "**First row as header**" option. Finally, click on **Ok**.

Again, click on the three dots in front of **Dataset** and click on **New Dataset**.

**New Dataset** tab would open. Select **Azure SQL Database** under **Azure** tab and click on **Continue.**

The **Set Properties** page would appear. Enter the name as **Azuresqltable** and under **Linked Service** click on the **"+ New"** option.

Under the **New Linked Service (Azure SQL Database)** page, select your **free subscription** under **Azure Subscription**, select **dcadfserver** under **Server Name**, and **adfdemo** under **Database Name**.

Now, enter the **username & password** and click on the **Test Connection** button.

It would give you an error for not being able to access the server using a particular IP. To resolve this error, like earlier, there is the requirement to add our IP into the firewall of the server that we have created.

For doing so, open the resource group from home in the separate tab and click on the server that you have created.

Now, from the leftmost section, scroll down to **Security** and click on **Firewall and Virtual Networks.**

After this, add a **Rule Name**, the **Start IP** as the copied IP with the last octet replaced by o,, and the copied IP as the **End IP**.

Then, click on the **Save** button.

Now, go back to the **New linked service (Azure SQL Database)** page and click on **Test Connection** again. This time the test would be successful.

Finally, Click on **Create**.

Now, in the **Set Properties** page, select **Table Name** and click on **Ok**.

Go back to the **Pipeline** page in Azure Data Factory. From the left section, drag and drop the **Copy Data Activity** from the **Move & Transform**

Now, under the **Source** tab, select **InputSource** for **Source Dataset**.

After this, under the **Sink** tab, select **AzureSqlTable** for **Sink Dataset**.

Click on the **Debug** button for checking the correctness of the Pipeline.

If there is no error, then it will show you an operation **succeeded** message.

But let's do a double check.

Go to SQL Database and run the query **`SELECT \* FROM Buses`.**

Now let's run the query `**SELECT Count(\*) FROM Buses**` and get the count of rows.

We have successfully set up an ETL pipeline on Azure Data Factory.