Beginning Pig

This tutorial will work through the basics of getting started with Pig. You will be creating a Pig Latin script to analyse a dataset of 50,000 events from lovecleanstreats.com. First you will work through getting pig set up on your local machine and running Pig in local mode via the shell. Once you have done this you will be submitting your pig script to an HDInsight cluster using C# and the Hadoop SDK.

Lovecleanstreets.com is an Azure service that collects data about incidents such as fly tipping on behalf of local authorities in the UK. You should begin by downloading the dataset from here. It is worth opening it in excel and taking a look so you can get a feel for the data:

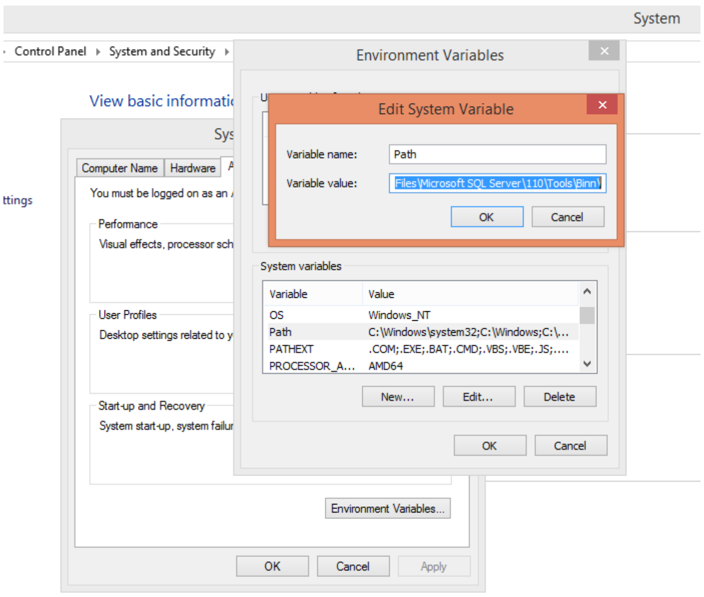
http://elastastorage.blob.core.windows.net/hdinsight/LoveCleanStreets.txt

Downloading and installing Pig

The next thing to do is to download and install Pig. Below is a link the version of Pig that we will be using. Go ahead and download it.

http://mirror.gopotato.co.uk/apache/pig/pig-0.11.1/pig-0.11.1.tar.gz

Once you have un-packed the distribution you need to add /pig-0.11.1/bin to your path. If you are on Windows go to your environment variables and under system variables edit the path entry adding the path to Pig as shown below:



Variable name: Path

Variable value: Files\Microsoft SQL Server\110\Tools\Binn\

System variables

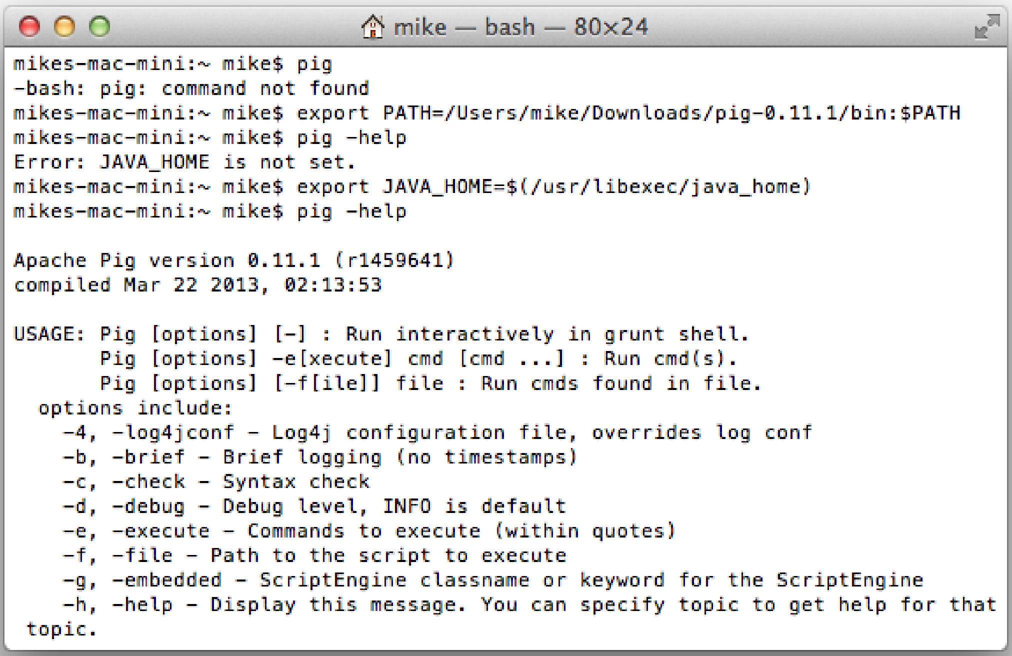
Variable	Value
OS	Windows_NT
Path	C:\Windows\system32;C:\Windows;C:\...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;....
PROCESSOR_A...	AMD64

If you are on a mac open up terminal and add paste in the following putting in the path to you unpacked distribution.

export PATH=/<my-path-to-pig>/pig-0.11.1/bin:$PATH

Next test the Pig installation with this simple command: $ pig -help

Please note that Pig requires Java to be installed and you need to set JAVA_HOME to the root of your Java installation. You will get an error when you try and run the above command if this has not been set. Below shows an example of setting the path and JAVA_HOME. Once pig is installed correctly you should see the below output.



```
mikes-mac-mini:~ mike$ pig
-bash: pig: command not found
mikes-mac-mini:~ mike$ export PATH=/Users/mike/Downloads/pig-0.11.1/bin:$PATH
mikes-mac-mini:~ mike$ pig -help
Error: JAVA_HOME is not set.
mikes-mac-mini:~ mike$ export JAVA_HOME=$(/usr/libexec/java_home)
mikes-mac-mini:~ mike$ pig -help

Apache Pig version 0.11.1 (r1459641)
compiled Mar 22 2013, 02:13:53

USAGE: Pig [options] [-] : Run interactively in grunt shell.
       Pig [options] -e[xecute] cmd [cmd ...] : Run cmd(s).
       Pig [options] [-f[ile]] file : Run cmds found in file.
  options include:
    -4, -log4jconf - Log4j configuration file, overrides log conf
    -b, -brief - Brief logging (no timestamps)
    -c, -check - Syntax check
    -d, -debug - Debug level, INFO is default
    -e, -execute - Commands to execute (within quotes)
    -f, -file - Path to the script to execute
    -g, -embedded - ScriptEngine classname or keyword for the ScriptEngine
    -h, -help - Display this message. You can specify topic to get help for that
 topic.
```

Finally, if you are on Windows then you will need to install Cygwin:

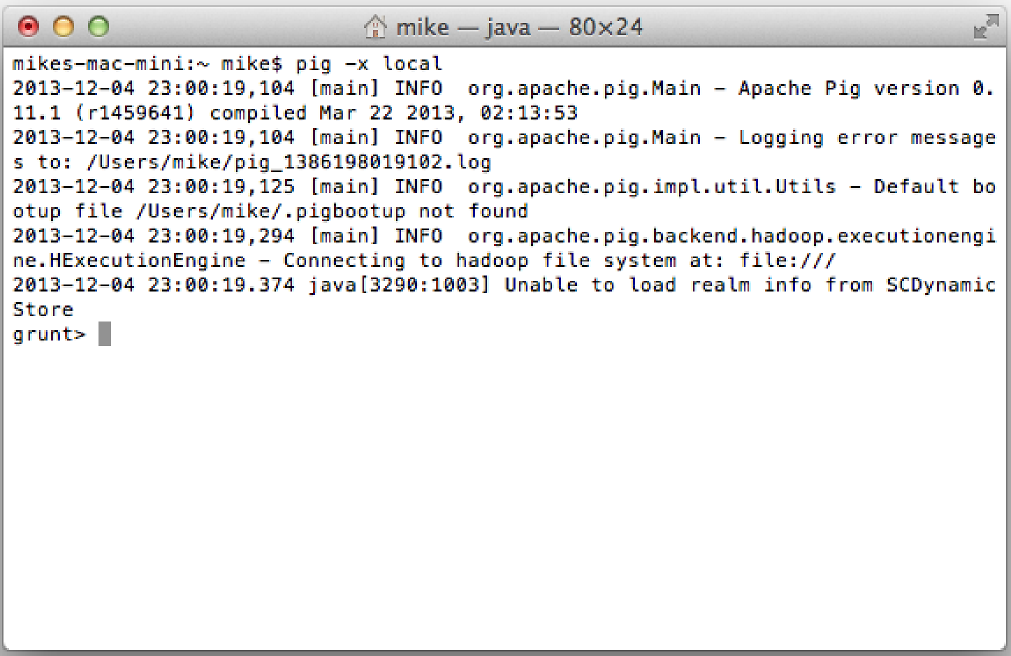
http://cygwin.com/setup-x86_64.exe

Usually you would also install Hadoop but if you don't Pig will run with the embedded version, currently Hadoop 1.0.0. This is fine for the purpose of this workshop.

Running a script in grunt

Pig has two execution modes. Local mode and mapreduce mode. To run Pig in mapreduce mode, you need access to a Hadoop cluster and HDFS installation so for the purpose of this workshop we will be running pig in Local Mode. To run pig in local mode type the following:

Pig –x local



```
mikes-mac-mini:~ mike$ pig -x local
2013-12-04 23:00:19,104 [main] INFO  org.apache.pig.Main - Apache Pig version 0.
11.1 (r1459641) compiled Mar 22 2013, 02:13:53
2013-12-04 23:00:19,104 [main] INFO  org.apache.pig.Main - Logging error message
s to: /Users/mike/pig_1386198019102.log
2013-12-04 23:00:19,125 [main] INFO  org.apache.pig.impl.util.Utils - Default bo
otup file /Users/mike/.pigbootup not found
2013-12-04 23:00:19,294 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to hadoop file system at: file:///
2013-12-04 23:00:19.374 java[3290:1003] Unable to load realm info from SCDynamic
Store
grunt>
```

As you can see above we are now in grunt and can start executing some Pig!

We are going to attempt something very simple but it will introduce you to a few Pig constructs. We want to know a count of the number of incidents for each local authority on a certain date.

Type the following command into the terminal window and press enter:

cleanstreetdata = load '/path/to/LoveCleanStreets.txt' using PigStorage('\t');

This will instruct pig to load the data from the tab-delimited file into the dataset *cleanstreetdata*. Pig Latin is a dataflow language. Each processing step results in a new dataset. In the line *cleanstreetdata* = load *'/path/to/LoveCleanStreets.txt'*, *cleanstreetdata* is the name of the dataset that results from loading the data in the file.

Next type in the following and press enter:

grouped = group cleanstreetdata by ($13,SUBSTRING($4,0,10));

This will collect all of the records together with the same key (local authority and date of the report). Also note that group is a blocking operator and requires repartitioning and shuffling which will force a reduce phase.

Side note – Schemas in Pig

There are some interesting things to note about the above group by statement. Firstly $13 will access the field at that index within the dataset. It is possible in pig to impose a schema on a dataset with named fields and types. In the example below we impose a schema on the data so we can access fields by their name.

It is worth noting that Pig will tolerate some inconsistencies between the schema declared and the file. For example if the 'market_data_prices' file had four fields then the forth would be ignored. Also, note that we haven't specified a load function (e.g. using PigStorage). By default Pig will just use the PigStorage loader and will expect a tab delimited file.

e.g with schema

```
prices = load 'market_data_prices' as (ISIN, openprice, closeprice);

change   = foreach prices generate closeprice - openprice;
```

Side note – UDFs

Notice that we used the built in SUBSTRING User defined function in our group by statement to remove the time part of the date. It is possible to extend pigs with UDFs in order to add custom processing logic. UDFs are typically written in Java but can also be written in python. Pig comes with a substantial set of built in functions and libraries such as Piggy Bank add even more.

Side note – Pig Data Types

Pig has two categories of types - scalar types and complex types.

The Pigs scalar types are typical of most programming language and include things such as int, long, chararry, bytearray. It also has three main complex types – Map, Tuple and Bag. A map is essentially a key value pair. A tuple is a fixed length, ordered collection of fields and is like a row in sql. A Bag is an unordered collection of Tuples.

The records produced by the group by statement have two fields, the key and the bag of records. The key field is named *group*. The bag is named after the alias that was grouped - in our case *cleanstreetdata.* It also has same schema but as the dataset that was grouped however as *cleanstreetdata* has no schema, the bag *cleanstreetdata* will also have no schema.

Next type in the following and press enter:

```
count = foreach grouped generate group, COUNT(cleanstreetdata);
```

As you would expect the foreach applies the generate expression to each item in the pipeline and for each one sends a new record down the pipeline. In this case it will generate us a count of all the items sharing the same key.

If you need to you can view the items in a dataset by dumping them to the console. This will force the map reduce job to run and dump the output to the screen:

```
dump count; -
```

Once we have finished processing the data we need to store it somewhere.

```
store count into '/some/valid/path/output/';
```

That's it! We now have a complete pig script which reads in some data, groups it by a key, counts the members of each group and saves the output to disk. The complete script is as follows:

```
cleanstreetdata = load '/path/to/LoveCleanStreets.txt' using PigStorage('\t');

grouped = group cleanstreetdata by ($13,SUBSTRING($4,0,10));

count = foreach grouped generate group, COUNT(cleanstreetdata);

store count into '/some/valid/path/output/';
```

Now lets get it running on HDInsight!

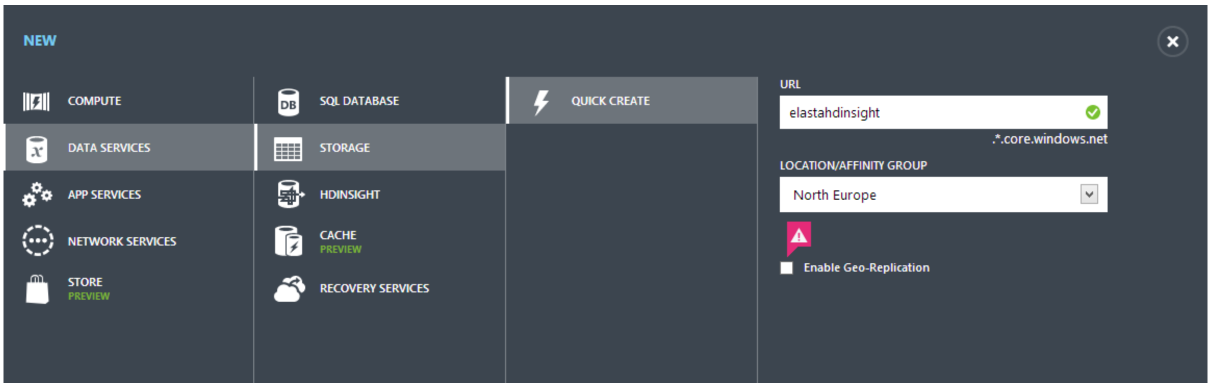
Creating the cluster

Now that we have run the script in pig using local mode we can submit it to a real cluster. We will do this using the Hadoop SDK. There are a few things we need to prepare first.

Copy the script below into a text file and save it as lovecleanstreets.pig.

```
cleanstreetdata = load '/input/lovecleanstreets.txt' using PigStorage('\t');

grouped = group cleanstreetdata by ($13,SUBSTRING($4,0,10));

count = foreach grouped generate group, COUNT(cleanstreetdata);

store count into '/output/';
```

Next create a Windows Azure Storage account. To do this, navigate to the Windows Azure Portal and use QuickCreate as shown below:

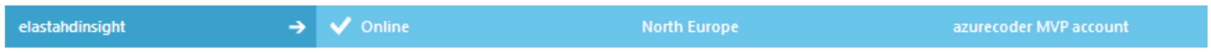


URL: elastahdinsight .*.core.windows.net

LOCATION/AFFINITY GROUP: North Europe

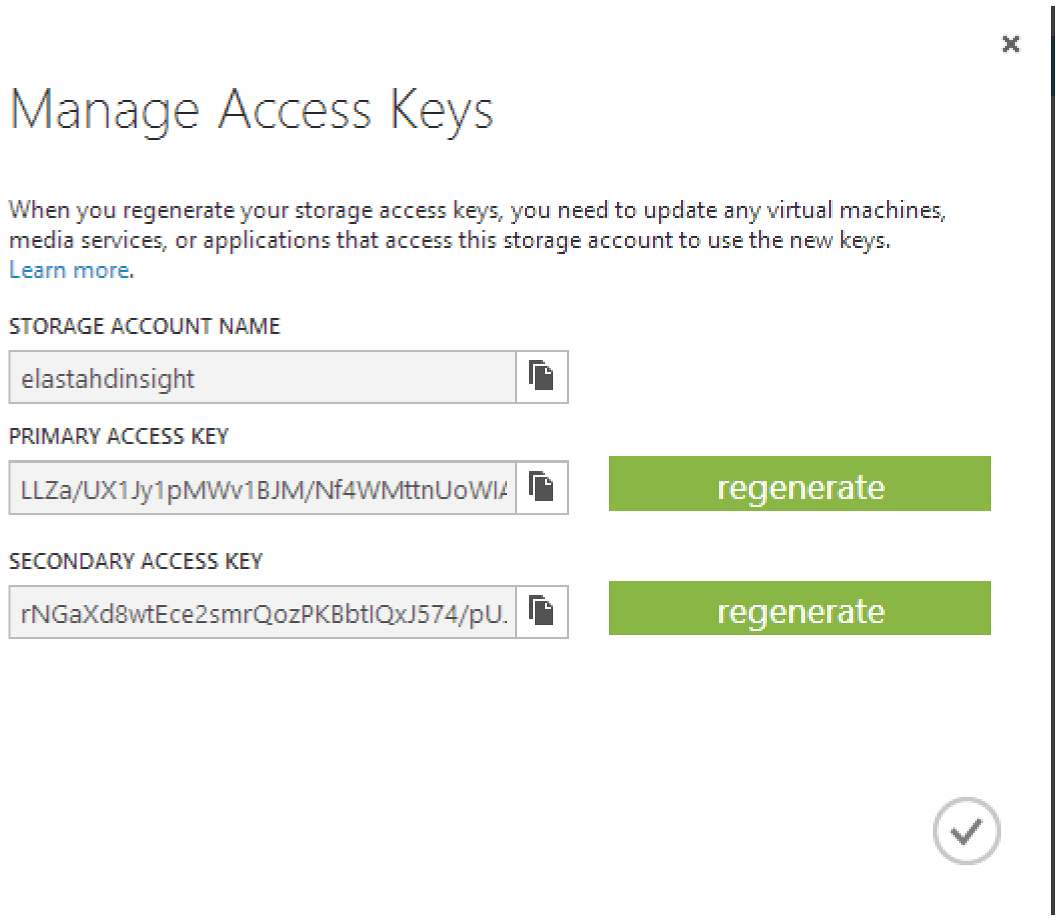
Enable Geo-Replication

It should take no more than a minute and the account will be available.



elastahdinsight → ✓ Online | North Europe | azurecoder MVP account

You should be able to highlight it and select the Manage Access Keys option from the AppBar menu.



Manage Access Keys

When you regenerate your storage access keys, you need to update any virtual machines, media services, or applications that access this storage account to use the new keys. Learn more.

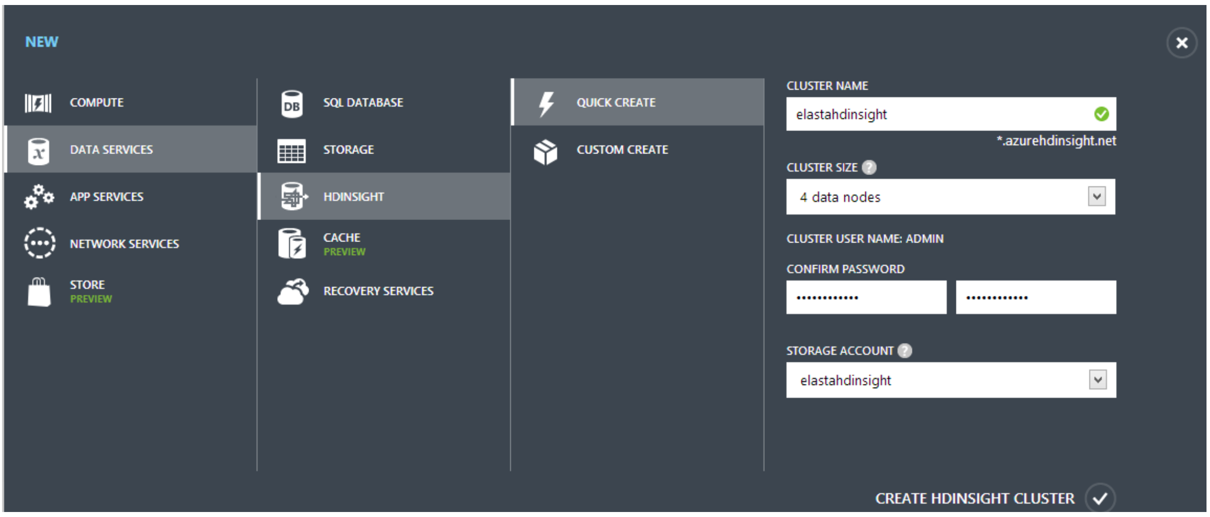
STORAGE ACCOUNT NAME: elastahdinsight

PRIMARY ACCESS KEY: LLZa/UX1Jy1pMWv1BJM/Nf4WMttnUoWI/

SECONDARY ACCESS KEY: rNGaXd8wtEce2smrQozPKBbtlQxJ574/pU.

Copy the primary key using the symbol to the right of the textbox.

Next create an HDInsight cluster as follows setting the storage account as the one you have just made:



CLUSTER NAME: elastahdinsight .azurehdinsight.net

CLUSTER SIZE: 4 data nodes

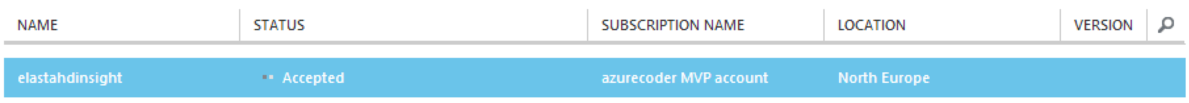
CLUSTER USER NAME: ADMIN

CONFIRM PASSWORD: ....

STORAGE ACCOUNT: elastahdinsight

By default the username is Admin. Stick with a 4 node cluster which should cost around £1.20/hour.

The cluster will take a little longer than the storage account to be created, around 12-15 minutes.

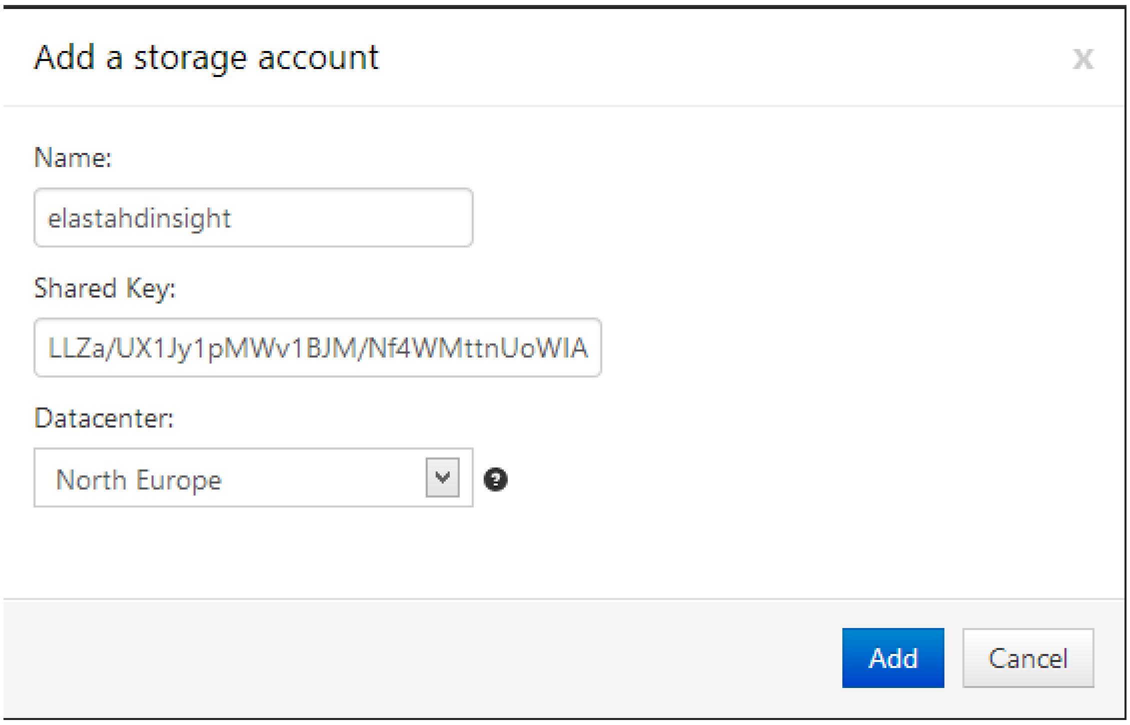


NAME	STATUS	SUBSCRIPTION NAME	LOCATION	VERSION
elastahdinsight	Accepted	azurecoder MVP account	North Europe	

Once the cluster is set up you will see a container with the same name as you cluster in your storage account.

You'll need to use a storage management tool like zud.io or Azure Management Studio to upload the LCS dataset to the input

The example shows zud.io but you can also use AMS if you're used to that. Both are equally good.



Add a storage account

Name: elastahdinsight

Shared Key: LLZa/UX1Jy1pMWv1BJM/Nf4WMttnUoWIA

Datacenter: North Europe

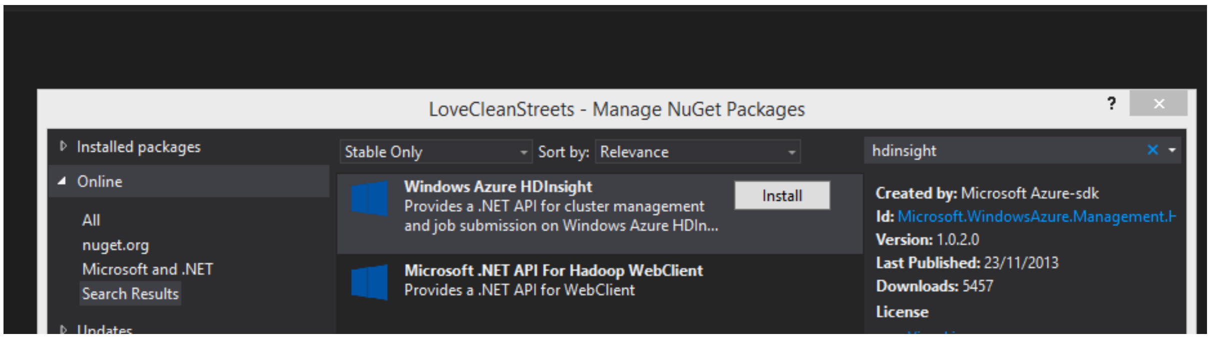
Use the storage management tool to upload **LoveCleanStreets.txt** to a folder called input in the cluster container. The file is 22MB so it may take a few minutes.

Next upload your pig script at the root level of the cluster.

Submitting and Running the Job

Open visual studio and create a console application called LoveCleanStreets.

Add a reference to the HD insight SDK as shown below.


LoveCleanStreets - Manage NuGet Packages

Installed packages
Online
All
nuget.org
Microsoft and .NET
Search Results
Updates

Stable Only | Sort by: Relevance | hdinsight

Windows Azure HDInsight
Provides a .NET API for cluster management and job submission on Windows Azure HDIn...
Install

Microsoft .NET API For Hadoop WebClient
Provides a .NET API for WebClient

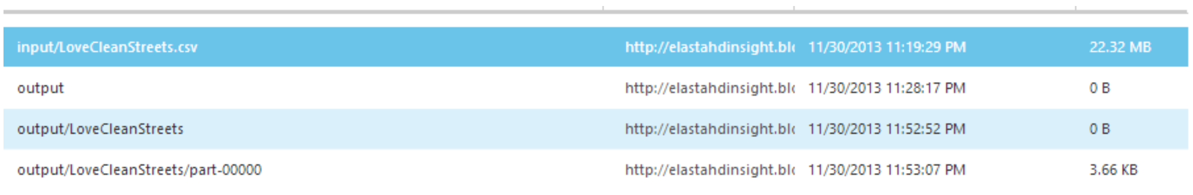
Created by: Microsoft Azure-sdk
Id: Microsoft.WindowsAzure.Management.H
Version: 1.0.2.0
Last Published: 23/11/2013
Downloads: 5457
License

Paste in the following text replacing the setting with the ones appropriate for your cluster.

```
class Program
    {
        static void Main(string[] args)
        {
            var credentials = new BasicAuthCredential
            {
                UserName = "admin",
                Password = "password",
                Server = new Uri("https://mikepiglab.azurehdinsight.net")
            };

            var submissionClient = JobSubmissionClientFactory.Connect(credentials);
            submissionClient.JobStatusEvent += (o, e) =>
Console.WriteLine(e.JobDetails.StatusCode);

            var pigJobParams = new PigJobCreateParameters { StatusFolder =
"wasbs://mikepiglab-1@mikepiglab.blob.core.windows.net/status", File =
"wasbs://mikepiglab-1@mikepiglab.blob.core.windows.net/lovecleanstreets.pig" };
            var job = submissionClient.CreatePigJob(pigJobParams);

            bool complete = false;
            while (!complete)
            {
                var jobDetails = submissionClient.GetJob(job.JobId);
                Console.WriteLine(jobDetails.StatusCode);
                Console.WriteLine(jobDetails.ExitCode);

                if (jobDetails.StatusCode == JobStatusCode.Completed)
                    complete = true;

                Thread.Sleep(2000);

            }

            Console.ReadLine();
        }
    }    }
```

Confirming the output

When the job complete we can inspect the output by looking at the Windows Azure portal and downloading the output file from Blob Storage.



input/LoveCleanStreets.csv	http://elastahdinsight.bl	11/30/2013 11:19:29 PM	22.32 MB
output	http://elastahdinsight.bl	11/30/2013 11:28:17 PM	0 B
output/LoveCleanStreets	http://elastahdinsight.bl	11/30/2013 11:52:52 PM	0 B
output/LoveCleanStreets/part-00000	http://elastahdinsight.bl	11/30/2013 11:53:07 PM	3.66 KB

We should have a 288kB file called **output/part-00000**

When we inspect the output we can see that we have an empty LA. Data generally contains noise and we need to determine whether this is meaningful or noisy to our output. A subsequent update of the script can filter these out if we believe them to be meaningless to what we're trying to show.

```
,2011-08-12     1
,2011-09-11     1
,2011-12-04     1
```

Also look at the data for Lewisham. An additional exercise could be done by enriching the data. A quick scan of dates will show that there is a higher incidence of reports over the weekends. This could be because more crime happens over the weekend or more people are around to report it.

```
Lewisham London Borough,2011-07-04  1
Lewisham London Borough,2011-07-11  3
Lewisham London Borough,2011-08-11  6
Lewisham London Borough,2011-09-11  4
Lewisham London Borough,2011-10-11  4
Lewisham London Borough,2012-01-04  4
Lewisham London Borough,2012-01-05  27
Lewisham London Borough,2012-02-04  22
Lewisham London Borough,2012-02-05  18
Lewisham London Borough,2012-03-04  33
Lewisham London Borough,2012-04-04  22
Lewisham London Borough,2012-04-05  1
Lewisham London Borough,2012-05-04  30
Lewisham London Borough,2012-06-04  2
Lewisham London Borough,2012-06-08  1
Lewisham London Borough,2012-07-04  3
Lewisham London Borough,2012-08-04  3
Lewisham London Borough,2012-08-05  2
Lewisham London Borough,2012-09-04  7
Lewisham London Borough,2012-10-04  50
Lewisham London Borough,2012-11-04  23
Lewisham London Borough,2012-12-04  28
```

Challenge

We've looked at simple usage of Map-Reduce. In this challenge we want you to enhance the mapper and reducer to enable the determination of the average time of resolution of a reported incident per LA and per LA/Ward (two jobs).