Data Movement: Data Lake Store to Azure SQL DB

Updated on: 12/29/2016

# Introduction

You will learn how to setup a recurring job to run and how to copy the output of that job in a recurring format from the Data Lake Store to SQL DW. This is a common pattern employed to move transformed data to an database for reporting/analytics on aggregated data scenarios.

# Prerequisites

For this lab, you will need:

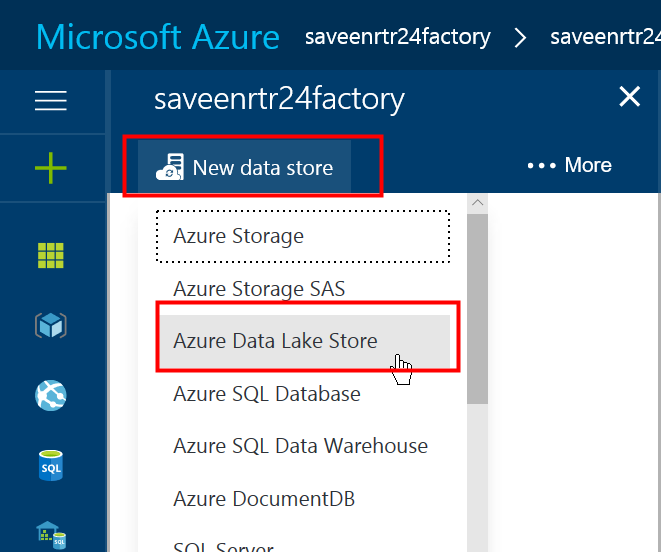
* Access to a Data Lake Store account that you can write to (MyStoreAccount)
* Access to a Data Lake Analytics account that you can submit jobs to (MyStoreAccount)
* Access to the **adltrainingsampledata** Data Lake Store account that you can read from
  + All Microsoft FTE already have this
  + If you are a not a Microsoft FTE request access to the aldsandbox security group.
* Access to a Data Factory account (MyFactory)
* Access to a Data Warehouse DB
* Access to an Azure SQL Server

# Exercise 1: Setup a recurring job

* Create a new ADF account or reuse an existing one
* Go to the Azure Portal <http://portal.azure.com>
* Navigate to your ADF account
* Click **Author and deploy**

## Linking the Data Lake Store Account

* Click **New data store > Azure Data Lake Store**



* You’ll see a draft of some JSON Text. Modify it as indicated below
  + Remove the properties marked “[Optional]”
  + Set **name** to **ADLTrainingSampleData**
  + Set **dataLakeStoreUri** to **https://adltrainingsampledata.azuredatalakestore.net/webhdfs/v1**
* Click on **Authorize** and login
* The JSON will look like this:

{

"name": "ADLTrainingSampleData",

"properties": {

"type": "AzureDataLakeStore",

"description": "",

"typeProperties": {

"authorization": "\*\*\* a very long https url \*\*\*",

"dataLakeStoreUri": "https://**adltrainingsampledata**.azuredatalakestore.net/webhdfs/v1",

"sessionId": "\*\*\* a very long string \*\*\*",

}

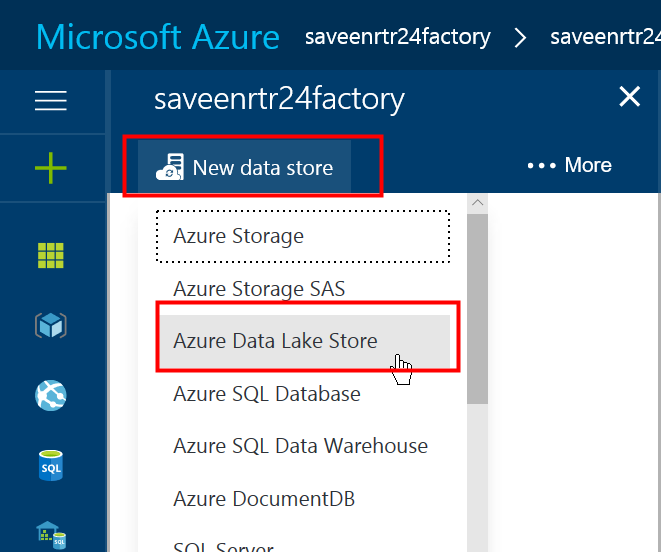
}

}

* Click **Deploy**

## Linking the Data Lake Store Account (2)

* Click **New data store > Azure Data Lake Store**



* You’ll see a draft of some JSON Text. Modify it as indicated below
  + Remove the properties marked “[Optional]”
  + Set **name** to **MyADLS**
  + Set **dataLakeStoreUri** to **https://saveenrtr24store.azuredatalakestore.net/webhdfs/v1**
* Click on **Authorize** and login
* The JSON will look like this:

{

"name": "MyADLS",

"properties": {

"type": "AzureDataLakeStore",

"description": "",

"typeProperties": {

"authorization": "long string",

"dataLakeStoreUri": "adl://saveenrtr24store.azuredatalakestore.net",

"sessionId": "long string"

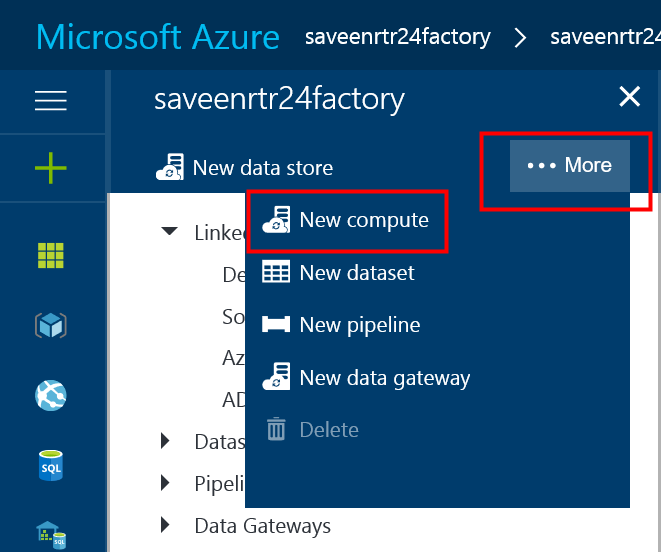
}

}

}

* Click **Deploy**

## Linking the Data Lake Analytics Account

* Click the ellipses button (…)
* Click **New Compute**
* 
* Select **Azure Data Lake Analytics**
* You’ll see a draft of some JSON Text. Modify it as indicated below
* Remove the properties marked “[Optional]”
* Set **name** to **ADLACompute**
* Set **accountName** to your ADLA Account’s name (**MyAnalytics**)
* Click on **Authorize** and login
* The JSON will look like this:

{

"name": "ADLACompute",

"properties": {

"type": "AzureDataLakeAnalytics",

"description": "",

"typeProperties": {

"authorization": "a very long https url",

"accountName": "saveenrtr24analytics",

"sessionId": "a very long string"

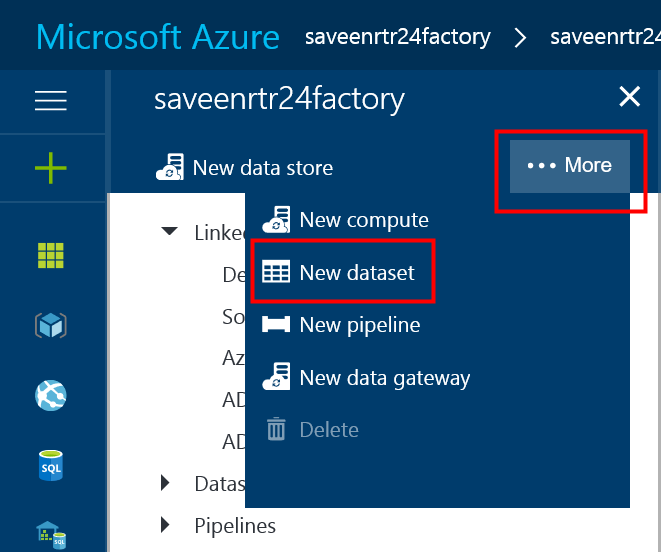
}

}

}

* Click **Deploy**

## Define where our data is stored.

* Click the ellipses button (…)
* Click **New dataset**
* 
* Click **Azure Data Lake Store**
* NOTE: Our job takes in 3 input Datasets and writes 1 output Dataset so we need to define 4 Datasets in total
* Input Dataset 1: GithubProjectMembers

{

"name": "GithubProjectMembers",

"properties": {

"published": false,

"type": "AzureDataLakeStore",

"linkedServiceName": "ADLTrainingSampleData",

"typeProperties": {

"fileName": "ProjectMembers\_large.csv",

"folderPath": "/GHData/",

"format": {

"type": "TextFormat",

"rowDelimiter": "\n",

"columnDelimiter": ","

}

},

"availability": {

"frequency": "Day",

"interval": 1

},

"external":true

}

}

* Dataset 2: GithubProjects

{

"name": "GithubProjects",

"properties": {

"published": false,

"type": "AzureDataLakeStore",

"linkedServiceName": "ADLTrainingSampleData",

"typeProperties": {

"fileName": "Projects.csv",

"folderPath": "/GHData/",

"format": {

"type": "TextFormat",

"rowDelimiter": "\n",

"columnDelimiter": ","

}

},

"availability": {

"frequency": "Day",

"interval": 1

},

"external":true

}

}

* Dataset 3: GithubUsers

{

"name": "GithubUsers",

"properties": {

"published": false,

"type": "AzureDataLakeStore",

"linkedServiceName": "ADLTrainingSampleData",

"typeProperties": {

"fileName": "Users.csv",

"folderPath": "/GHData/",

"format": {

"type": "TextFormat",

"rowDelimiter": "\n",

"columnDelimiter": ","

}

},

"availability": {

"frequency": "Day",

"interval": 1

},

"external":true

}

}

* Dataset 4: UsersPerProject

{

"name": "UsersPerProject",

"properties": {

"structure": [

{

"name": "ProjectName",

"type": "String"

},

{

"name": "CountryCode",

"type": "String"

},

{

"name": "NumberOfUsers",

"type": "Int32"

}

],

"published": false,

"type": "AzureDataLakeStore",

"linkedServiceName": "MyADLS",

"typeProperties": {

"fileName": "CountofProjectUsers.csv",

"folderPath": "/output/",

"format": {

"type": "TextFormat",

"rowDelimiter": "\n",

"columnDelimiter": ","

}

},

"availability": {

"frequency": "Day",

"interval": 1

}

}

}

## Create a Pipeline

* Click “New Pipeline”
* Note that we’ve simplified the configuration of this pipeline for this exercise. ADF supports many additional parameters and options.

{

"name": "ComputeNumberOfUsersPerProject",

"properties": {

"description": "This is a pipeline that computes the number of users per project",

"activities": [

{

"type": "DataLakeAnalyticsU-SQL",

"typeProperties": {

"script": "@projects = EXTRACT id int, url string, owner\_id int?, name string, descriptor string, language string, created\_a DateTime?, forked\_from int?, deleted int?, updated\_a DateTime? FROM \"adl://adltrainingsampledata.azuredatalakestore.net/GHData/Projects.csv\" USING Extractors.Csv(); @projectmembers = EXTRACT repo\_id int, user\_id int, created DateTime?, ext\_ref\_id string FROM \"adl://adltrainingsampledata.azuredatalakestore.net/GHData/ProjectMembers\_large.csv\" USING Extractors.Csv(); @users = EXTRACT id int, login string, name string, company string, city\_country string, email string, created DateTime?, type string, fake int?, deleted int?, longitude decimal?, latitude decimal?, country\_code string, state string, city string FROM \"adl://adltrainingsampledata.azuredatalakestore.net/GHData/Users.csv\" USING Extractors.Csv(); @result\_set = SELECT p.name, u.country\_code, COUNT(DISTINCT u.id) AS NumberOfUsers FROM @projects AS p INNER JOIN @projectmembers AS pm ON p.id == pm.repo\_id INNER JOIN @users AS u ON u.id == pm.user\_id GROUP BY p.name, u.country\_code; OUTPUT @result\_set TO \"/output/CountofProjectUsers.csv\" ORDER BY NumberOfUsers DESC USING Outputters.Csv();",

"degreeOfParallelism": 3,

"priority": 100,

"parameters": {}

},

"inputs": [

{

"name": "GithubProjectMembers"

},

{

"name": "GithubProjects"

},

{

"name": "GithubUsers"

}

],

"outputs": [

{

"name": "UsersPerProject"

}

],

"policy": {

"timeout": "06:00:00",

"concurrency": 1,

"executionPriorityOrder": "NewestFirst",

"retry": 1

},

"scheduler": {

"frequency": "Day",

"interval": 1

},

"name": "ComputeNumberOfUsersPerProject",

"linkedServiceName": "ADLACompute"

}

],

"start": "2016-07-12T00:00:00Z",

"end": "2019-08-08T01:00:00Z",

"isPaused": false,

"pipelineMode": "Scheduled"

}

}

* Click Deploy