Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 11, November-2018

An Alternative Approach to SSIS: using Azure Data Lake with Microsoft's Big Data Language U-SQL

Supriya Pande
File & Serve Xpress LLC.
Irving, TX, USA

***Abstract*— With a lot of data being generated in varied formats there is always been a high dependency on highly efficient ETL tools like SSIS. With the increase of usage of cloud these days, we need to find different options as well so that there is another alternative that we can use to transfer and perform analysis on Cloud. One of such ways is storing the data on Azure Cloud using Azure Data Lake Store and further process the data using Big Data query language called U-SQL. We will see the key highlights of using this approach and try to find how to implement it.**

Keywords—SSIS; SQL Server Integration Service; Azure; Azure Data Lake; Microsoft's Big Data Language; U-SQL

I. INTRODUCTION

Nowadays, organizations are generating a lot of data in varied formats which needs to be stored and analyzed efficiently. We all know that to perform ETL functions organizations are using highly efficient and reliable tools like SSIS since over a decade. In this article, we are going to have a look at an alternative approach to the SSIS where we will be storing the data on Azure Cloud using Azure Data Lake Store and further process the data using Big Data query language called U-SQL. Also, we will look deeper into why the new approach is better than the SSIS approach. So, without further delay let's move on to briefly understand the keywords involved in the article.

A. SQL SERVER INTEGRATION SERVICE (SSIS)

SQL Server Integration Service (SSIS) is the extraction, transformation, and loading (ETL) solution to load the data from various sources such as XML, CSV, XLS, TSV, etc. It can be used for a variety of high-performance integration and migration related tasks to update the Data Warehouse. In SSIS development approach, a developer has to build the packages where each package executes the set of commands which takes the data from the data source and further transforms the data into another data format. This transformed data is then further loaded into the target system. The developer has to update or create the package depending on what format the source file is. This whole process of can include writing queries, creating database objects and cleaning up the data. These tasks any be considered as a burden on developer as it might be sometimes cumbersome and repetitive to do all the steps for every task. Now a days Organizations are generating more and more data in various formats and its becoming difficult to store and process them by creating different format dependent tasks. Managing these tasks and keeping track of all of them is another hurdle. The plethora of technologies such as Data Lake, Hadoop etc. is the answer to these challenges.

B. Azure Data Lake Store

Azure Data Lake Store is an extensible store in Azure cloud which allows you to rapidly store the data in any form and lets you perform analysis on it. So, you can imagine putting all your unstructured data into a basket called Data Lake. We can store the structured data into the Data Lake by creating databases within Data Lake. You might have understood by now that the data being stored in Data Lake can be structured or unstructured. Data can be loading in various different ways to the Data Lake store such as by using the language (.NET, JAVA) SDKs, REST APIs, Azure Data Factory and so on. Once the data is uploaded to the Data Lake Microsoft's big data query language U-SQL can be used to query the data. Now let's move our focus to understanding what U-SQL is.

C. *U-SQL*

U-SQL is Microsoft's big data query language which is combination of T-SQL and C#. It a very powerful language which can help to extract and transform data in the required form. You can write the scripts to perform these operations. The scripts be programmed to write the output any of the desired format files. U-SQL enables you to efficiently analyze data in Data Lake Store, Azure Storage Blobs.

U-SQL comes with built in extractors to extract the values from various types of files. Currently, it supports Text, CSV and Tab separated files. You can also develop your custom extractors depending on the file format. In addition to extractors, there are a built-in U-SQL outputters which will help you to build your output files with the required format of your choice. Similar to extractors you can also develop your own outputters.

II. DEMONSTRATION OF AZURE DATA LAKE AND U-SQL APPROACH

Let's go much deeper to what we understood in previous section. Here I will demonstrate how we can use capabilities Azure Data Lake and U-SQL to load, extract, and transform the data. I am going to take a simple example where you have a CSV file which lists "CourseCode" which has the course code values, Total number of students as "TotalStudents" and course title mentioned as "CourseTitle" as shown below.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 11, November-2018

```
CourseCode,TotalStudents,CourseTitle
"CS-121",35,"Theory Of Computation"
"CS-122",38,"Programming Languages"
"ECE-121",20,"Introduction to Electronics"
"IT-122",40,"Information Technology for Beginners"
"CS-125",24,"Introduction to C"
```

Fig. 1. Course.csv file contents

For demonstration purposes, we are going to load this CSV file into the Azure Data Store and then write a U-SQL script that will bring us the output values where we want the total number of students in the CS code series. The output that we will essentially see for our input file will be 97 for "CS-121", "CS-122" and "CS-125" (35+38+24=97).

To work on this solution, we will have to create the Azure Data Lake Store Account for storing and replicating the data. In addition to store account we need Azure Data Lake Analytics Account which has all the scripts or queries stored to generate the desired output. These accounts reside in the Azure subscription that you can get here [1].

III. Creating Data Lake Storage Account & Azure Data Lake Analytics Account in Azure Portal

Once you login to Azure Portal [1], you will need to select to All services from the left options panel. Look for Analytics options. You will notice that there is an option for "Data Lake Storage Gen1" available under Analytics.

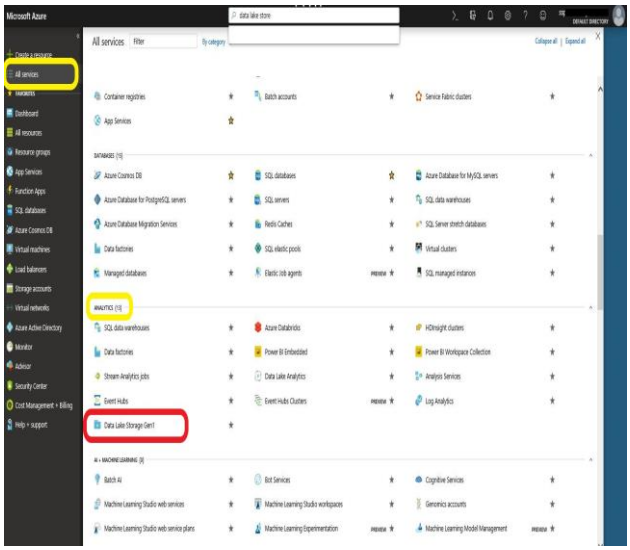

Fig. 2. Azure Data Lake Storage Gen1 option on Azure Portal

Add a new Data Lake Azure Storage by clicking Add button and provide all the relevant details by selecting the correct subscription. Here I am going to provide the name of the storage account as "datalakestoragedemosp". For my subscription I am going to choose "Visual Studio Enterprise" and create a new Resource Group called "datalakergp". I am going to select Central US as the closest location and I am going to select the Pricing Package as "Pay-as-You-Go" that means will pay the charges incurred based on usage. After all the details are filled up, I am going to click Create.

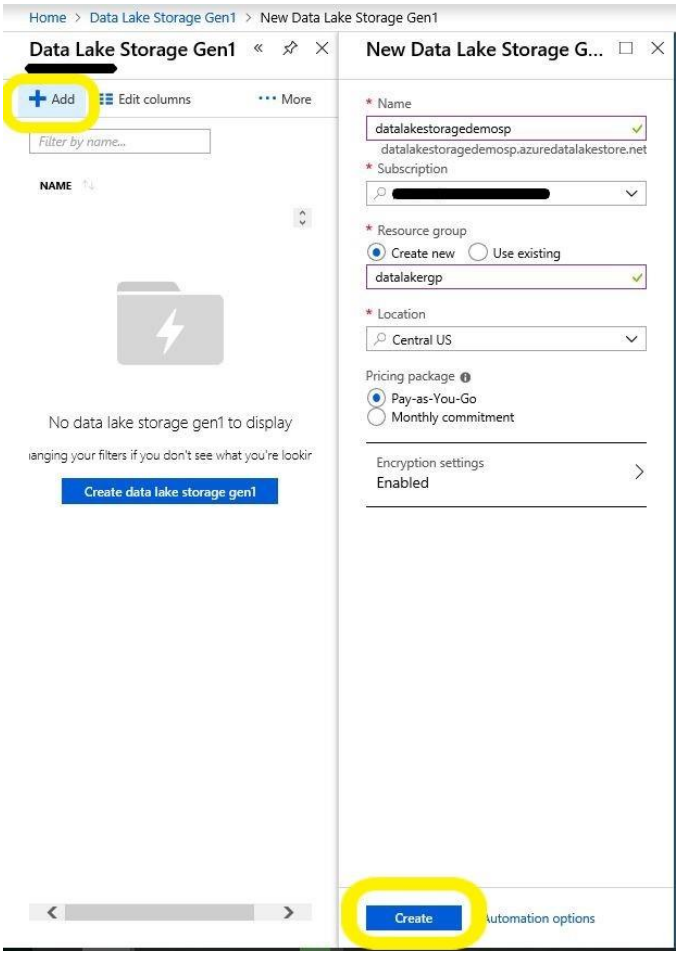

Fig. 3. Adding new Azure Data Lake

Once you are done creating the storage, we can now move on to creating Azure Data Lake Analytics account. Go to Azure home page and go to All services and look for Data Lake Analytics in Analytics section.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 11, November-2018

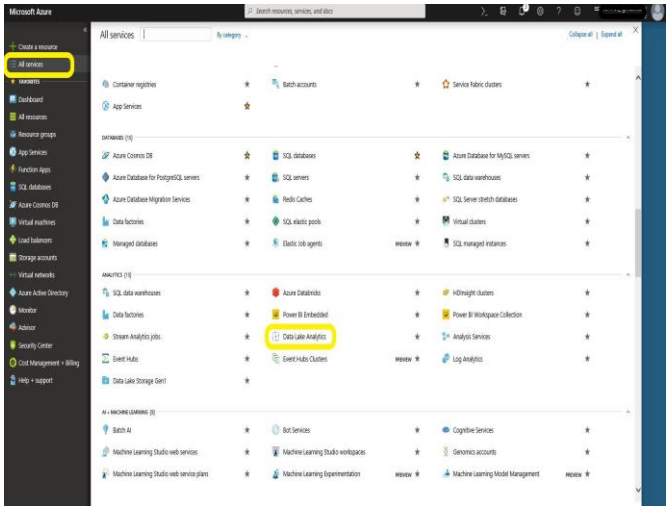

Fig. 4. Data Lake Analytics option on Azure portal

The whole process of creating the analytics account is almost similar to the Data Lake creation we just saw. Once you select Data Lake Analytics option you will see a new page being opened. Click on Add. Let's create a new Analytics account with name "demodatalakeanalyticssp". We will use the same details for the Resource Group and Location as we used for storage. Next, for the Data Lake Store lets select the store "datalakestoragedemosp" that we just created some time back and click Create.

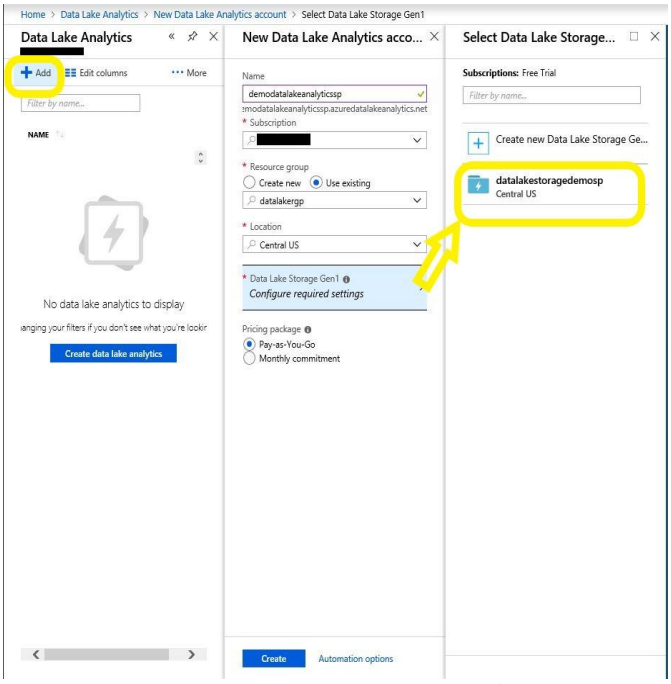

Fig. 5. Data Lake Analytics configuration

.

If you click Refresh on Data Lake Analytics page, you will notice that the Data Lake Analytics account is added.

Fig. 6. Data Lake Analytics account added sucessfully

IV. UPLOADING THE DATA TO AZURE DATA LAKE STORAGE

As I mentioned before, there are various ways to upload the data to Data Lake Storage. I am going to upload our sample CSV file using Azure portal. As you are on the home screen of the Azure portal, you can again go to All services => Data Lake Storage Gen1 and your Data Lake store will start showing up as below or you can just look up for your Data Lake Store in the Search option provided on top of Azure page.

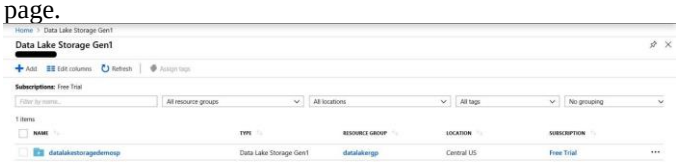

Fig. 6. Data Lake search

Once you open your Data Lake store, to look into the details of the contents of your store you will have to click on *Data Explorer option* from the list. You will notice that there are two default folders already present in our storage account.

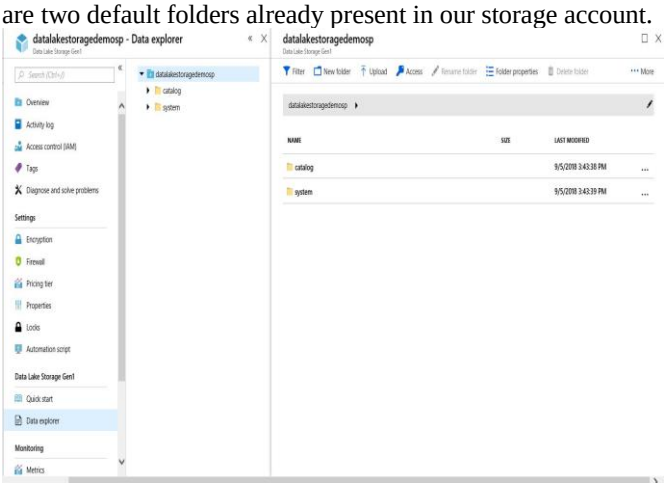

Fig. 7. Data Explorer option

We will add a new folder to keep our experiment files separate from other folders so that the contents will be properly organized and easily accessible. Let's click a New Folder option and create a "SampleFiles"

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 11, November-2018

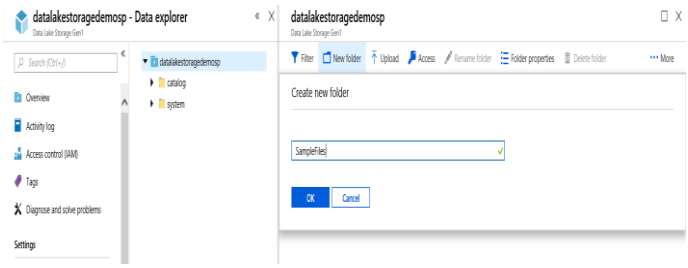

Fig. 8. Adding sample file

After the new folder is created I am going to upload our CSV file to this folder. Now click upload and select the file and click "Add Selected Files".

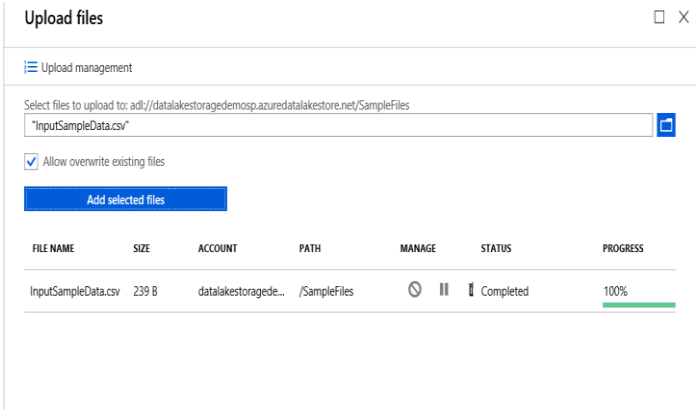

Fig. 7. Uploading .csv file

Now as we have our data loaded into the Data Lake store, we will focus on writing the U-SQL script.

V. WRITING U-SQL SCRIPT

For performing various analytical operations on the files we uploaded to Azure Data Lake storage we need to write the Azure Data Lake Analytics job essentially a U-SQL script that can extract the information from the source file and transform it into the required file format using built-in extractors and outputters. Similar to the way we navigated to Azure Data Lake, you can navigate to the page where the Azure Data Lake Analytics 'demodatalakeanalyticssp' account. To add a new U-SQL script click on New Job.

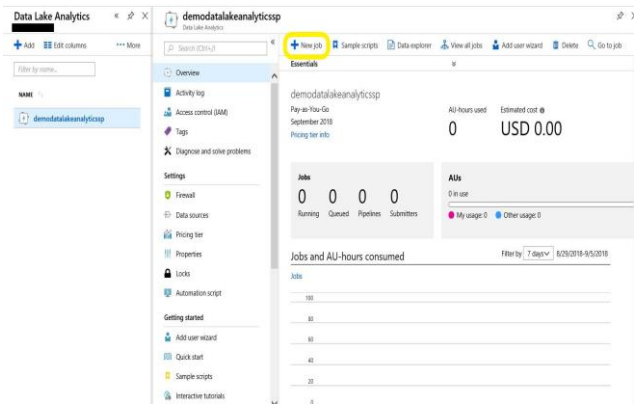

Fig. 7. Creating a new job

The script will have to extract the data from the csv file then perform the transformations on the data and finally have them written to the output file. U-SQL supports the concept of extraction by providing different extractors. There are different extractors available n U-SQL for various file types such as csv, Text, and Tab Delimited files. All the columns are extracted using extractors and supporting C# types are used depending on the type of the data elements. For instance, in the sample csv file we have TotalStudents column which is a number / int format in C#. Here is the code that I have written to extract all the columns from the *InputSampleData.csv* file stored in Azure Data Lake Storage. Here I will have a row-set variable like T-SQL naming convention `@coursecode` that will store the results of the extracted row-set.

```
@coursecode = EXTRACT CourseCode string,
TotalStudents int,
CourseTitle string
FROM "/SampleFiles/InputSampleData.csv"
```

As I mentioned before, there are various extractors supported by U-SQL as we have csv file as input we will use csv extractor. To skip the first row from the file we have explicitly assigned value for skipFirstNRows as 1.

```
USING Extractors.Csv(skipFirstNRows:1);
```

For transformation, we will have to write code similar to SQL code where we will take the substring value of the Course Code and sum up the number of students. The results will then be collected into the @total variable.

```
@total = SELECT CourseCode.Substring(0,2) AS DeptCode, SUM(TotalStudents) AS
TotalNumberOfStudents
FROM @coursecode
GROUP BY CourseCode.Substring(0,2)
```

For collecting the output into csv file I am going to write an output command which will write contents to StudentsCalculator.csv which will be created into OutputFiles folder

```
OUTPUT @total TO "/OutputFiles/StudentsCalculator.csv"
```

We can format output file using outputter command where we will mention that we need the column headers in the output as below code.

```
USING Outputters.Csv(outputHeader:true);
```

VI. COMPLETE CODE FOR U-SQL FILE

```
@coursecode = EXTRACT CourseCode string,
TotalStudents int,
CourseTitle string
FROM "/SampleFiles/InputSampleData.csv"

USING Extractors.Csv(skipFirstNRows:1);

@total = SELECT CourseCode.Substring(0,2) AS DeptCode, SUM(TotalStudents) AS
TotalNumberOfStudents
FROM @coursecode
GROUP BY CourseCode.Substring(0,2);

OUTPUT @total TO "/OutputFiles/StudentsCalculator.csv"
USING Outputters.Csv(outputHeader:true);
```

You can give a name to your job sol that you can keep track of it in future.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 11, November-2018

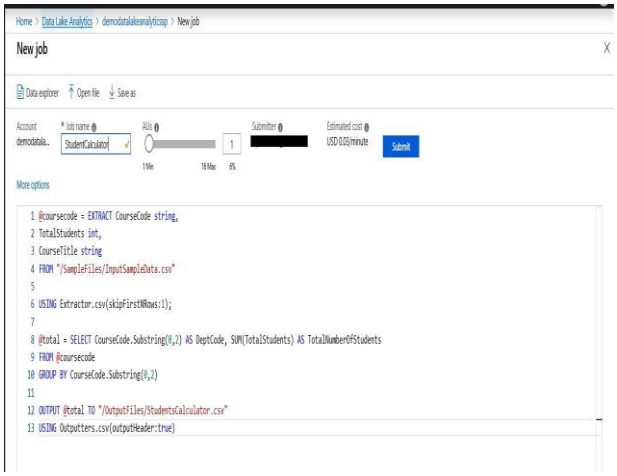

Fig. 9. Create a job with U-SQL commands

Once all the details are ready click on Submit. It will take few seconds to process the results. After every step has successfully executed you will see that all the steps are checked green.

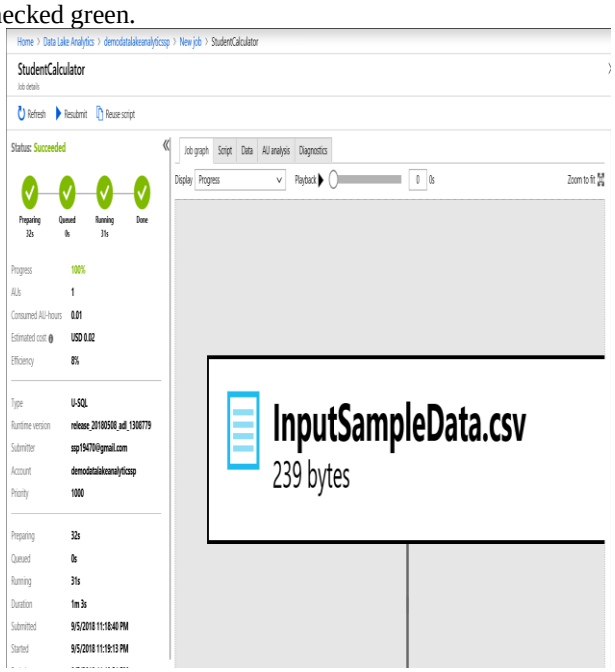

Fig. 10. Output after submission

The job graph will show you the details of the job where input file is transformed and finally converted to an output file StudentCalculator.csv.

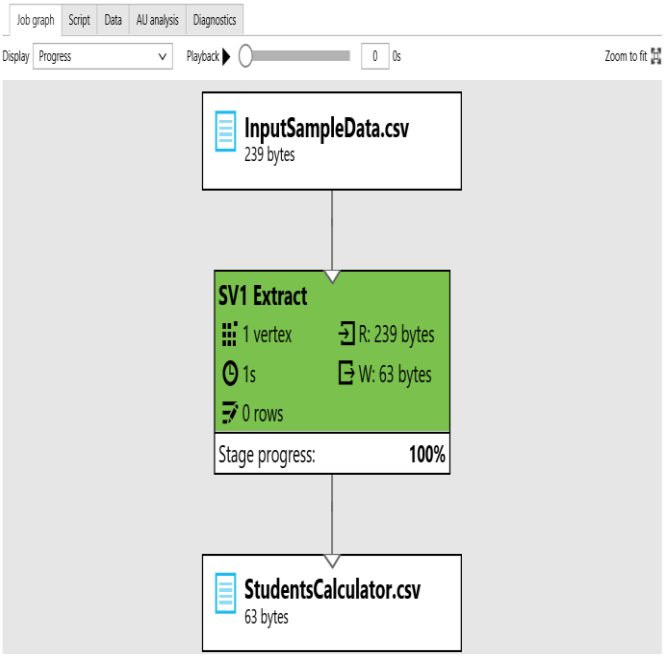

Fig. 11. Details of job run

Let's just click on the output file to see the results. You can also download the file for your edits using the download button in file preview page which is shown after you click the output file. Hurray! Our job has run successfully and we can see that the number of students has summed up as per our requirements. We got our results as 97 for CS course code as expected.

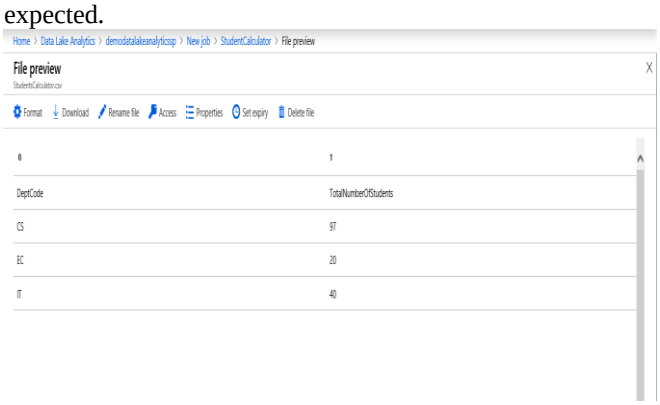

Fig. 11. Output file view

You can also navigate to the Data Lake Storage Gen1 account and see that a folder called OutputFiles is created and StudentsCalculator.csv is present in this folder.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 11, November-2018

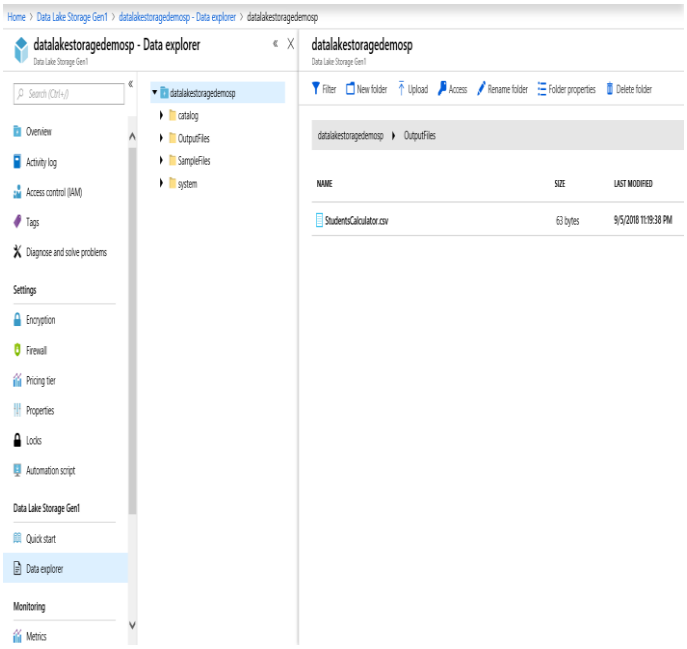

Fig. 12. Output file location in Data Lake Storage

VII. KEY HIGHLIGHTS OF USING AZURE DATA LAKE AND U-SQL OVER SSIS APPROACH

A. Easy simplified approach

As you can understand from the demo that the Data Lake approach has reduced a lot of development time as compared with SSIS approach for the given problem. We all know that SSIS is very established and feature rich product and there is a great development going on integrating SSIS with Azure. But, in case of SSIS solution for the problem we just solved, developer has to write scripts for extracting, transforming, loading the data to the database table and further write a SQL to aggregate the results. Creating the database and related database tables again takes some of developer's time. This is overall a very tedious task as compared to Data Lake + U-SQL approach where we just need to write one U-SQL script instead of performing too many tasks. This reduces overall development time and increases the productivity.

B. Scalability and reliability taken care by Azure Data Lake Store

Working on huge datasets might need more processing power and space. Azure Data lake provides easy and simple approach to ease the development and efficiently managing the data in HDFS (Hadoop Distributed File System) which is main building block of Azure Data Lake Storage. HDFS comes with a lot of benefits itself such as scalability, durability and replication.

C. Easy U-SQL scripts to rescue

Programmer needs to write a U-SQL scripts which makes ETL an easy and quick one- script job. U-SQL provides a very productive and programmer friendly environment by providing the ability to write the scripts using SQL and C# constructs.

REFERENCES

[1] Microsoft Azure Portal https://portal.azure.com
[2] https://docs.microsoft.com/en-us/azure/data-lake-analytics/data-lake-analytics-u-sql-get-started
[3] https://azure.microsoft.com/en-us/solutions/data-lake/