Procesando Big Data con Azure Data Lake

Lab 1 - Introduciendonos en Azure Data Lake

Introducción

En esta práctica, creará una cuenta de Azure Data Lake Analytics y un de Azure Data Lake store asociado a esta cuenta. A continuación, cargará algunos archivos de datos en Azure Data Lake store y creará un trabajo U-SQL para procesar datos.

Prerrequisitos

Para completar esta práctica, usted necesita

- Un web browser
- Un Microsoft account
- Un Microsoft Azure subscription
- Un ordenador Windows, Linux, o Mac OS X

Creando un Azure Data Lake Analytics Account

En este ejercicio, creará una cuenta de Azure Data Lake Analytics y un de Azure Data Lake store asociado a esta cuenta.

Creando un Azure Data Lake Analytics Account

Antes de poder utilizar Azure Data Lake Analytics para procesar datos, debe crear una cuenta de Azure Data Lake Analytics, y asóciarla con al menos un Azure Data Lake store.

1. En un navegador web, vaya a http://portal.azure.com y, si se le solicita, inicie sesión con la Cuenta de Microsoft que está asociada con su suscripción de Azure.
2. En el portal de Microsoft Azure, en el Hub Menú, haga clic en New. Luego, en Intelligence and analytics menú, haga clic en Data Lake Analytics.
3. En New Data Lake Analytics Account, ingrese la siguiente configuración, y luego haga clic Create:
   - Name: ingrese un nombre único (¡y anótelo!)
   - Subscription: seleccione su suscripción de Azure
   - Resource Group: cree un nuevo grupo de recursos con un nombre único
   - Location: seleccione cualquier región disponible
   - Data Lake Store: cree una nueva tienda Data Lake con un nombre único (y anotelo)
   - Pin to dashboard: no seleccionado
4. En Azure Portal, vea Notifications para verificar que la implementación haya comenzado y espere a que los recursos sean creados (esto puede tomar unos minutos).

Explorando el Data Lake Store

El Azure Data Lake store se usará para almacenar datos, scripts y bases de datos que serán usados por el Azure Data Lake Analytics account.

1. En el portal de Microsoft Azure, vaya a Azure Data Lake Store.

2. En la página Overview, tenga en cuenta la utilización de almacenamiento actual (que debe ser de 0 bytes).
3. Vea la página Data Explorer, que muestra los contenidos actuales de su storage account. Debe contener dos carpetas (catalog y system). Más tarde, usaremos esta herramienta para cargar y descargar datos en el store.

Explorando el Data Lake Analytics Account

1. En el portal de Microsoft Azure, vaya a Azure Data Lake Analytics account.
2. En la página Overview puede revisar la información disponible
3. Vea la página Data Explorer, que muestra los contenidos actuales de su storage account. Debe contener dos carpetas (catalog y system). Más tarde, usaremos esta herramienta para cargar y descargar datos en el store.

Ejecutando Azure Data Lake Analytics Jobs

En este ejercicio, ejecutara un job siomple para procesar un web server log

Provisionando un Source data file

En este lab, usara Azure Data Lake Analytics para procesar web server logs. Inicialmente ejecutara algunos Jobs para procesar un solo archivo

1. En el folder donde extrajo los archivos de esta práctica, expandir el iislogs folder.
2. Use un editor de texto para revisar el contenido del file 2008-01.txt, el file contiene una fila que tiene como prefijo # para indicar que esta es una fila de cabecera, y los demás registros son web server requests delimitadas por espacios.
3. En el Azure portal, vaya a la página Data Explorer del Azure Data Analytics account, debe crear un nuevo folder llamdo iislogs en el root del Azure Data Lake store.
4. Expanda el folder iislogs, luego click en Upload y carge el archivo 2008-01.txt

Creando un nuevo Job

Una vez que tenemos listo el data source en el Azure Data Lake store, podemos usar U-SQL query para leer y procesar datos.

1. En el Azure portal, en el blade para su Azure Data Lake Analytics account, click en New Job
2. En el blade New U-SQL job, introduzca **Read Log File** como nombre del job en el Job Name box, luego en la ventana de código escriba lo siguiente:

```
@log = EXTRACT entry string
FROM "/iislogs/2008-01.txt"
USING Extractors.Text();

OUTPUT @log
TO "/output/log.txt"
USING Outputters.Text();
```

Este código usa el **Text** extractor para leer el contenido del archivo **2008-01.txt.** El delimitador de campos por defecto para el Text extrator es una coma, que los datos no contienen, por lo tanto, lee cada línea en el archivo y usa el defaul **Text outputter** para escribir los resultados en un output file.

3. Click Submit Job y observe los detalles del job, una vez que el job está listo para ejecutarse un gráfico desplegara los pasos a ejecutar.
4. Cuando el job finalize, click Output tab y seleccione log.text para previsualizar los resultados, que son los mismos del data source origen.

Escribiendo un SELECT para filtrar datos

El job anterior no hace nada significativo, simplemente lee los datos de un archivo. Ahora modificaremos el job para filtrar datos, removiendo el row de cabecera

1. En el New SQL Job blade, modificar la consulta anterior tal como se muestra a continuación

```
@log = EXTRACT entry string
     FROM "/iislogs/2008-01.txt"
USING Extractors.Text();

@cleaned = SELECT entry
          FROM @log
          WHERE entry.Substring(0,1) != "#";

OUTPUT @cleaned
TO "/output/cleaned.txt"
USING Outputters.Text();
```

Este query usa un SELECT para filtrar los datos recuperados con el Extractor. La instrucción WHERE usa el método Substring de C# para filtrar filas que comienzan con el carácter #. La habilidad de enlazar C# and SQL es lo que hace a U-SQL un lenguaje flexible y extensible para procesar datos.

2. Click en **Submit Job**, y monitoree los detalles de ejecución
3. Cuando el job haya terminado, click Output tab y seleccione cleaned.txt para ver los resultados, que ahora están libres de las filas de encabezado. Los resultados se muestran en formato tabla, detectando espacios como delimitadores, sin embargo, el resultado es enteramente texto plano.

Aplicando schema a los datos

Hasta el momento hemos usado U-SQL para leer y filtrar texto basados en las filas del texto. Ahora aplicaremos esquema a los datos, separando los datos en campos discretos que pueden ser procesados individualmente

1. En el Azure portal, en el blade del Azure Data Lake Analytics account, click en la página New Job.
2. En el blade New U-SQL Job, escriba **Process Log Entries** como nombre para el nuevo job, y en la ventana de edición de codigo escriba

```
@log = EXTRACT date string,
     time string,
     client_ip string,
     username string,
     server_ip string,
     port int,
     method string,
     stem string,
     query string,
     status string,
     server_bytes int,
     client_bytes int,
     time_taken int,
     user_agent string,
     referrer string
FROM "/iislogs/2008-01.txt"
USING Extractors.Text(' ', silent:true);

OUTPUT @log
TO "/output/log.csv"
USING Outputters.Csv();
```

3. Click **Submit Job** y monitoree los detalles de ejecución.
4. Cuando el job haya completado su ejecución, click en el **Output** tab y seleccione log.csv para visualizar los resultados
5. Descargue el archivo resultado y ábralo en un editor de texto o en un spreadsheet y note que cada fila contiene múltiples campos.

Creando datos agregados (Aggregate data)

Ahora que hemos aplicado un esquema a los datos, podemos escribir consultas (queries) haciendo referencia a campos individuales, úsarlos para filtrar y enriquecer los datos.

1. En el Azure portal, en el blade de Azure Data Analytics account, click en la pagina Jobs y revise los detalles de los Jobs que ha ejecutado hasta el momento
2. Click en el job mas reciente: **Process Log Entries**
3. En el blade **Process Log Entries,** click View Script para desplegar el U-SQL, luego click en **Duplicate Script** para crear un nuevo job basado en este script
4. En el blade **New U-SQL Job**, en el **Job Name** box escriba **Summarize Log**, luego modifique el script tal como se muestra a continuación

```
@log = EXTRACT date string,
```

```
        time string,
        client_ip string,
        username string,
        server_ip string,
        port int,
        method string,
        stem string,
        query string,
        status string,
        server_bytes int,
        client_bytes int,
        time_taken int,
        user_agent string,
        referrer string
FROM "/iislogs/2008-01.txt"
USING Extractors.Text(' ', silent:true);

@dailysummary = SELECT date,
        COUNT(*) AS hits,
        SUM(server_bytes) AS bytes_sent,
        SUM(client_bytes) AS bytes_received
FROM @log
GROUP BY date;

OUTPUT @dailysummary
TO "/output/daily_summary.csv"
ORDER BY date
USING Outputters.Csv();
```

Este codigo usa una instrucción SELECT con un GROUP BY para sumarizar el total de log entries, server bytes and client bytes por dia. Los resultados se escriben en el Azure Data Lake store usando el Csv outputter, con un ORDER BY que ordena los resultados en orden ascendente al campo date.

5. Click **Submit Job** y monitoree los detalles de ejecución.
6. Cuando el job haya completado su ejecución, click en el **Output** tab y seleccione **daily_summary.csv** para visualizar los resultados
7. Descargue el archivo resultado y ábralo en un editor de texto o en un spreadsheet y note que cada fila contiene un daily summary de hits, bytes enviados y bytes recibidos.

Procesando múltiples archivos

En este ejercicio, procesaremos múltiples archivos

Subiendo source data files

Ademas al log file de Enero 2008 que ha sido cargado con anterioridad, ahora procesaremos datos de febrero a junio.

1. En el folder donde extrajo los archivos de esta practica, verifique que hay archivos para 5 meses mas
2. En el Azure portal, vaya a la pagina Data Explorer del Azure Data Lake Analytic account y expanda el iislog folder
3. Click Upload, y luego seleccione los archivos restantes y cárguelos

Creando un job para procesar multiples archivos

Ahora los datos están diseminados en multiples archivos, podemos usar un wildcard en el query para leer los datos de todos ellos

1. En el Azure portal, en el blade de Azure Data Analytics account, click en la pagina Jobs y revise los detalles de los Jobs que ha ejecutado hasta el momento
2. Click en el job mas reciente: **Process Log Entries**
3. En el blade **Process Log Entries,** click View Script para desplegar el U-SQL, luego click en **Summarize Log** para crear un nuevo job basado en este script
4. En el blade **New U-SQL Job**, en el **Job Name** box escriba **Summarize Logs**, luego modifique el script tal como se muestra a continuación

```
@log = EXTRACT date string,
          time string,
          client_ip string,
          username string,
          server_ip string,
          port int,
          method string,
          stem string,
          query string,
          status string,
          server_bytes int,
          client_bytes int,
          time_taken int,
          user_agent string,
          referrer string
     FROM "/iislogs/{*}.txt"
     USING Extractors.Text(' ', silent:true);

     @dailysummary =
     SELECT date,
          COUNT(*) AS hits,
          SUM(server_bytes) AS bytes_sent,
          SUM(client_bytes) AS bytes_received
          FROM @log
     GROUP BY date;

     OUTPUT @dailysummary
     TO "/output/six_month_summary.csv"
     ORDER BY date
     USING Outputters.Csv();
```

Este query usa un wildcard placeholder **{*}** para leer todos los archivos con extensión .txt del **iis** folder.

5. Click **Submit Job** y monitoree los detalles de ejecución.
6. Cuando el job haya completado su ejecución, click en el **Output** tab y seleccione **six_month_summary.csv** para visualizar los resultados
7. Descargue el archivo resultado y ábralo en un editor de texto o en un spreadsheet y note que cada fila contiene un daily summary de hits, bytes enviados y bytes recibidos entre enero y junio.