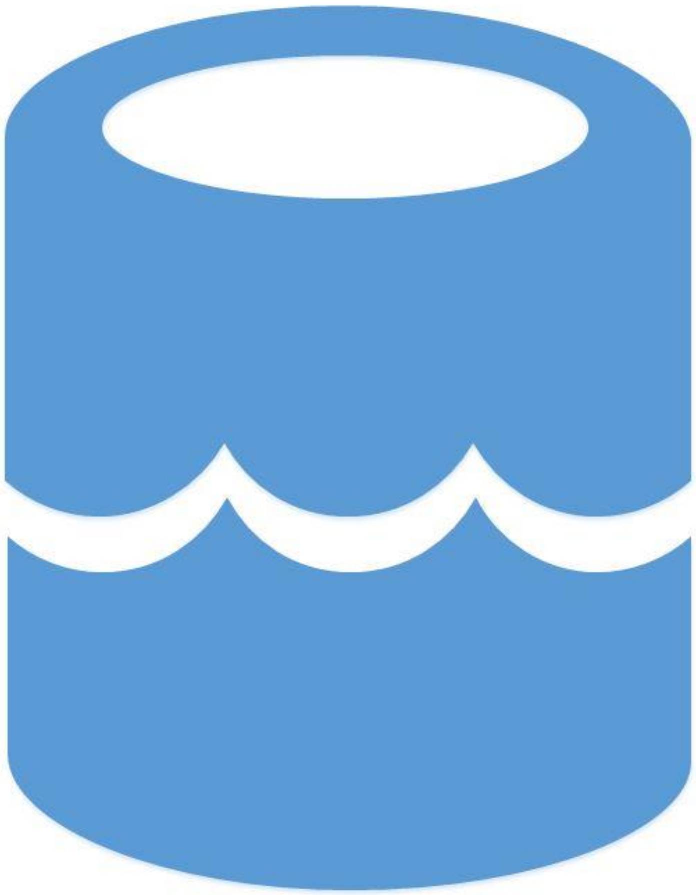
Microsoft

Cortana Intelligence Suite

Lab CIS003

Hyper-scale Repository with Azure Data Lake

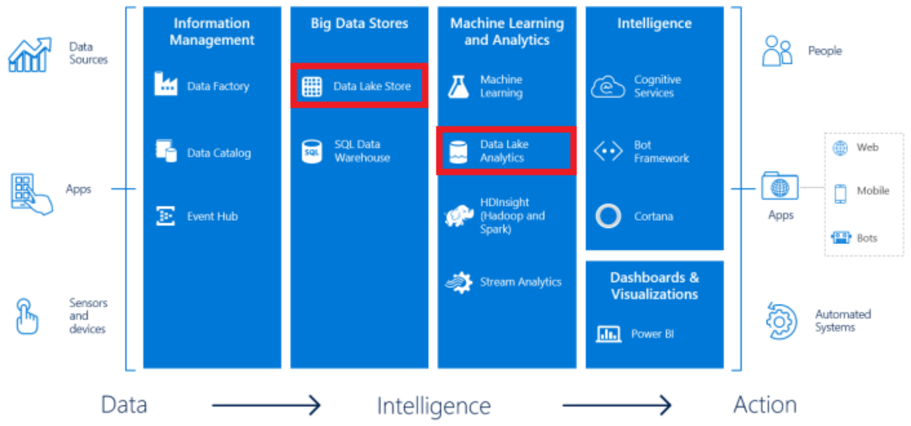


Contents

Overview

Summary

This lab explores processing big data with Azure Data Lake Store and Azure Data Lake Analytics. Azure Data Lake is a key component of the Cortana Intelligence Suite that allows the storage of relational and non-relational data. It also has its own query language that easily allows processing and analysis of data stored in any form.



In this lab, you will upload a file to Data Lake Store and process it using a combination of U-SQL jobs, table valued functions, and tables in the Catalog.

Business Case

Being able to monitor and analyze customer complaints can be key to understanding issues with products and gaps in customer service. If a product feature has a higher rate of complaints compared to other features, it may mean the feature is not performing as expected or is not desirable. This insight can lead to product revisions and/or recalls. In addition, understanding more about the timeliness and resolutions of complaints can lead to a more effective, customer-friendly support process.

In this lab, you will be using a subset of customer complaint data for financial products and services pulled from data.gov (http://catalog.data.gov/dataset/consumer-complaint-database). You will do some relatively straightforward manipulation of the raw data to create new files that aggregate the data to help show the timeliness of company responses to complaints as well as identify the companies with the most complaints.

Learning Objectives

Upon completing this lab, you will have hands-on experience with the following functions and concepts related to Azure Data Lake:

Data Lake Store

- Uploading files to the store
- Previewing files and their properties
- Browsing the data store and catalog

Data Lake Analytics

- Executing and Monitoring a U-SQL job
- Basic U-SQL language components like rowsets
- Extracting data from and outputting data to files stored in the Data Lake Store
- Creating and querying Table Valued Functions
- Creating and querying Tables

Lab Requirements/Prerequisites

- An Azure subscription is required to create and use the Azure Data Lake services. If you do not have an Azure Subscription, you can create one with a free credit at the following link:

  https://azure.microsoft.com/en-us/free/

- Files/data used in this lab can be downloaded here:

  https://github.com/CortanaAnalyticsLabs/CortanaAnalyticsLabs/raw/master/AzureDataLake/DataLakeLab.zip

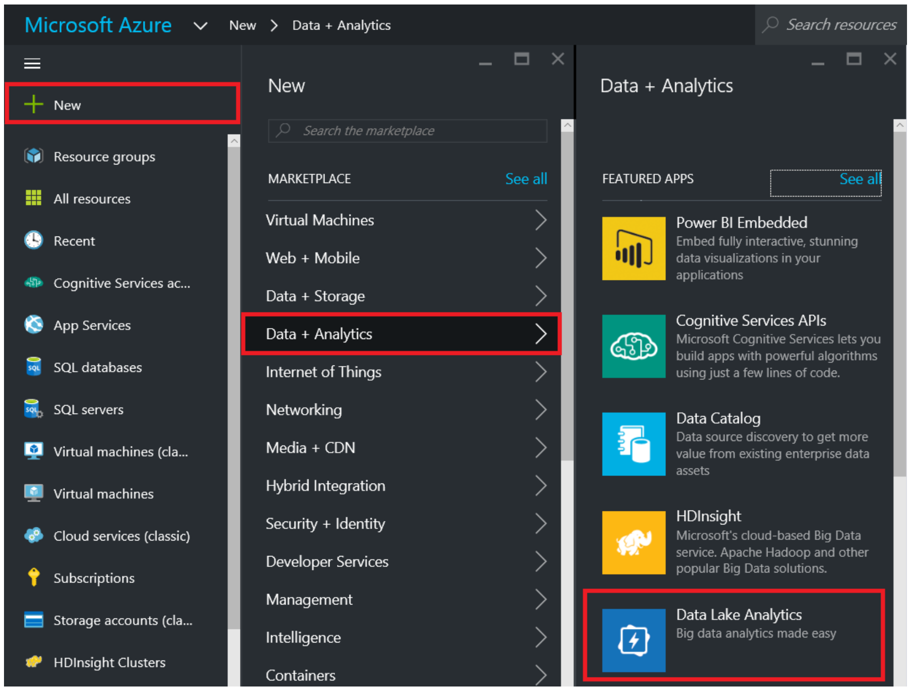
Create a Data Lake Analytics Account

Create a Data Lake Analytics Account

The instructions below walk you through creating a Data Lake Analytics account in the Azure Portal. Make sure you have logged into Azure before proceeding (http://portal.azure.com). If you already have a Data Lake Analytics account, you can skip this section and proceed to the next activity.

1. Click the **New** button on the left side of the portal.

2. Expand **Data + Analytics** and select **Data Lake Analytics**



Embed fully interactive, stunning data visualizations in your applications

Microsoft Cognitive Services lets you build apps with powerful algorithms using just a few lines of code.

Data source discovery to get more value from existing enterprise data assets

Microsoft's cloud-based Big Data service. Apache Hadoop and other popular Big Data solutions.

Big data analytics made easy

3. In the New Data Lake Analytics Account blade, fill out the required components as described below:

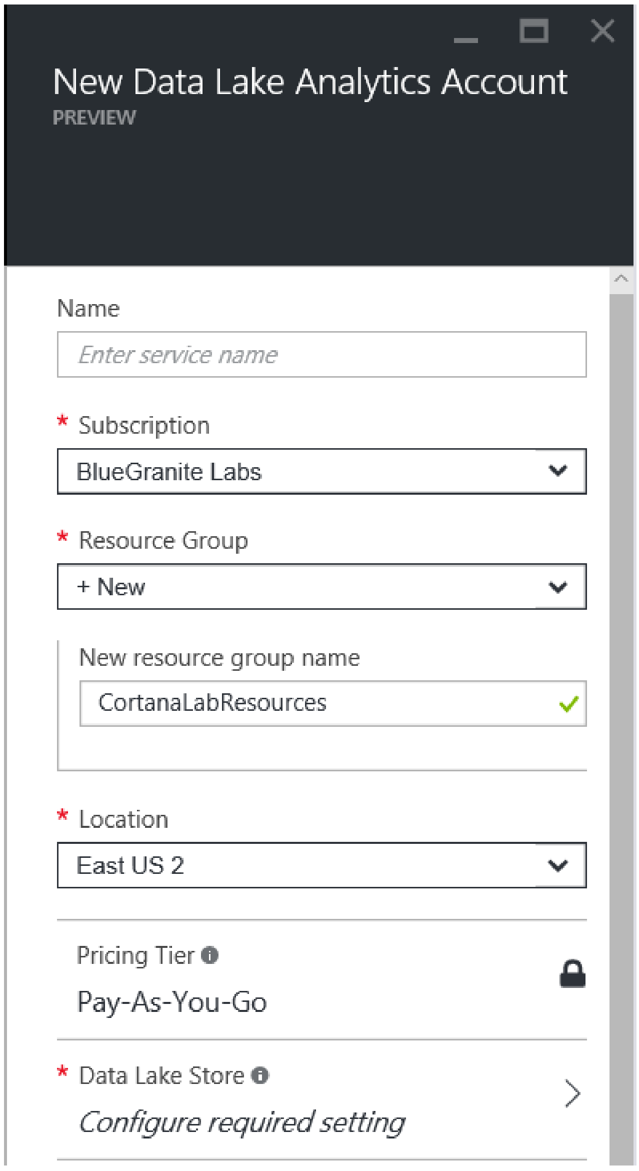
    a. **Name:** Name your Data Lake Analytics account

    b. **Subscription:** This is your Azure Subscription name

    c. **Resource Group:** Select an existing Azure Resource Group or create a new one

    d. **Location:** This is the geographic region of the Microsoft data center where the Data Lake Analytics service will run

    e. **Data Lake Store:** The Data Lake Analytics account requires a Storage Account to connect to. Click **Configure required setting** and follow the instructions to create a Data Lake Store.


New Data Lake Analytics Account

PREVIEW

Name: *Enter service name*

* Subscription: BlueGranite Labs

* Resource Group: + New

New resource group name: CortanaLabResources

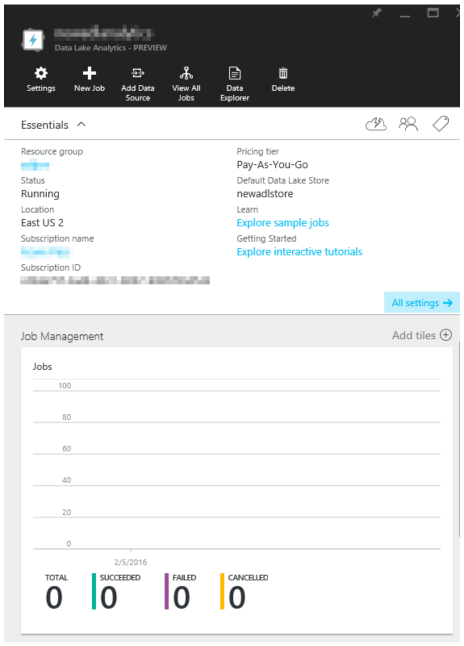
* Location: East US 2

Pricing Tier: Pay-As-You-Go

* Data Lake Store: *Configure required setting*

4. Make sure the **Pin to dashboard** checkbox is selected and click the **Create** Button.

   This will start the creation of your Data Lake Store and Analytics accounts. Once complete, the Data Lake Analytics home blade will open.



[redacted]
Data Lake Analytics - PREVIEW

Essentials

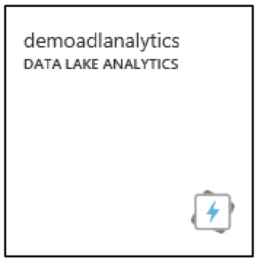
Resource group
[redacted]
Status
Running
Location
East US 2
Subscription name
[redacted]
Subscription ID
[redacted]

Pricing tier
Pay-As-You-Go
Default Data Lake Store
newadlstore
Learn
Explore sample jobs
Getting Started
Explore interactive tutorials

Job Management

TOTAL 0
SUCCEEDED 0
FAILED 0
CANCELLED 0

If you close the blade, you will also see a new tile on the Azure Portal for the Data Lake Analytics service.



demoadlanalytics
DATA LAKE ANALYTICS

Upload a File to Azure Data Lake Store

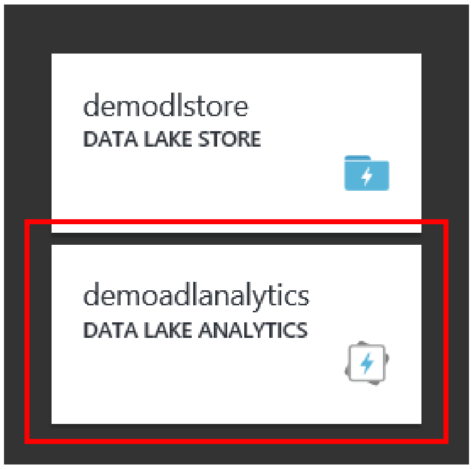
Access Azure Data Lake Analytics in the Azure Portal

In this activity, you will need to access the Data Lake Analytics user interface (UI) through the Azure Portal.

1. Open **Microsoft Edge** or another preferred browser and go to the **Azure Portal**: http://portal.azure.com.

2. If you are not automatically logged into the Azure Portal, sign in using the Microsoft Account that is associated with your Azure Subscription.

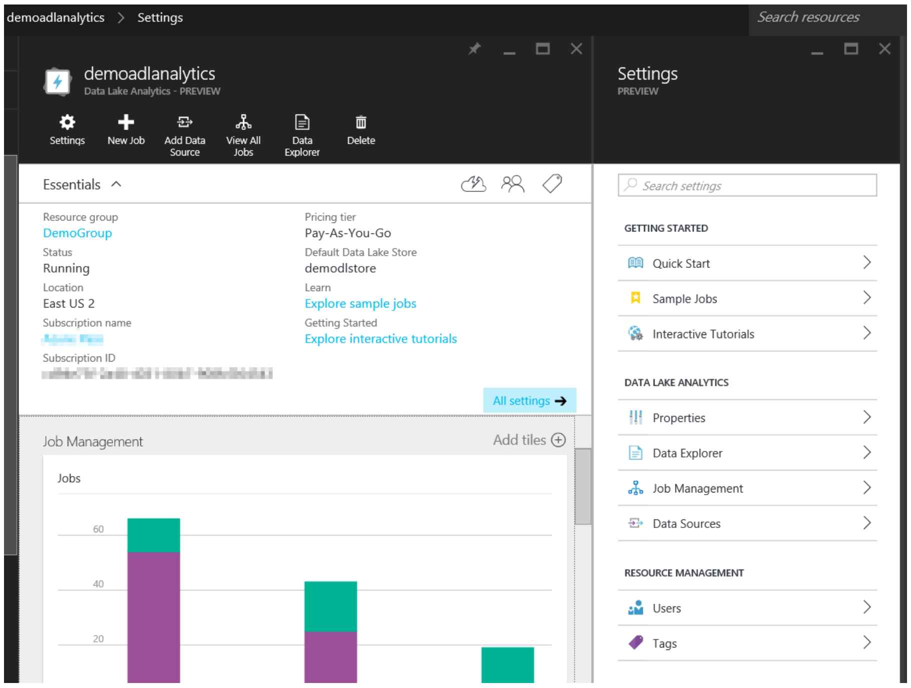
3. Click on the **Data Lake Analytics** tile from on the **Azure Portal Dashboard** to open the Data Lake Analytics UI.



demodlstore
DATA LAKE STORE

demoadlanalytics
DATA LAKE ANALYTICS

You should now be on the Data Lake Analytics home blade as seen below.



demoadlanalytics > Settings

Search resources

demoadlanalytics
Data Lake Analytics - PREVIEW

Settings | New Job | Add Data Source | View All Jobs | Data Explorer | Delete

Essentials

Resource group
DemoGroup
Status
Running
Location
East US 2
Subscription name
[redacted]
Subscription ID
[redacted]

Pricing tier
Pay-As-You-Go
Default Data Lake Store
demodlstore
Learn
Explore sample jobs
Getting Started
Explore interactive tutorials

All settings →

Job Management
Add tiles

Jobs

Settings
PREVIEW

Search settings

GETTING STARTED
Quick Start
Sample Jobs
Interactive Tutorials

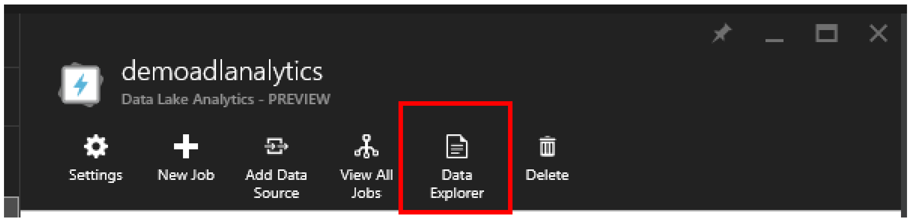
DATA LAKE ANALYTICS
Properties
Data Explorer
Job Management
Data Sources

RESOURCE MANAGEMENT
Users
Tags

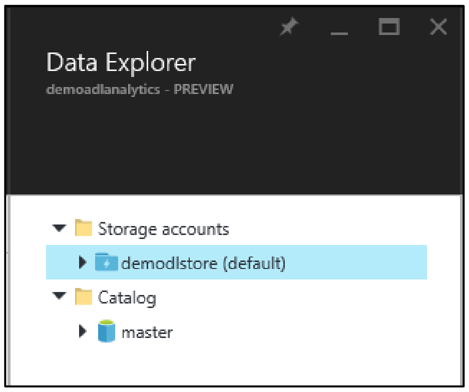
Create Input Directories and Upload an Input File

Next, you will create a series of directories for the input file that you will be processing.

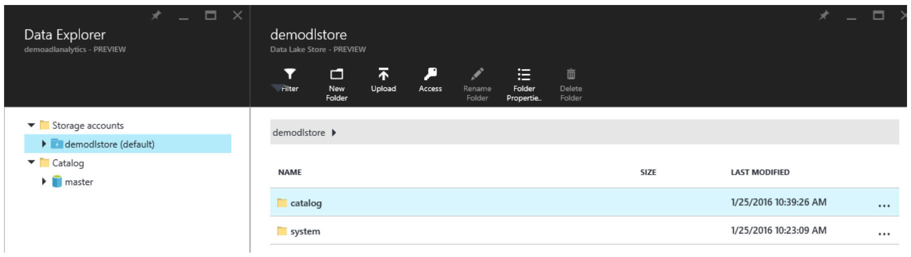
1. In the Data Lake Analytics blade, click the **Data Explorer** button.



This will bring up the *Data Explorer* blades. On the left is a blade where you can explore your *Storage Accounts* and *Catalog* in a navigation tree.

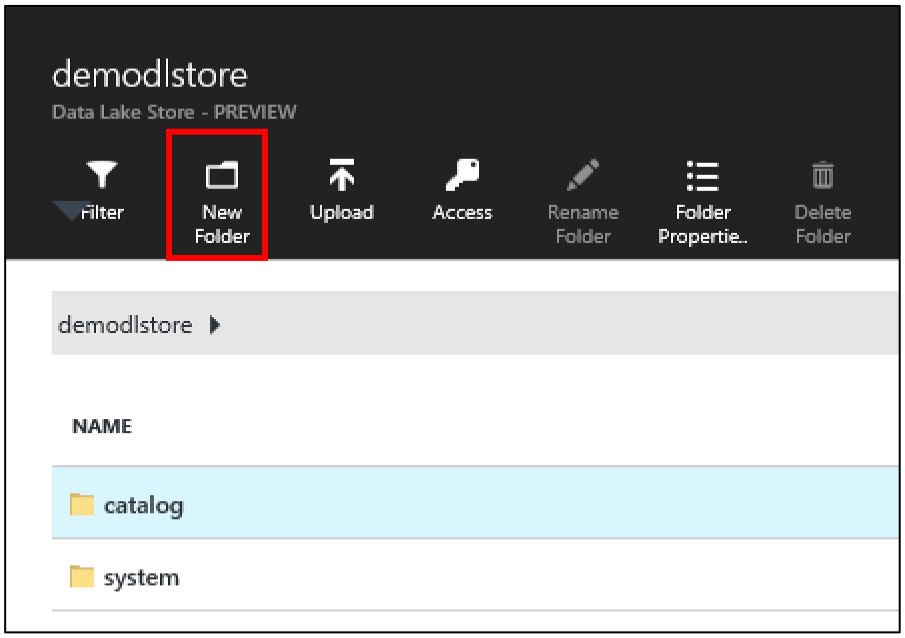


The blade on the right displays the contents and options for the selected Data Explorer object in the navigation blade.



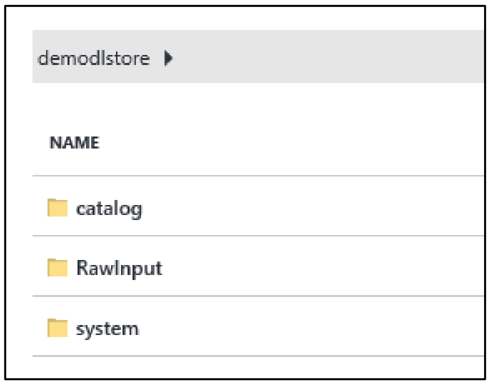
NAME	SIZE	LAST MODIFIED
catalog		1/25/2016 10:39:26 AM
system		1/25/2016 10:23:09 AM

2. Select your primary Data Lake Store account in the *Data Explorer* blade, then click the **New Folder** button near the top of the contents blade.



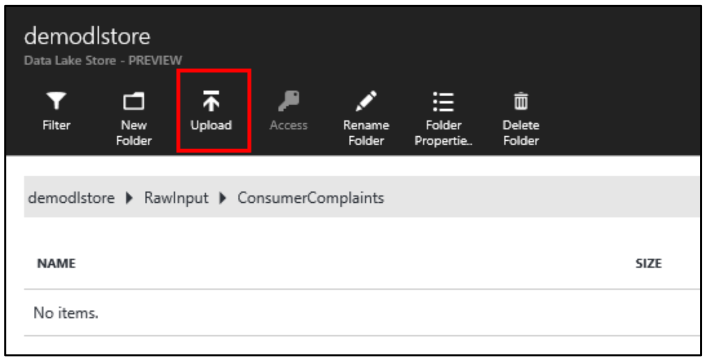
3. Enter the name ***RawInput*** in the *Create new folder* textbox, and click the **OK** button.

   Note that the new RawInput folder is created at the root of the Data Lake Store.

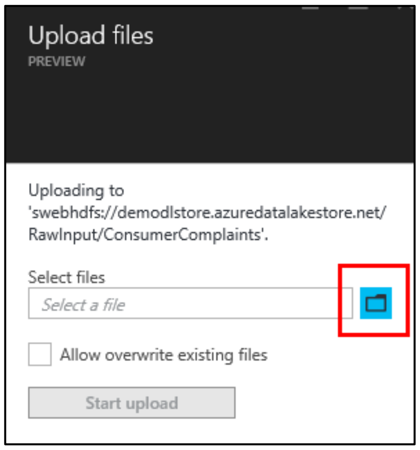


4. Click on the **RawInput** folder to open it and view the folder's contents.
5. Click the **New Folder** button again and create a folder named ***ConsumerComplaints*** within the *RawInput* folder.
6. Click the **ConsumerComplaints** folder to open it.

7. Click the **Upload** button near the top of the contents blade.



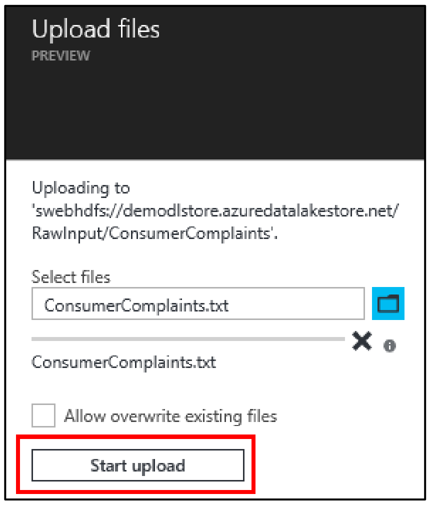
8. In the *Upload files* blade, click the **folder** icon to upload a file to the **ConsumerComplaints** folder.



Uploading to 'swebhdfs://demodlstore.azuredatalakestore.net/RawInput/ConsumerComplaints'.

Browse to the location where you downloaded the **ConsumerComplaints.txt** file, select the file, and click the **Open** button. If you have not yet obtained the lab materials, download and extract the zip file from the following link: https://github.com/CortanaAnalyticsLabs/CortanaAnalyticsLabs/raw/master/AzureDataLake/DataLakeLab.zip

9. Click the **Start upload** button on the **Upload files** blade.



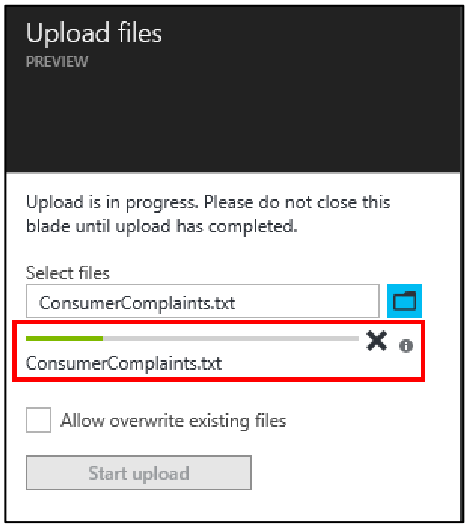
Upload files
PREVIEW

Uploading to 'swebhdfs://demodlstore.azuredatalakestore.net/RawInput/ConsumerComplaints'.

Select files
ConsumerComplaints.txt

ConsumerComplaints.txt

10. Wait for the file to finish uploading. You can monitor its status using the progress bar on the *Upload files* blade and/or in the *notification area* on the Azure Portal.



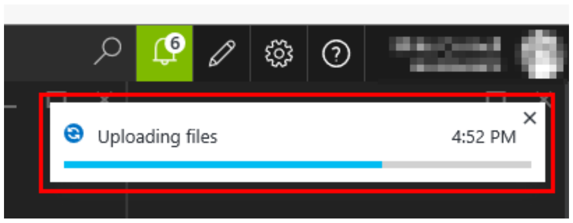
Upload files
PREVIEW

Upload is in progress. Please do not close this blade until upload has completed.

Select files
ConsumerComplaints.txt

ConsumerComplaints.txt

Until the file has uploaded successfully, you will see a progress bar and notification. Once the upload completes, a new notification will appear.



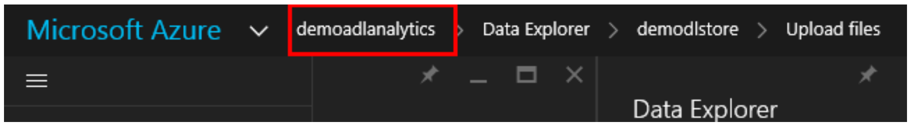
Uploading files 4:52 PM

Process Input File with U-SQL

Convert Tab Delimited File to Comma Separated

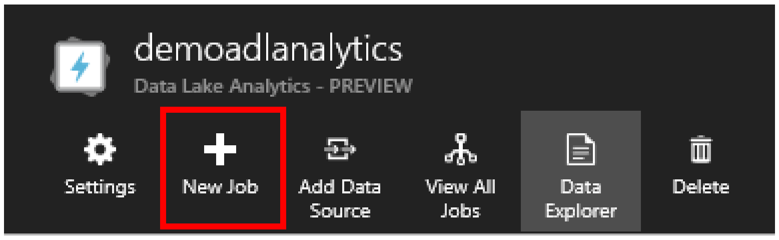
Next you will run your first U-SQL script that will convert the tab delimited ConsumerComplaints.txt file to a CSV file and output it to a different folder.

1. Click the **first** link in the blade's navigation bar at the top of the Azure portal. This link should correspond with the name and home blade for your Data Lake Analytics account.

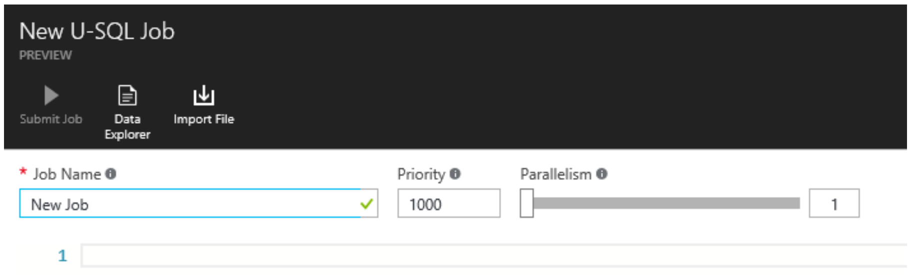
   

   This will focus the web browser back to the home blade.

2. Click the **New Job** button near the top of the blade

   Note that any other open blades will close, and a *New U-SQL Job* blade will display.

   Job Name: New Job
   Priority: 1000
   Parallelism: 1

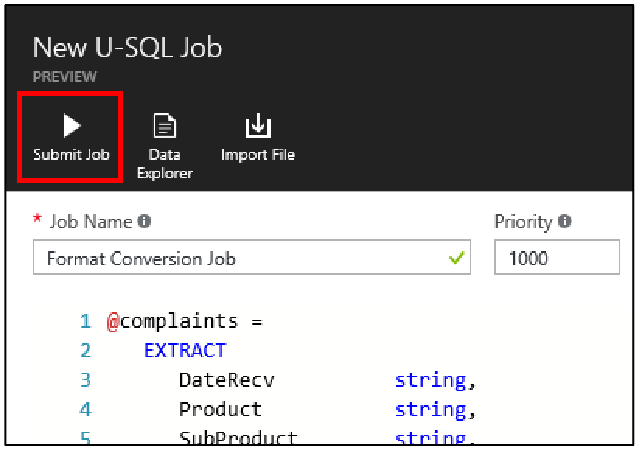
   On this blade, you can directly input U-SQL scripts and submit them as jobs with varying levels of Priority and Parallelism.

3. Change the text in the *Job Name* text box to **Format Conversion Job**.

4. Copy and paste the following code into the unlabeled *script editor* box that appears next to the number **1** on the **New U-SQL Job** blade.

```
@complaints =
   EXTRACT
      DateRecv                  string,
        Product                 string,
        SubProduct              string,
        Issue                   string,
        SubIssue                string,
        Narrative               string,
        ResponsePub             string,
        Company                 string,
        State                   string,
        Zip                     string,
        Src                     string,
        DateCompany             string,
        ResponseCust            string,
        Timely                  string,
        Disputed                string,
        Id                      string
   FROM
"/RawInput/ConsumerComplaints/ConsumerComplaints.txt"
   USING Extractors.Tsv();

   OUTPUT @complaints
   TO "/Processed/ConsumerComplaints/Complaints_All.csv"
   USING Outputters.Csv();
```

5. Click the **Submit Job** button near the top of the *New U-SQL Job* blade. This will submit the job and open a new blade with the Job Details**.** It may take a few minutes for the job to complete.

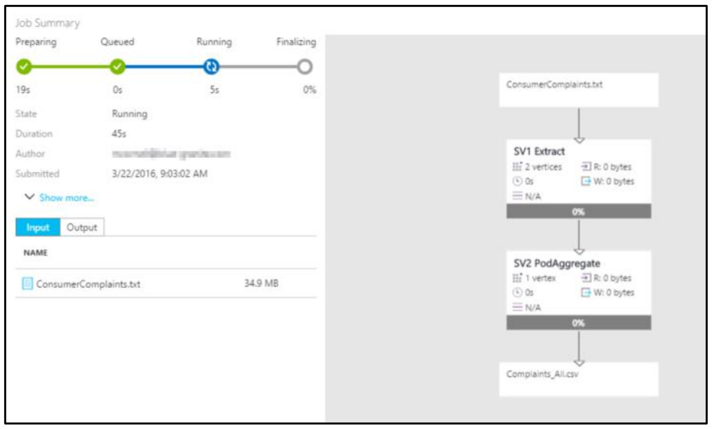


* Job Name: Format Conversion Job

Priority: 1000

```
@complaints =
   EXTRACT
      DateRecv           string,
      Product            string,
      SubProduct         string,
```

On the *Job Details* blade, you will see the status of the job change from **Preparing** to **Queued** to **Running** and then to **Finalizing** over the course of a couple of minutes.

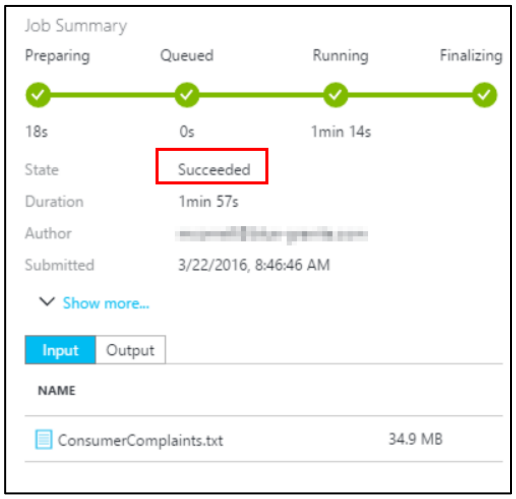


State	Running
Duration	45s
Author	[redacted]
Submitted	3/22/2016, 9:03:02 AM

NAME	
ConsumerComplaints.txt	34.9 MB

The code extracts the data from the ConsumerComplaints.txt file into the @complaints rowset variable, and then outputs the rowset to a csv file in a new set of folders (Processed/ConsumerComplaints).

Once the job completes, the blade will become static showing a state of **Succeeded**.



State	Succeeded
Duration	1min 57s
Author	[redacted]
Submitted	3/22/2016, 8:46:46 AM

NAME	
ConsumerComplaints.txt	34.9 MB

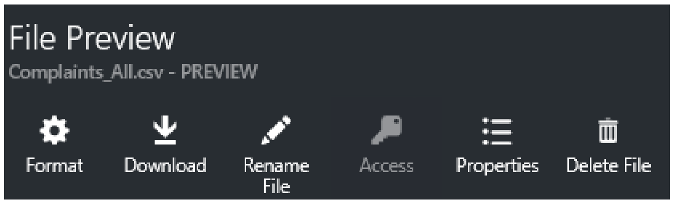
Note at the bottom that there are tabs displaying any Input and Output files from the job.

6. Click the **ConsumerComplaints.txt** file on the *Input* tab.

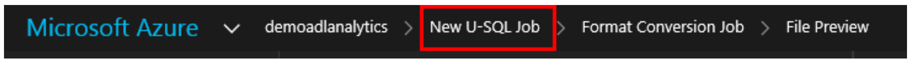
   Note that a *File Preview* blade opens to the right that shows a preview of the **ConsumerComplaints.txt** file contents.

7. Click the *Output* tab on the *Job Details* blade and open the **Complaints_All.csv** file.

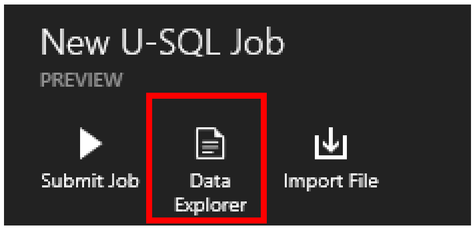
Note that a new *File Preview* blade will appear that previews the contents of the **Complaints_All.csv** file. Also, notice the different options to manage the file at the top of the *File Preview* blade.



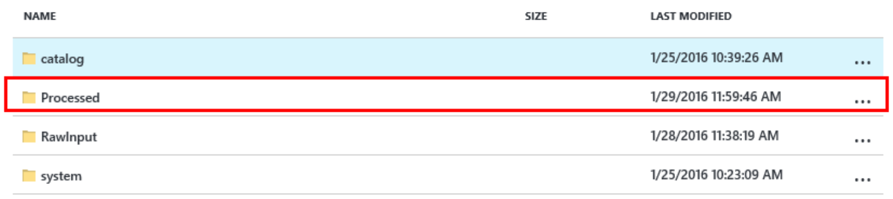
8. In the blade navigation bar at the top of the Azure portal, click on **New U-SQL Job** to return to that blade.



9. Click the **Data Explorer** button near the top of the *New U-SQL Job* blade.

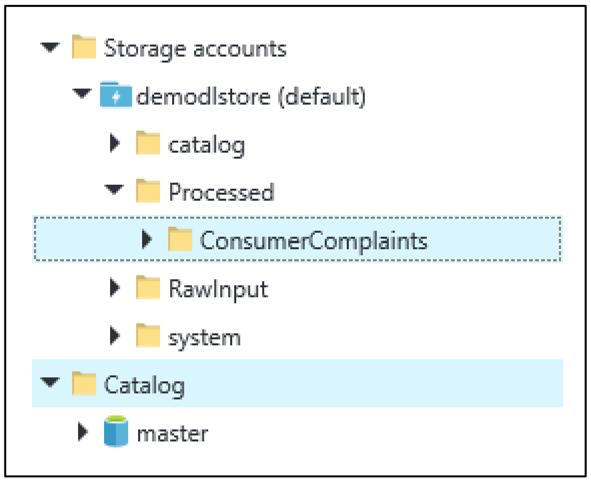


Notice in the *Data Lake Store* objects blade that a new **Processed** folder exists. This folder was created as a part of your U-SQL script.

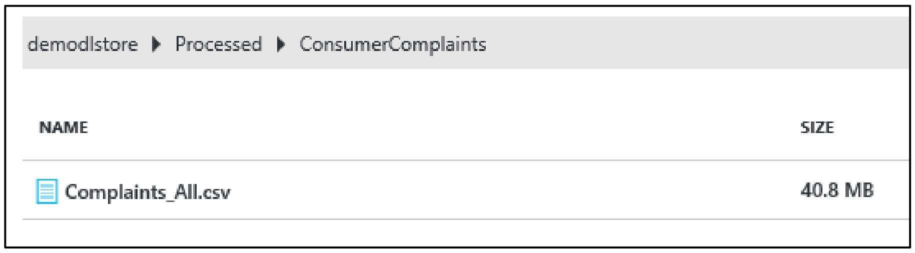


NAME	SIZE	LAST MODIFIED	
catalog		1/25/2016 10:39:26 AM	...
Processed		1/29/2016 11:59:46 AM	...
RawInput		1/28/2016 11:38:19 AM	...
system		1/25/2016 10:23:09 AM	...

10. Using the navigation tree in the *Data Explorer* blade, expand your default **Storage Account**; then expand the **Processed** folder. Finally, click the **ConsumerComplaints** folder.



Note that the **Complaints_All.csv** file that was created by the U-SQL job was output to the **ConsumerComplaints** folder.



NAME	SIZE
Complaints_All.csv	40.8 MB

Aggregate Values in the Input File

Next, you will execute a U-SQL script that processes the ConsumerComplaints.txt file and summarizes the data. This provides context to the timeliness of company response to complaints.

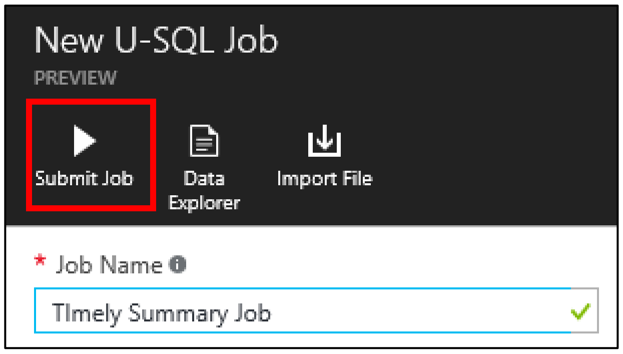
1. Navigate back to the *New U-SQL Job* blade by using the blade navigation bar near the top of the Azure Portal.

2. Remove any text that may appear in the *script editor* box on the *New U-SQL Job* blade, then copy and paste the following code.

```
@complaints =
   EXTRACT
      DateRecv          string,
        Product         string,
        SubProduct      string,
        Issue           string,
        SubIssue        string,
        Narrative       string,
        ResponsePub     string,
        Company         string,
        State           string,
        Zip             string,
        Src             string,
        DateCompany     string,
        ResponseCust    string,
        Timely          string,
        Disputed        string,
        Id              string
   FROM
"/RawInput/ConsumerComplaints/ConsumerComplaints.txt"
   USING Extractors.Tsv();

   @timeliness =
      SELECT DISTINCT
         Company
         ,Timely
         ,COUNT(Id) OVER(PARTITION BY Company, Timely)
AS TimelyComplaints
             ,COUNT(Id) OVER(PARTITION BY Company) AS
AllComplaints
      FROM @complaints;

OUTPUT @timeliness
TO
"/Processed/ConsumerComplaints/Complaints_Timeliness.txt
"
ORDER BY Company
USING Outputters.Tsv();
```

3. Change the *Job Name* to **Timely Summary Job**.

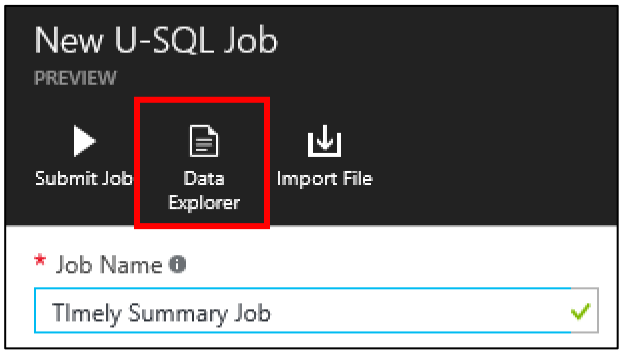
4. Click the **Submit Job** button.



* Job Name
TImely Summary Job

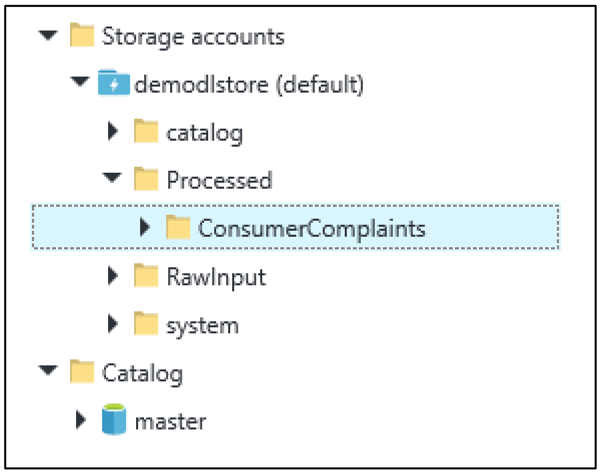
This code extracts the data from the ConsumerComplaints.txt file into the @complaints rowset variable. It then summarizes the data by Company and Timely, and it outputs the results to a tab delimited file in the /Processed/ConsumerComplaints folder.

5. Once the job completes, navigate back to the *New U-SQL Job* blade and click the **Data Explorer** button.

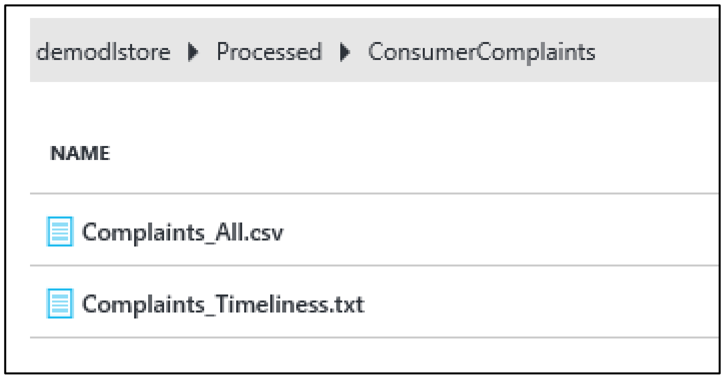


* Job Name: TImely Summary Job

6. Once the *Data Explorer* blade opens, browse to the **Processed/ConsumerComplaints** folder in the default Storage Account.



Note that there is a new file in the folder called **Complaints_Timeliness.txt**.

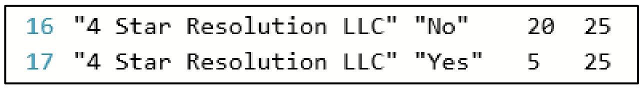


NAME
Complaints_All.csv
Complaints_Timeliness.txt

7. Click the **Complaints_Timeliness.txt** file to preview it.

   The file displayed in the *File Preview* blade lists each Company, a timeliness indicator, the number of complaints for the indicator-company, and the total number of complaints for the company overall.

   As an example, you should see that the company *"4 Star Resolution LLC"* has provided timely responses for only 5 of their 25 total complaints.



```
"4 Star Resolution LLC" "No"    20  25
"4 Star Resolution LLC" "Yes"   5   25
```

Create a Table Valued Function

If you look back at the previous two U-SQL scripts, you will notice that both scripts use the same code to extract data from the input file into the rowset *@complaints*. When you have a block of code you want to reuse, it may make sense to manage that code as a Table Valued Function. These named functions are created in the Data Lake Analytics Catalog and can be called from U-SQL scripts. In the section below, we will create a Table Valued Function from the code that built the *@complaints* rowset in previous scripts.

1. Navigate back to the **New U-SQL Job** blade by using the blade navigation bar near the top of the Azure Portal.

2. Change the *Job Name* to **Create Function Job*.***

3. Copy and paste the following code into the script editor on the *New U-SQL Job* blade.

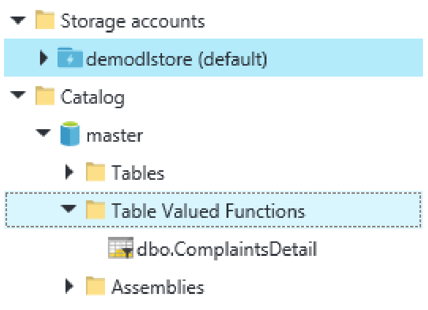
```
DROP FUNCTION IF EXISTS ComplaintsDetail;

CREATE FUNCTION ComplaintsDetail()
  RETURNS @complaints TABLE
  (
     DateRecv       string,
     Product        string,
     SubProduct     string,
     Issue          string,
     SubIssue       string,
     Narrative      string,
     ResponsePub    string,
     Company        string,
     State          string,
     Zip            string,
     Src            string,
     DateCompany    string,
     ResponseCust   string,
     Timely         string,
     Disputed       string,
     Id             string
  )
  AS BEGIN

 @complaints =
   EXTRACT
      DateRecv                string,
        Product               string,
        SubProduct            string,
        Issue                 string,
        SubIssue              string,
        Narrative             string,
        ResponsePub           string,
        Company               string,
        State                 string,
        Zip                   string,
        Src                   string,
        DateCompany           string,
        ResponseCust          string,
        Timely                string,
        Disputed              string,
        Id                    string
   FROM
"/RawInput/ConsumerComplaints/ConsumerComplaints.txt"
   USING Extractors.Tsv();

   RETURN;
END;
```

4. Click the **Submit Job** button.

   This code creates a Table Valued Function called **ComplaintsDetail** that will return the @Complaints rowset from the ConsumerComplaints.txt file.

5. Once the job finishes executing, navigate back to the *New U-SQL Job* blade and click the **Data Explorer** button.

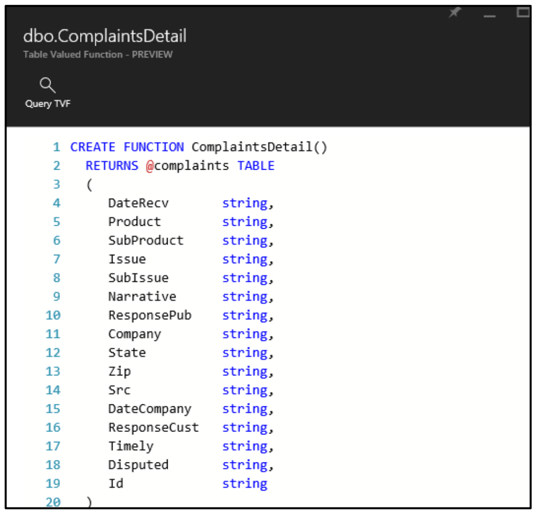
6. On the *Data Explorer* blade, under **Catalog**, expand **master** and then **Table Valued Functions**.



Notice the function you just created called *dbo.ComplaintsDetail.*

7. Click the **dbo.ComplaintsDetail** function.

Note the new blade that displays the function's code.



```
CREATE FUNCTION ComplaintsDetail()
  RETURNS @complaints TABLE
  (
     DateRecv       string,
     Product        string,
     SubProduct     string,
     Issue          string,
     SubIssue       string,
     Narrative      string,
     ResponsePub    string,
     Company        string,
     State          string,
     Zip            string,
     Src            string,
     DateCompany    string,
     ResponseCust   string,
     Timely         string,
     Disputed       string,
     Id             string
  )
```

Remember that no data is physically stored in the function—just code that will return a rowset.

Create a Table

Next, you will create a new database and table in the Catalog. Unlike functions, tables actually store data. Tables are great for storing schematized data for a faster, more familiar retrieval.

1. Navigate back to the *New U-SQL Job* blade by using the blade navigation bar near the top of the Azure Portal.

2. Change the *Job Name* to **Create Table Job**.

3. Copy and paste the following code into the script editor on the *New U-SQL Job* blade.

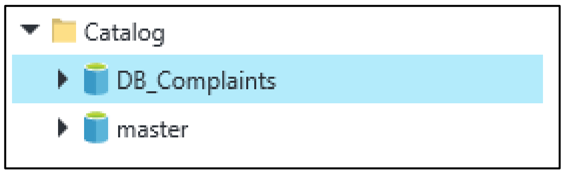
```
DROP DATABASE IF EXISTS DB_Complaints;
CREATE DATABASE DB_Complaints;

USE DATABASE DB_Complaints;

DROP TABLE IF EXISTS ConsumerComplaints;

CREATE TABLE ConsumerComplaints
(
   INDEX idx1 CLUSTERED (Id ASC)
   PARTITIONED BY HASH (Company)
) AS
SELECT
     DateTime.Parse(DateRecv) AS DateRecv
     ,Product
     ,SubProduct
     ,Issue
     ,SubIssue
     ,Narrative
     ,ResponsePub
     ,Company
     ,State
     ,Zip
     ,Src
     ,DateTime.Parse(DateCompany) AS DateCompany
     ,ResponseCust
     ,Timely
     ,Disputed
     ,int.Parse(Id) AS Id
FROM master.dbo.ComplaintsDetail() AS Detail
WHERE Id != "Complaint ID";
```

4. Click the **Submit Job** button.

   This job will take longer than previous ones to finish executing (approximately ten minutes). It creates a database named *DB_Complaints*. It then creates a table in *DB_Complaints* called *ConsumerComplaints* and populates the table with the results of a query against the *ComplaintsDetail()* function that you created previously.

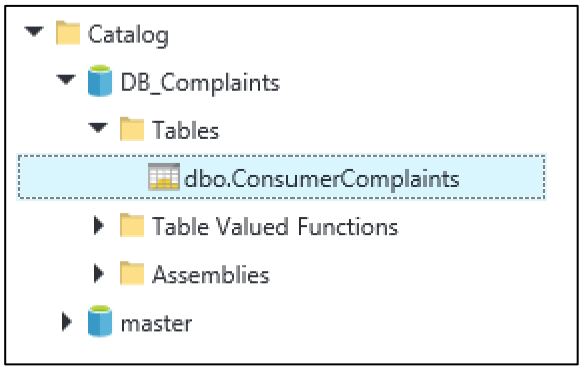
5. Once the job finishes executing, navigate back to the *New U-SQL Job* blade and click the **Data Explorer** button.

   Notice on the *Data Explorer* blade that there is a new database called **DB_Complaints**.

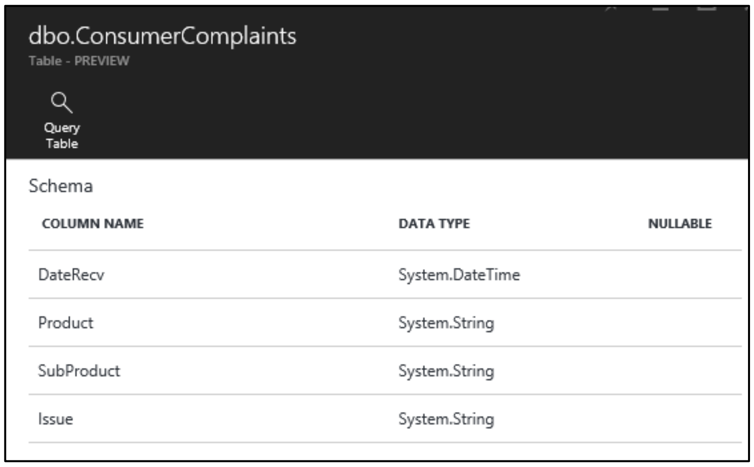
6. On the *Data Explorer* blade, under **Catalog**, expand **DB_Complaints** and then **Tables**.

   Note the new **dbo.ConsumerComplaints** table that was created from the U-SQL script that you executed.

7. Click on the **dbo.ConsumerComplaints** table.

   Note the blade that opens and shows the schema and indexes in the table.

Schema

COLUMN NAME	DATA TYPE	NULLABLE
DateRecv	System.DateTime	
Product	System.String	
SubProduct	System.String	
Issue	System.String	

Query Data in a Table Using Multiple Files in a Single Job

Finally, you will execute a U-SQL script that queries the table you created and outputs a series of files.

1. Navigate back to the *New U-SQL Job* blade by using the blade navigation bar near the top of the Azure Portal.

2. Change the *Job Name* to **Most Complaints Job**.

3. Copy and paste the following code into the script editor on the *New U-SQL Job* blade.

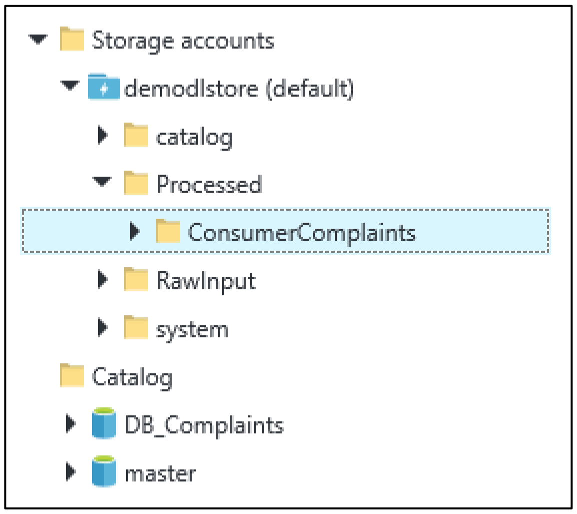
```
@ByCompany =
    SELECT Company,
           COUNT(Id) AS TotalComplaints
    FROM [DB_Complaints].[dbo].[ConsumerComplaints]
    GROUP BY Company;

@MostComplaints =
    SELECT Company,
           COUNT(Id) AS TotalComplaints
    FROM [DB_Complaints].[dbo].[ConsumerComplaints]
    GROUP BY Company
    ORDER BY COUNT(Id) DESC
    FETCH 10 ROWS;


OUTPUT @ByCompany
    TO
"/Processed/ConsumerComplaints/ComplaintsByCompany.csv"
    USING Outputters.Csv();

OUTPUT @MostComplaints
    TO
"/Processed/ConsumerComplaints/Top10Companies.txt"
    USING Outputters.Tsv();
```

4. Click the **Submit Job** button.

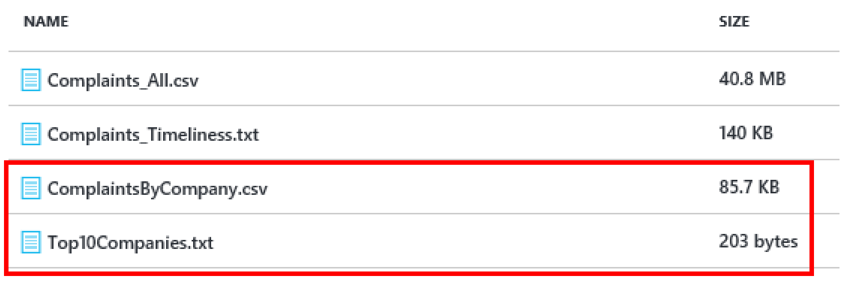
   This code executes two SQL queries against the **ConsumerComplaints** table. The first query *(@ByCompany)* creates a rowset containing the total number of complaints for each company. The second query *(@MostComplaints)* creates a rowset containing the top ten companies by complaints along with their counts. Each rowset is output to a file in the **Processed/ConsumerComplaints** folder.

5. Once the job finishes executing, navigate back to the *New U-SQL Job* blade and click the **Data Explorer** button.

6. Once the *Data Explorer* blade opens, browse to the **Processed/ConsumerComplaints** folder in the default Storage Account.



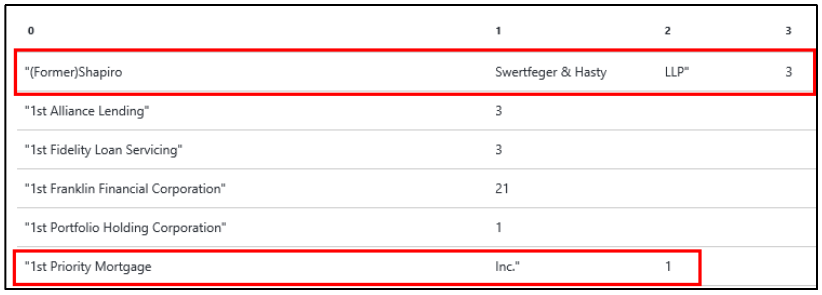
Notice there are two new files in the **ConsumerComplaints** folder that were created by the U-SQL script you executed.



NAME	SIZE
Complaints_All.csv	40.8 MB
Complaints_Timeliness.txt	140 KB
ComplaintsByCompany.csv	85.7 KB
Top10Companies.txt	203 bytes

7. Click the file named **ComplaintsByCompany.csv** to preview it.

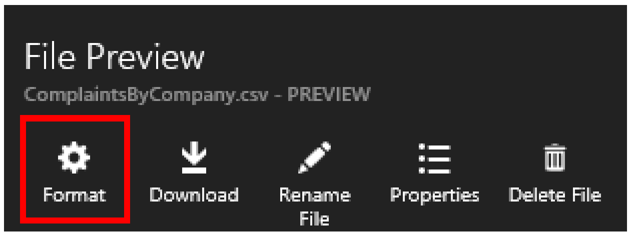
The file should have one column for **company** and another for **count**. Note that in some cases, the preview displays part of the company name in additional columns.



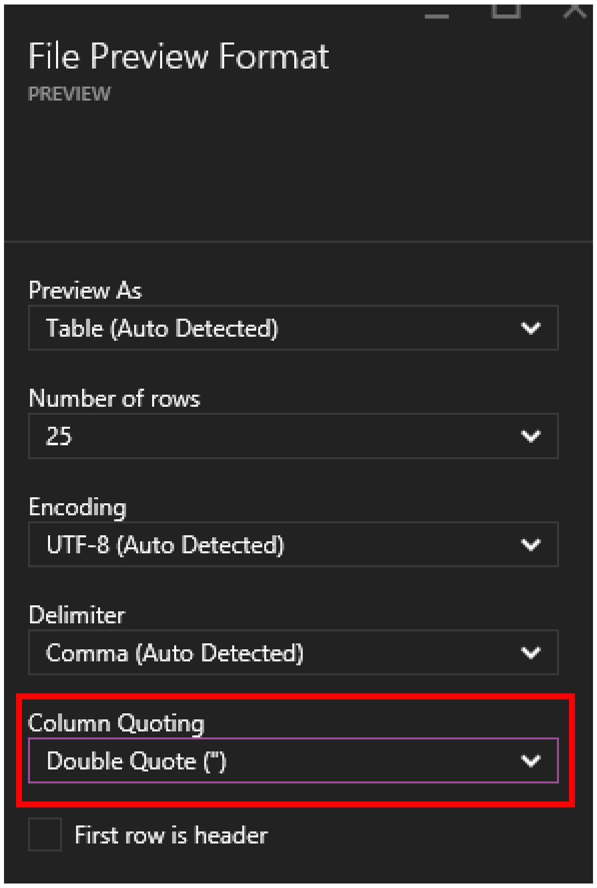
0	1	2	3
"(Former)Shapiro	Swertfeger & Hasty	LLP"	3
"1st Alliance Lending"	3		
"1st Fidelity Loan Servicing"	3		
"1st Franklin Financial Corporation"	21		
"1st Portfolio Holding Corporation"	1		
"1st Priority Mortgage	Inc."	1	

This is because the file is comma-separated, and some company names have commas in them. You can tell the previewer to ignore those columns using the *Format* button.

8. Click the **Format** button.


File Preview
ComplaintsByCompany.csv - PREVIEW

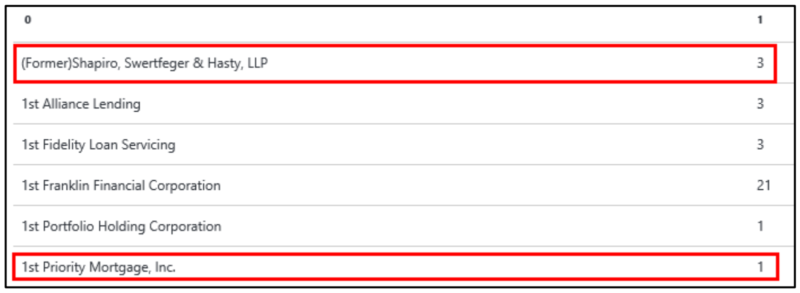
9. In the *File Preview Format* blade, change the *Column Quoting* drop-down box to **Double Quote(")**.


File Preview Format
PREVIEW

Preview As: Table (Auto Detected)
Number of rows: 25
Encoding: UTF-8 (Auto Detected)
Delimiter: Comma (Auto Detected)
Column Quoting: Double Quote (")
First row is header

10. Click the **OK** button near the bottom of the *File Preview Format* blade.

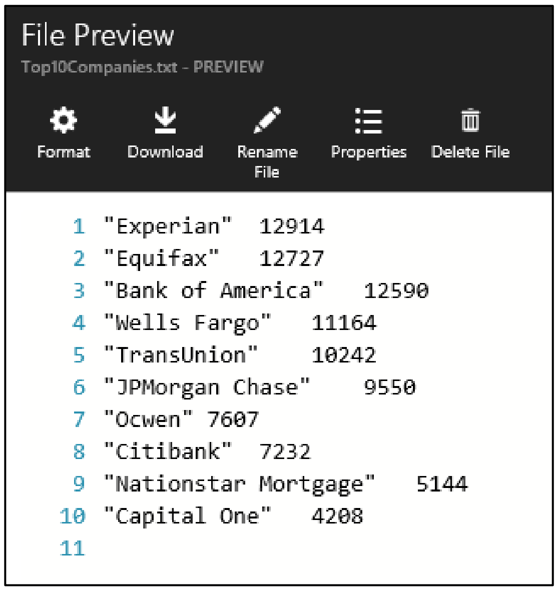
Notice that the *File Preview* blade now correctly shows the company and counts columns.



0	1
(Former)Shapiro, Swertfeger & Hasty, LLP	3
1st Alliance Lending	3
1st Fidelity Loan Servicing	3
1st Franklin Financial Corporation	21
1st Portfolio Holding Corporation	1
1st Priority Mortgage, Inc.	1

11. Navigate back to the *Data Explorer* blade and click the file named **Top10Companies.txt**.

    Notice the File Preview blade displays the names and counts of the top 10 companies ordered by number of complaints.



```
"Experian"	12914
"Equifax"	12727
"Bank of America"	12590
"Wells Fargo"	11164
"TransUnion"	10242
"JPMorgan Chase"	9550
"Ocwen"	7607
"Citibank"	7232
"Nationstar Mortgage"	5144
"Capital One"	4208
```

Conclusion

This concludes the *Processing Big Data with Azure Data Lake* lab. You have successfully uploaded the complaints file to your Data Lake Store and processed the file multiple times using Data Lake Analytics. Through a combination of U-SQL jobs, Table Valued Functions, and Tables; the data now resides in a format ready for additional consumption, transformation, or visualization.

To expand on what you learned, some suggested paths include:

- Continue to process and aggregate the ConsumerComplaints.txt file using other fields and U-SQL functions
- Utilize the built-in C# libraries to create complex functions to parse the Issue and Sub-issue fields and perform more advanced text analytics
- Re-run the scripts after changing the Parallelism option on the *New U-SQL Job* blade to see how higher degrees of parallelism impact query performance

For more detail on Azure Data Lake, see the Azure documentation:

Data Lake Analytics
https://azure.microsoft.com/documentation/services/data-lake-analytics/

Data Lake Store
https://azure.microsoft.com/documentation/services/data-lake-store/

Terms of Use