



SPEED LAB:
AZURE STREAM ANALYTICS
SETUP AND RUNNING IN MINUTES
TO GET REAL-TIME INSIGHTS

ABSTRACT

Learn how to get your own real-time insights from your data set up and running in minutes using Azure Stream Analytics. Azure's new stream processing service enables developers to easily tackle the space of data in motion by combining streams of data such as click-streams, logs and device generated events with historical records or reference data to derive business insights easily and quickly. Being a fully managed, real-time stream computation service hosted in Microsoft Azure, Azure Stream Analytics provides built-in resiliency, low latency, and scalability to get you up and running in minutes. Join us to become a guru in your enterprise in delivering real-time insights using Azure Stream Analytics!

Contents

1.	Goals	3
2.	Azure Stream Analytics.....	4
3.	“Hello, Toll!” – Introduction	4
	Incoming data	4
	Commercial Vehicle Registration Data	6
4.	Setting up environment for Azure Stream Analytics	6
	Provisioning Azure resources required for the Lab	7
	Connect to Database from Visual Studio	12
	Event Generator - TollApp Sample Project	14
	Enable Azure Stream Analytics feature	15
5.	Create Stream Analytics Job	15
	Define input sources	16
	Define output	23
	Azure Stream analytics query	24
6.	Question 1 - Number of vehicles entering a toll booth	26
	Application time - TimeStamp	25
	Time windows – Tumbling Window	26
7.	Running Azure Stream Analytics Jobs.....	26
	Check results in Visual Studio	33
8.	Question 2 - Report Total time for the car to pass the toll booth	28
9.	Question 3 – Report all commercial vehicles with expired registration.....	30
	CREATE TABLE Syntax	Error! Bookmark not defined.
10.	Scaling out Azure Stream Analytics Jobs	34
11.	Monitoring	35
12.	Ordering events.....	37
	Adjust Policy (<i>default</i>).....	37
	Drop Policy.....	37
	Tolerance Window.....	37
13.	Conclusion	38
14.	Cleanup your Azure account	38
APPENDIX.....		39
	Provisioning Event Hub input sources	39

Provisioning a Windows Azure SQL Database	41
Provision a Windows Azure Storage Account.....	44
Starting the Toll Booth Event Generator	45

1. GOALS

After completing this lab you will be able to:

- Familiarize yourself with the Azure Stream Analytics Portal.
- Configure and deploy a streaming job.
- Articulate real world problems and solve them using Stream Analytics Query Language.
- Develop streaming solutions for your customers using Azure Streaming Analytics with confidence.
- Use the monitoring and logging experience to troubleshoot issues.

2. AZURE STREAM ANALYTICS

Azure Stream Analytics is a fully managed service which is highly available, scalable and provides low latency complex event processing over streaming data in the cloud. In its initial release, Azure Stream Analytics enables customers to easily setup a job to analyze streams of data allowing them to drive near real-time analytics and dashboards. The following scenarios are targeted:

- Performing complex event processing on high volume and high velocity data
- Collecting event data from globally distributed assets or equipment such as connected cars or utility grids
- Processing telemetry data for near real time monitoring and diagnostics
- Capturing and archiving real-time events for future processing

This lab will use sample highway toll data to help you learn Azure Stream Analytics.

3. “HELLO, TOLL!” – INTRODUCTION

A tolling station is a common phenomenon – we encounter them in many expressways, bridges, and tunnels across the world. Each toll station has multiple toll booths, which may be manual – meaning that you stop to pay the toll to an attendant, or automated – where a sensor placed on top of the booth scans an RFID card affixed to the windshield of your vehicle as you pass the toll booth. It is easy to visualize the passage of vehicles through these toll stations as an event stream over which interesting operations can be performed.



INCOMING DATA

We will work with two streams of data which are produced by sensors installed in the entrance and exit of the toll stations and a static look up dataset with vehicle registration data.

ENTRY DATA STREAM

Entry data stream contains information about cars entering toll stations.

Toll Id	EntryTime	License Plate	State	Make	Model	Vehicle Type	Vehicle Weight	Toll	Tag
1	2014-09-10 12:01:00.000	JNB 7001	NY	Honda	CRV	1	0	7	
1	2014-09-10 12:02:00.000	YXZ 1001	NY	Toyota	Camry	1	0	4	123456789
3	2014-09-10 12:02:00.000	ABC 1004	CT	Ford	Taurus	1	0	5	456789123
2	2014-09-10 12:03:00.000	XYZ 1003	CT	Toyota	Corolla	1	0	4	
1	2014-09-10 12:03:00.000	BNJ 1007	NY	Honda	CRV	1	0	5	789123456
2	2014-09-10 12:05:00.000	CDE 1007	NJ	Toyota	4x4	1	0	6	321987654
	...								

Here is a short description of the columns:

TollID	Toll booth ID uniquely identifying a toll booth
EntryTime	The date and time of entry of the vehicle to Toll Booth in UTC
LicensePlate	License Plate number of the vehicle
State	Is a State in United States
Make	The manufacturer of the automobile
Model	Model number of the automobile
VehicleType	1 for Passenger and 2 for Commercial vehicles
WeightType	Vehicle weight in tons; 0 for passenger vehicles
Toll	The toll value in USD
Tag	e-Tag on the automobile that automates payment, left blank where the payment was done manually

EXIT DATA STREAM

Exit data stream contains information about cars leaving the toll station.

TollId	ExitTime	LicensePlate
1	2014-09-10T12:03:00.0000000Z	JNB 7001
1	2014-09-10T12:03:00.0000000Z	YXZ 1001
3	2014-09-10T12:04:00.0000000Z	ABC 1004
2	2014-09-10T12:07:00.0000000Z	XYZ 1003
1	2014-09-10T12:08:00.0000000Z	BNJ 1007
2	2014-09-10T12:07:00.0000000Z	CDE 1007
	...	

Column descriptions:

TollID	Toll booth ID uniquely identifying a toll booth
ExitTime	The date and time of exit of the vehicle from Toll Booth in UTC
LicensePlate	License Plate number of the vehicle

COMMERCIAL VEHICLE REGISTRATION DATA

We will use a static snapshot of commercial vehicle registration database.

LicensePlate	RegistrationId	Expired
SVT 6023	285429838	1
XLZ 3463	362715656	0
BAC 1005	876133137	1
RIV 8632	992711956	0
SNY 7188	592133890	0
ELH 9896	678427724	1
...		

Column descriptions:

LicensePlate	License Plate number of the vehicle
RegistrationId	RegistrationId
Expired	0 if vehicle registration is active, 1 if registration is expired

4. SETTING UP ENVIRONMENT FOR AZURE STREAM ANALYTICS

To perform this lab, you will need a Microsoft Azure subscription. Microsoft offers free trial for Microsoft Azure services as described below.

If you do not have an Azure account, you can request a free trial version by going to <http://azure.microsoft.com/en-us/pricing/free-trial/>.

Note: To sign up for a free trial, you will need a mobile device that can receive text messages and a valid credit card.

Be sure to follow the “Clean up your Azure account” section steps at the end of this exercise so that you can make the most use of your \$200 free Azure credit.

PROVISIONING AZURE RESOURCES REQUIRED FOR THE LAB

This lab will require 2 Azure Event Hubs to receive “Entry” and “Exit” data streams. We will use Azure SQL Database to output the results of the Stream Analytics jobs. We will also use Azure Storage to store reference data about vehicle registration.

The Setup.ps1 script in the TollApp sample folder on GitHub can be used to create all required resources. In the interest of time, we recommend that you run it. If you would like to learn more about configuring these resources in Azure portal, please refer to the appendix “Configuring Lab resources in Azure Portal”

Download and save the supporting [TollApp](#) folder and files.

Open a “Microsoft Azure PowerShell” window. If you do not yet have Azure PowerShell, follow the instructions here to install it: <http://azure.microsoft.com/en-us/documentation/articles/install-configure-powershell/>

Go to the directory with the scripts and generator application.

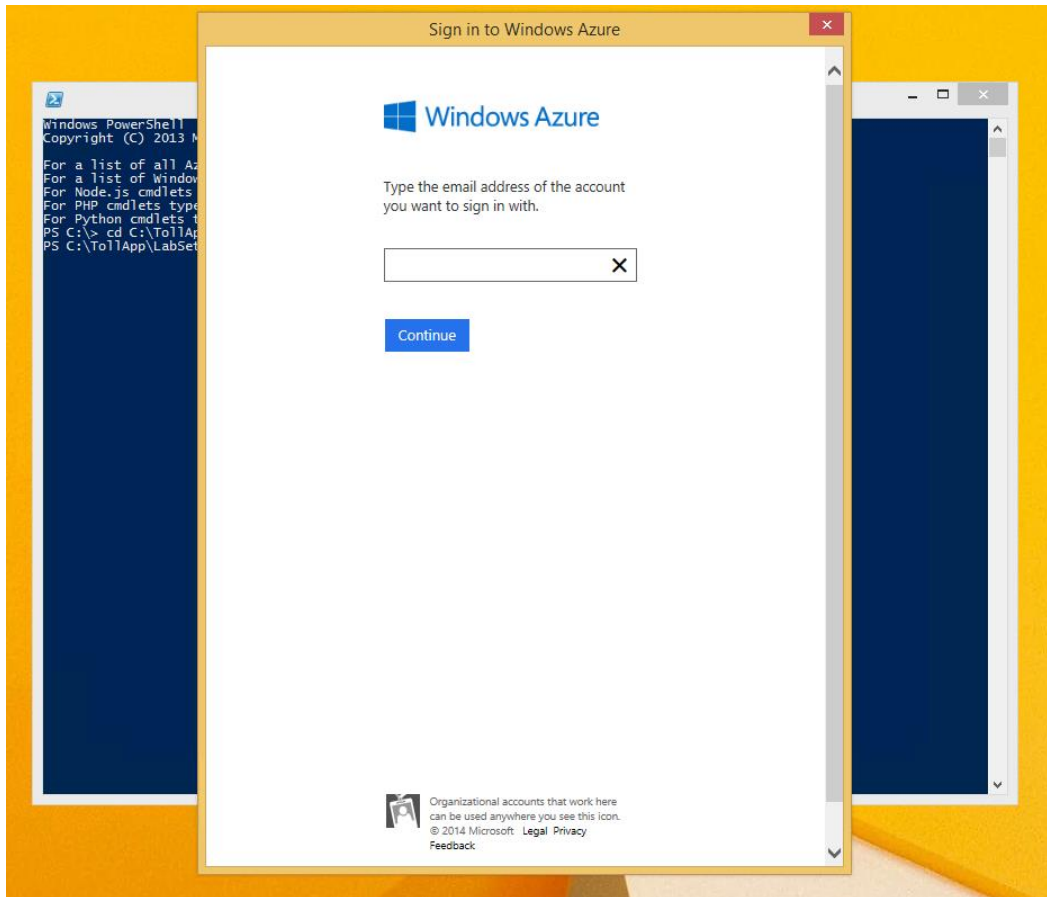
```
PS C:\> cd C:\TollApp\LabSetup
PS C:\TollApp\LabSetup>
```

Windows automatically blocks ps1, dll and exe files downloaded from the Internet. Please run “Unblock-File *” command to unblock files:

```
PS C:\TollApp\LabSetup> Unblock-File *
```

Type “.\Setup.ps1” to setup all required resources and start generating events

The script will open “Sign In” page for Windows Azure. Enter your account credentials.



Please note that if your account has access to multiple subscriptions, you will be asked to enter the subscription name that you want to use for the lab.

The script can take several minutes to run. Once completed, the output should look like the screenshot below.

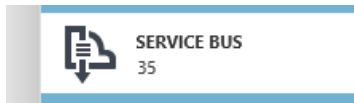
```
Microsoft Azure PowerShell
PS C:\TollApp\LabSetup> .\Setup.ps1
Create Service Bus namespace and Event Hubs
Created Service Bus namespace TollData4637388511
Created Event Hub: entry
Created Event Hub: exit
Create Azure Sql Database and tables
Created Sql Server qec8iyp4ja.database.windows.net
Created Sql Database: TollDataDB
Created Sql tables
Create storage account and upload reference data file
Created storage account tolldata4637388511
Uploaded reference data file
Start generating events
Sending event hub data... Press Ctrl+c to stop.
```

*Note: This application is sending events to your EventHub and is required to run the lab exercises. So you **should not stop the application or close this PowerShell window** until you finish the lab.*

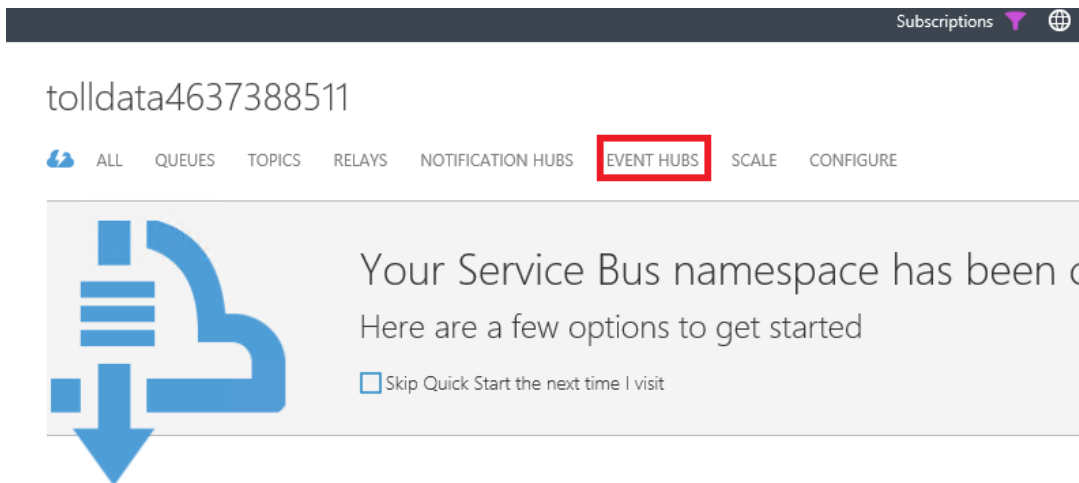
You should be able to see all created resources in Azure Management Portal now. Please go to <https://manage.windowsazure.com> and login with your account credentials.

EVENT HUBS

Click on “Service Bus” menu item on the left side of the Azure Management Portal to see Event Hubs created by the script from the previous section.



You will see all available namespaces in your subscription. Click on the one starting with “TollData”. (TollData4637388511 in our example). Click on “Event Hubs” tab.



You will see two event hubs named *entry* and *exit* created in this namespace.

tolldata4637388511

NAME		STATUS	PARTITION COUNT
entry	→	✓ Active	16
exit	→	✓ Active	16

AZURE STORAGE CONTAINER

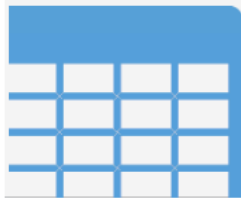
Click on “Storage” menu item on the left side of the Azure Management Portal to see storage container used in the Lab.



Click on the one starting with “tolldata”. (tolldata4637388511 in our example). Open “Containers” tab to see the created container.

tolldata4637388511

 DASHBOARD  MONITOR  CONFIGURE  CONTAINERS  IMPORT/EXPORT



Your storage account has been created
Here are a few options to help you get started.

☐ Skip Quick Start the next time I visit



Get the tools ?

[Storage Explorers](#)

[Install the Windows Azure SDK](#)

Click on “tolldata” container to see uploaded CSV file with vehicle registration data.

tolldata

NAME	URL	LAST MODIFIED	SIZE
registration.csv	https://tolldata4637388511.blob.core.windc	10/20/2014 3:51:08 AM	214.88 KB

AZURE SQL DATABASE

Click on “SQL Databases” menu item on the left side of the Azure Management Portal to see Azure SQL Database that will be used in the Lab.



Click on "TolIDataDB"

tolldatadb

[DASHBOARD](#) [MONITOR](#) [SCALE](#) [CONFIGURE](#) [GEO-REPLICATION](#) [AUDITING & SECURITY](#) [PREVIEW](#)



You created a new SQL database
Here are a few options to get you started

☐ Skip Quick Start the next time I visit



Get Microsoft database design tools ?

[Install Microsoft SQL Server Data Tools](#)



Design your SQL database ?

[Download a starter project for your SQL database](#) [Set up Windows Azure firewall rules for this IP address](#) [Download the Elastic Scale APIs preview](#)



Connect to your database ?

[Design your SQL database](#) [Run Transact-SQL queries against your SQL database](#) [View SQL Database connection strings for ADO .Net, ODBC, PHP, and JDBC](#)

Server: [REDACTED].database.windows.net 1433

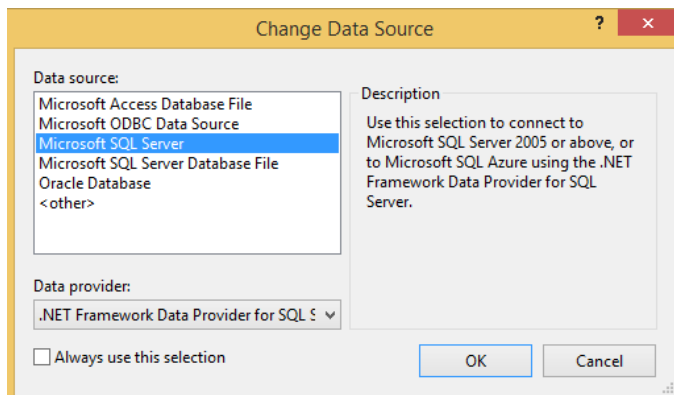
Copy the server name without the port number (<serverName>.database.windows.net for example)

CONNECT TO DATABASE FROM VISUAL STUDIO

We will use Visual Studio to access query results in the output database.

Connect to the Azure database (the destination) from Visual Studio:

- 1) Open Visual Studio then click “Tools” and then “Connect to Database...” menu item.
- 2) If asked, select “Microsoft SQL Server” as a data source



- 3) In the Server Name field paste the name of the SQL Server copied in the previous section from Azure Portal (i.e. <serverName>.database.windows.net)
- 4) In the Authentication field choose SQL Server Authentication
- 5) Enter a LOGIN NAME as “tolladmin” and LOGIN PASSWORD as “123toll!”
- 6) Choose TollDataDB as the database

Add Connection [?] [X]

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:

Server name:

Log on to the server

☐ Use Windows Authentication
☒ Use SQL Server Authentication

User name:

Password:

☐ Save my password

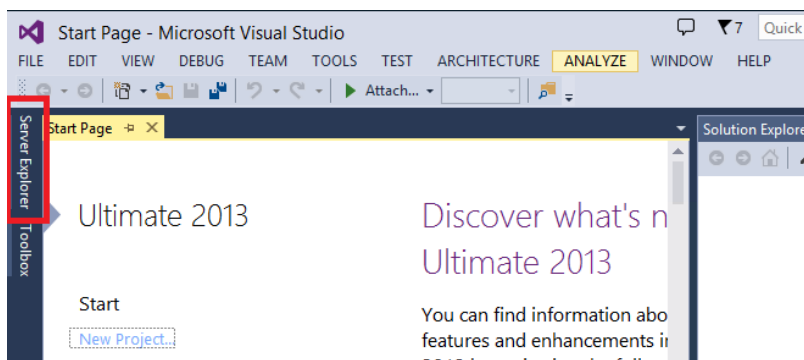
Connect to a database

☒ Select or enter a database name:

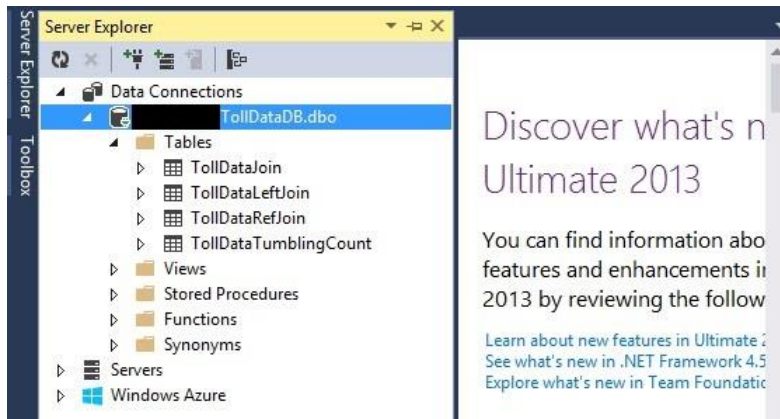
☐ Attach a database file:

Logical name:

- 7) Click OK.
- 8) Open Server Explorer



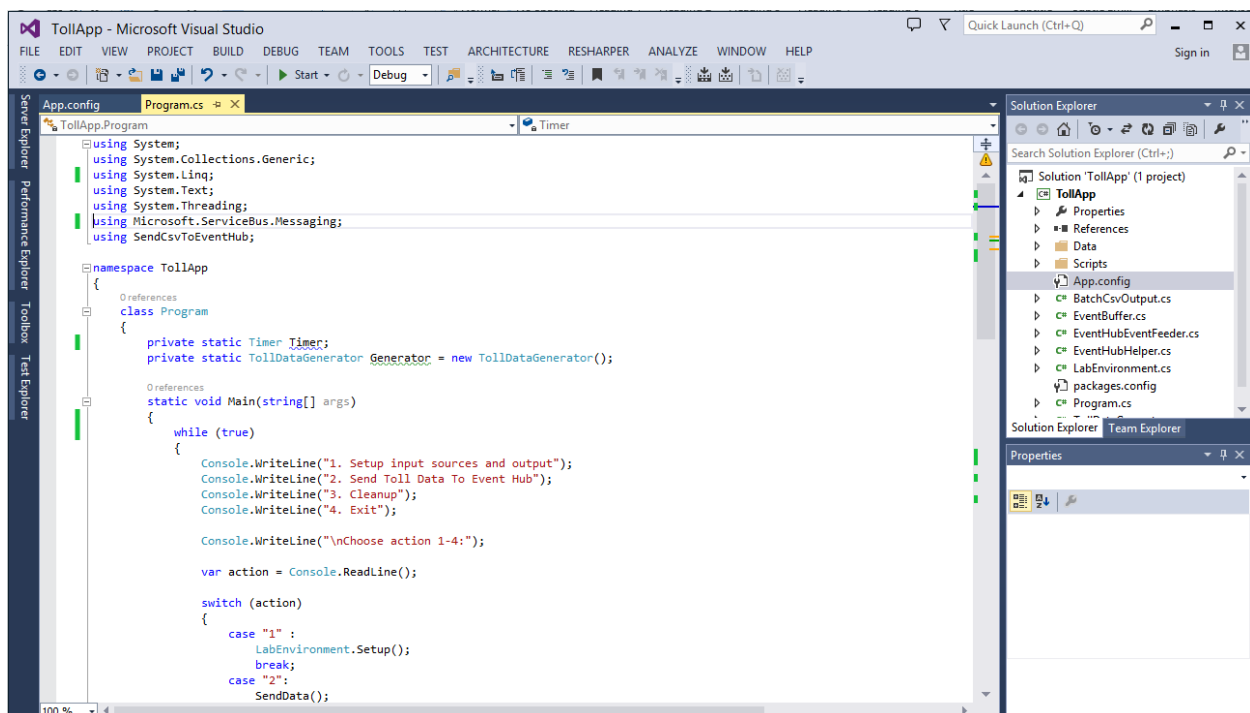
- 9) See 4 tables created in the TollDataDB database.



EVENT GENERATOR - TOLLAPP SAMPLE PROJECT

The PowerShell script automatically starts sending events using the TollApp sample application program. You don't need to perform any additional steps.

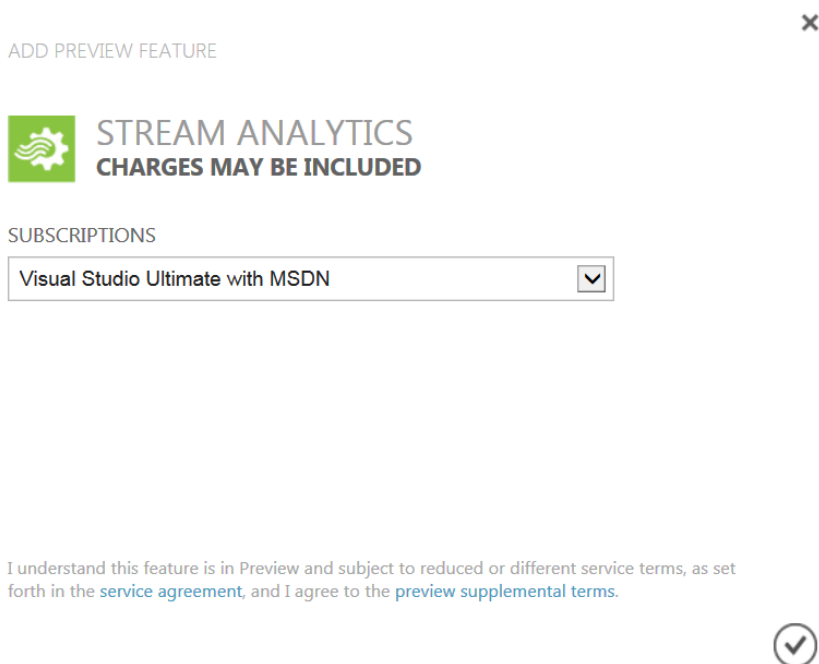
However, if you are interested in implementation details, you can find the source code of the TollApp application under TollApp\SourceCode



ENABLE AZURE STREAM ANALYTICS FEATURE

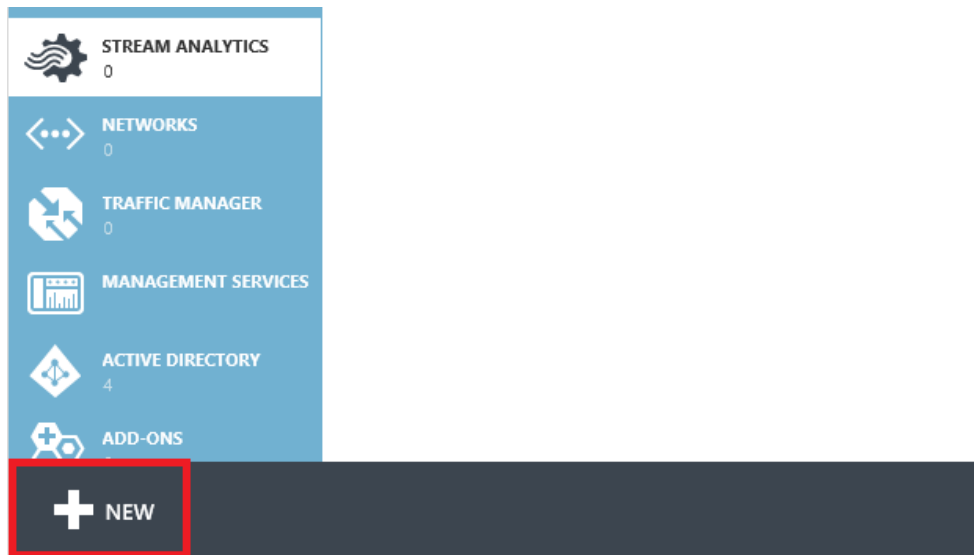
Azure Stream Analytics is in the Preview program. Please open <https://account.windowsazure.com/PreviewFeatures?fid=streamanalytics> and click on the OK button in the lower right corner of the window to enable Azure Stream Analytics in the Azure Portal.

Please note that you need to be Administrator of the subscription to be able to enable Preview features in the Azure Portal.



5. CREATE STREAM ANALYTICS JOB

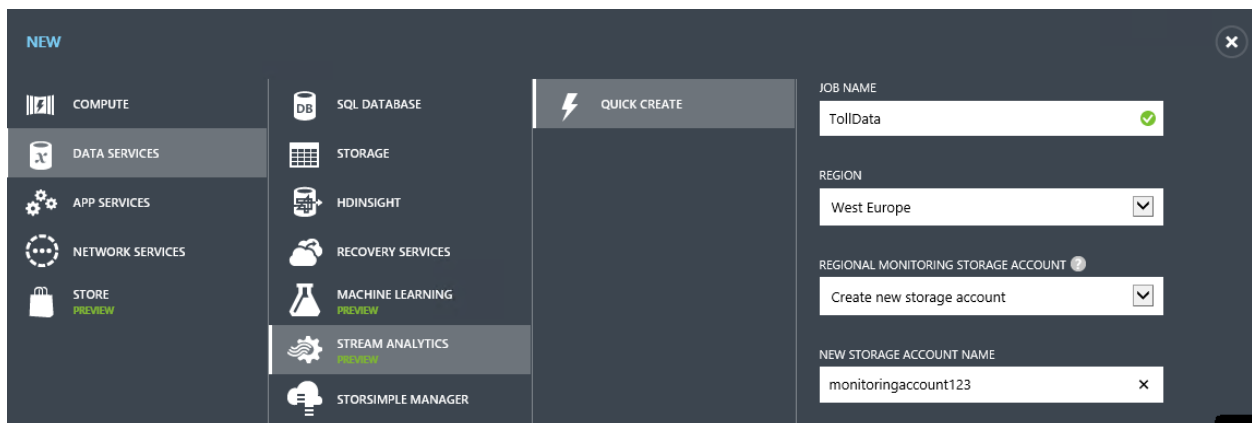
In Azure portal open Stream Analytics and click “New” in the bottom left hand corner of the page to create a new analytics job.



Click “Quick Create”. Select either “West Europe” or “Central US” as the region.

For “Regional Monitoring Storage Account” setting, select “Create new storage account” and give it any unique name. Azure Stream Analytics will use this account to store monitoring information for all your future jobs.

Click “Create Stream Analytics Job” at the bottom of the page.



DEFINE INPUT SOURCES

Click on the created analytics job in the portal.


stream analytics PREVIEW

NAME		STATUS	CREATED	SUBSCRIPTION	REGION	
TollData	→	✓ NOT STARTED	10/17/2014 7:22:30 PM	Prototypes	West Europe	

Open “Inputs” tab to define the source data.

tolldata PREVIEW

 [DASHBOARD](#) [MONITOR](#) **[INPUTS](#)** [QUERY](#) [OUTPUT](#) [SCALE](#) [CONFIGURE](#)



Your Stream Analytics job has been created!

Here are a few options to help you get started.

☐ Skip Quick Start the next time I visit

1

Add inputs

Create one or more new input sources (if you don't have one already).

2

Develop a query

Download [code samples](#), and [tools](#), or explore our [documentation](#) and [tutorials](#) if you want more information.

Click “Add an Input”

You have no inputs. Add one to get started!

ADD AN INPUT →

Select “Data Stream” on the first page

ADD AN INPUT

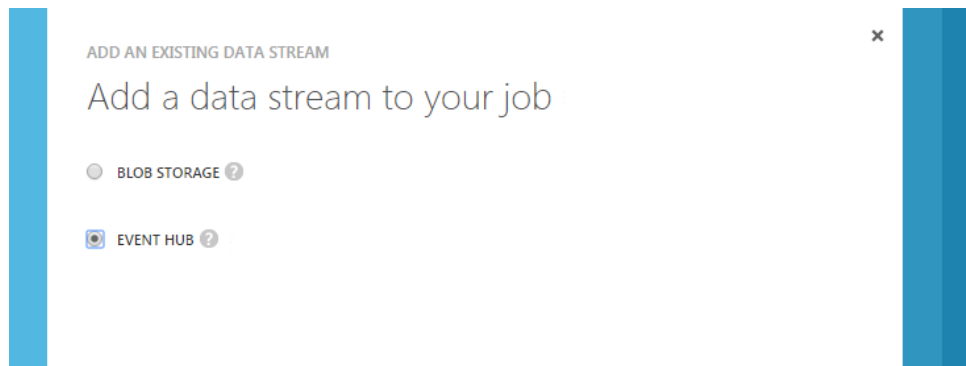
Add an input to your job

Each job requires at least one data stream input. Adding reference data input is optional.

DATA STREAM

A continuous sequence of data or events to be consumed and transformed by a Stream Analytics job.

Select “Event Hub” on the second page of the wizard.



Enter “EntryStream” as Input Alias.

Click on “Event Hub” drop down and select the one starting with “TollData” (e.g. TollData9518658221).

Select “entry” as Event Hub name and “all” as Event Hub policy name.

Your settings will look like:

ADD AN EXISTING SERVICE BUS EVENT HUB

Event Hub settings

INPUT ALIAS

EntryStream

SUBSCRIPTION

Use Event Hub from Current Subscription

SERVICE BUS NAMESPACE

TollData3337739128

EVENT HUB NAME ?

entry

EVENT HUB POLICY NAME ?

RootManageSharedAccessKey

12

←→4

Move to the next page. Leave default values (CSV, comma delimited, UTF8 encoding)

ADD AN EXISTING SERVICE BUS EVENT HUB

Serialization settings

EVENT SERIALIZATION FORMAT ?

CSV

DELIMITER ?

comma (,)

ENCODING ?

UTF8

Click OK at the bottom of the dialog to finish the wizard.

NAME	SOURCE TYPE	TYPE	CONNECTION STATUS	
EntryStream	Data stream	Event Hub	✓ Connected	

You will need to follow the same sequence of steps to create the second Event Hub input for the stream with Exit events. Make sure on the 3rd page you enter values as on the screenshot below.

1
2

ADD AN EXISTING SERVICE BUS EVENT HUB

Event Hub settings

INPUT ALIAS

ExitStream

SUBSCRIPTION

Use Event Hub from Current Subscription

SERVICE BUS NAMESPACE

TollData3337739128

EVENT HUB NAME ?

exit

EVENT HUB POLICY NAME ?

RootManageSharedAccessKey

←
→

4

You now have two input streams defined:

NAME	SOURCE TYPE	TYPE	CONNECTION STATUS	
EntryStream	Data stream	Event Hub	? Unknown	
ExitStream	Data stream	Event Hub	✓ Connected	

Next, we will add “Reference” data input for the blob file with car registration data.

Click “Add Input”. Select “Reference Data”

ADD AN INPUT

Add an input to your job

Each job requires at least one data stream input. Adding reference data input is optional.

☐ DATA STREAM

A continuous sequence of data or events to be consumed and transformed by a Stream Analytics job.

☒ REFERENCE DATA

Auxiliary data used for correlation and lookups. Reference data is static or slow-changing.

On the next page, select the storage account starting with “tolldata”. Container name should be “tolldata” and blob name should be “registration.csv”

ADD AN EXISTING BLOB STORAGE

Blob Storage Settings

INPUT ALIAS

Registration

SUBSCRIPTION

Use Storage Account from Current Subscription

STORAGE ACCOUNT

tolldata4637388511

STORAGE ACCOUNT KEY

CONTAINER ?

tolldata

BLOB NAME ?

registration.csv

1

3

Leave default values on the next page and click OK to finish the wizard.

ADD AN EXISTING BLOB STORAGE

Serialization settings

EVENT SERIALIZATION FORMAT ?

CSV

DELIMITER ?

comma (,)

ENCODING ?

UTF8

Now all inputs are defined.

NAME	SOURCE TYPE	TYPE	CONNECTION STATUS
EntryStream	Data stream	Event Hub	✓ Connected
ExitStream	Data stream	Event Hub	✓ Connected
Registration	Reference data	Blob storage	✓ Connected

DEFINE OUTPUT

Go to “Output” tab and click “Add an output”.

You have no output. Add one to get started!

[ADD AN OUTPUT](#) ➔

Choose “Sql Database”.

Select the server name that was used in “Connect to Database from Visual Studio”. The database name should be TollDataDB.

Enter “tolladmin” as the user name and “123toll!” as the password. The table name should be set to “TollDataRefJoin”

EDIT SETTINGS

SQL Database Settings

SERVER NAME
[redacted]database.windows.net

DATABASE
TollDataDB

USERNAME
tolladmin

PASSWORD
[masked]

TABLE
TollDataRefJoin

✓

AZURE STREAM ANALYTICS QUERY

The Query tab contains a SQL query that performs the transformation over the incoming data.

tolldata PREVIEW

DASHBOARD MONITOR INPUTS QUERY OUTPUT SCALE CONFIGURE

query

```
1 SELECT * FROM [your Input Alias]
```

?

Through this lab we will attempt to answer several business questions related to Toll data and construct Stream Analytics queries that can be used in Azure Stream Analytics to provide a relevant answer.

Before we start our first Azure Stream Analytics job, let's explore few scenarios and query syntax.

6. INTRODUCTION TO AZURE STREAM ANALYTICS QUERY LANGUAGE

Let's say, we need to count the number of vehicles that enter a toll booth. Since this is a continuous stream of events, it is essential we define a "period of time". So we need to modify our question to be "Number of vehicles entering a toll booth every 3 minutes". This is commonly referred to as the Tumbling Count.

Let's look at the Azure Stream Analytics query answering this question:

```
SELECT TollId, System.Timestamp AS WindowEnd, COUNT(*) AS Count
FROM EntryStream TIMESTAMP BY EntryTime
GROUP BY TUMBLINGWINDOW(minute,3), TollId
```

As you can see, Azure Stream Analytics is using a SQL-like query language with a few additional extensions to enable specifying time related aspects of the query.

The next sections will describe Timestamp and TumblingWindow constructs used in the query in detail.

APPLICATION TIME - TIMESTAMP

In any temporal system like Azure Stream Analytics, it's essential to understand the progress of time. Every event that flows through the system comes with a timestamp. In other words, every event in the system depicts a point in time.

This timestamp can be defined by the Application (e.g. specific field in the event). The user can specify it in the query using the "TIMESTAMP BY" clause (e.g. EntryTime column in our example).

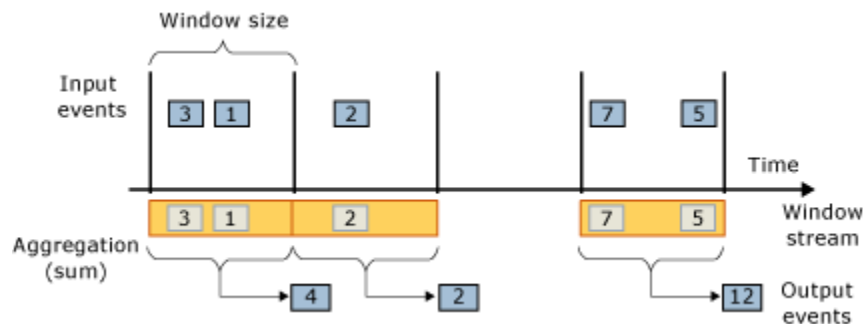
Alternatively, the system can assign the timestamp based on the event arrival time. Please note that in this case the value of the timestamp is potentially subject to certain factors such as network latencies. This can negatively affect accuracy of the further computations and analysis.

You can access the *timestamp* of any event using the *System.Timestamp* property which needs to be used with an alias. For any aggregate event like Sum(), Avg(), etc., produced as output from a window operation, it will also have a *timestamp* property like any other input event entering into the system. This timestamp of the output event aligns to the end time of the window.

TIME WINDOWS – TUMBLING WINDOW

In applications that process real-time events, a common requirement is to perform some set-based computation (aggregation) or other operations over subsets of events that fall within some period of time. In Azure Stream Analytics, these subsets of events are defined through windows.

A window contains event data along a timeline and enables you to perform various operations against the events within that window. For example, you may want to sum the values of payload fields in a given window as shown in the following illustration.



Azure Stream Analytics supports several functions to define window aggregates (Tumbling, Hopping, Sliding windows). Please refer to the Query Language Reference Guide in MSDN for more details.

Our sample query uses TumblingWindow function to specify the size of the window

```
GROUP BY TUMBLINGWINDOW (minute, 3)
```

7. TESTING AZURE STREAM ANALYTICS QUERIES

Now that we have written our first Azure Stream Analytics query, it is time to test it out using sample files here: <http://1drv.ms/1wmM9QZ>

Download and extract TollAppData.zip to your local machine. This contains the following files:

1. Entry.json
2. Exit.json
3. Registration.json

QUESTION 1 - NUMBER OF VEHICLES ENTERING A TOLL BOOTH

Open the Azure Management portal and navigate to your created Azure Stream Analytic job. Open the Query tab and copy paste Query from the previous section.

 DASHBOARD MONITOR INPUTS **QUERY** OUTPUT SCALE CONFIGURE

query

```
1 SELECT TollId, System.Timestamp AS WindowEnd, COUNT(*)AS Count
2 FROM EntryStream TIMESTAMP BY EntryTime
3 GROUP BY TUMBLINGWINDOW(minute,3), TollId
4
```

Test

Rerun

To validate this query against sample data, click the Test button. In the dialog that opens, navigate to Entry.json, a file with sample data from the EntryTime event stream.

TEST DATA ?

Input

Sample File

entrystream



Entry.json

Validate that the output of the query is as expected:

output

15 rows were generated by processing:

- 23 events from entristream

TOLLID	WINDOWEND	COUNT
1	2014-09-10T12:03:00.000Z	3
2	2014-09-10T12:03:00.000Z	1
3	2014-09-10T12:03:00.000Z	1
2	2014-09-10T12:06:00.000Z	2
1	2014-09-10T12:09:00.000Z	1
2	2014-09-10T12:09:00.000Z	3
3	2014-09-10T12:09:00.000Z	2
1	2014-09-10T12:12:00.000Z	2
3	2014-09-10T12:12:00.000Z	2



SAVE



DISCARD

QUESTION 2 - REPORT TOTAL TIME FOR EACH CAR TO PASS THROUGH THE TOLL BOOTH

We want to find average time required for the car to pass the toll to assess efficiency and customer experience.

For this we need to join the stream containing EntryTime with the stream containing ExitTime. We will join the streams on TollId and LicencePlate columns. JOIN operator requires specifying a temporal wiggle room describing acceptable time difference between the joined events. We will use DATEDIFF function to specify that events should be no more than 15 minutes from each other. We will also apply DATEDIFF function to Exit and Entry times to compute actual time a car spends in the toll. Note the difference of the use of DATEDIFF when used in a SELECT statement compared to a JOIN condition.

```

SELECT EntryStream.TollId, EntryStream.EntryTime,
ExitStream.ExitTime, EntryStream.LicensePlate, DATEDIFF(minute,
EntryStream.EntryTime, ExitStream .ExitTime) AS
DurationInMinutes
FROM EntryStream TIMESTAMP BY EntryTime
JOIN ExitStream TIMESTAMP BY ExitTime
ON (EntryStream.TollId= ExitStream.TollId AND
EntryStream.LicensePlate = ExitStream.LicensePlate)
AND DATEDIFF(minute, EntryStream, ExitStream ) BETWEEN 0 AND 15

```

To test this query, update the query on the Query tab of your job:

tolldata PREVIEW

 DASHBOARD  MONITOR  INPUTS **QUERY**  OUTPUT  SCALE  CONFIGURE

query



```

1 SELECT EntryStream.TollId, EntryStream.EntryTime, ExitStream.ExitTime, EntryStream.LicensePlate, DATEDIFF
2 FROM EntryStream TIMESTAMP BY EntryTime
3 JOIN ExitStream TIMESTAMP BY ExitTime
4 ON (ExitStream.TollId= ExitStream.TollId AND EntryStream.LicensePlate = ExitStream.LicensePlate)
5 AND DATEDIFF(minute, EntryStream, ExitStream ) BETWEEN 0 AND 15
6
7

```

Click test and specify sample input files for EntryTime and ExitTime.

TEST DATA 

Input	Sample File
entrystream	 Entry.json
exitstream	 Exit.json

Click the checkbox to test the query and view output:

output

23 rows were generated by processing:

- 23 events from entrystream
- 23 events from exitstream

TOLLID	ENTRYTIME	EXITTIME	LICENSEPLATE	DURATIONINMINUTES
1	2014-09-10T12:01:00.000Z	2014-09-10T12:03:00.000Z	JNB 7001	2
3	2014-09-10T12:02:00.000Z	2014-09-10T12:04:00.000Z	ABC 1004	2
1	2014-09-10T12:02:00.000Z	2014-09-10T12:03:00.000Z	YXZ 1001	1
2	2014-09-10T12:03:00.000Z	2014-09-10T12:07:00.000Z	XYZ 1003	4
1	2014-09-10T12:03:00.000Z	2014-09-10T12:08:00.000Z	BNJ 1007	5
2	2014-09-10T12:05:00.000Z	2014-09-10T12:07:00.000Z	CDE 1007	2
2	2014-09-10T12:06:00.000Z	2014-09-10T12:09:00.000Z	BAC 1005	3
1	2014-09-10T12:07:00.000Z	2014-09-10T12:10:00.000Z	ZYX 1002	3

QUESTION 3 – REPORT ALL COMMERCIAL VEHICLES WITH EXPIRED REGISTRATION

Azure Stream Analytics can use static snapshots of data to join with temporal data streams. To demonstrate this capability we will use the following sample question.

If a commercial vehicle is registered with the Toll Company, they can pass through the toll booth without being stopped for inspection. We will use Commercial Vehicle Registration lookup table to identify all commercial vehicles with expired registration.

```
SELECT EntryStream.EntryTime, EntryStream.LicensePlate,  
EntryStream.TollId, Registration.RegistrationId  
FROM EntryStream TIMESTAMP BY EntryTime  
JOIN Registration  
ON EntryStream.LicensePlate = Registration.LicensePlate  
WHERE Registration.Expired = '1'
```

Note that testing a query with Reference Data requires that an input source for the Reference Data is defined, which we have done in Step 5.

To test this query, paste the query into the Query tab, click Test, and specify the 2 input sources:

TEST DATA ?

Input

Sample File

entrystream



Entry.json

registration



Registration.json

View the output of the query:

output

2 rows were generated by processing:

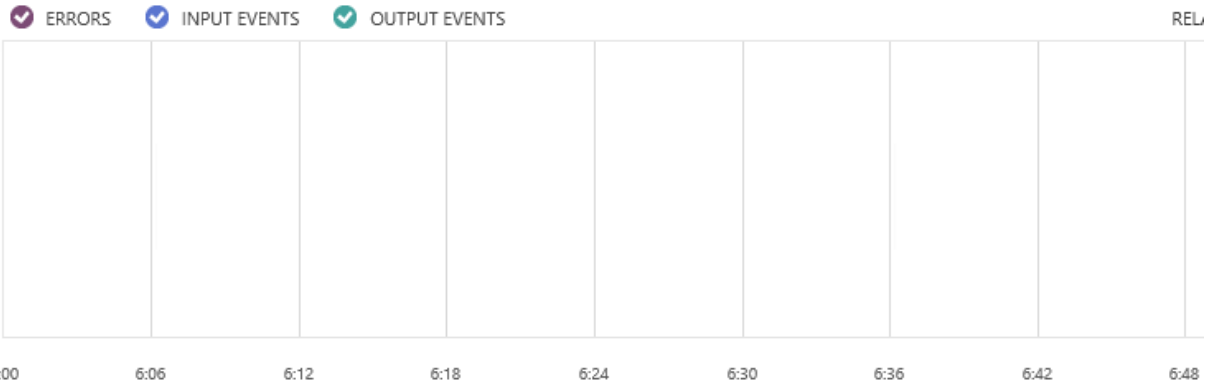
- 23 events from entrystream
- 10000 events from registration

ENTRYTIME	LICENSEPLATE	TOLLID	REGISTRATIONID	
2014-09-10T12:06:00.000Z	BAC 1005	2	876133137	
2014-09-10T12:15:00.000Z	BAC 1005	2	876133137	

8. START THE STREAM ANALYTICS JOB

Now as we have written our first Azure Stream Analytics query, it is time to finish the configuration and start the job. Save the query from Question 3, which will produce output that matches the schema of our output table TollDataRefJoin.

Navigate to the job Dashboard and click Start.



In the dialog that appears, change the Start Output time to Custom Time. Edit the Hour and set it to an hour back. This will ensure that we process all events from the Event Hub since the moment we started generating the events in the beginning of the lab. Now click the check mark to start the job.

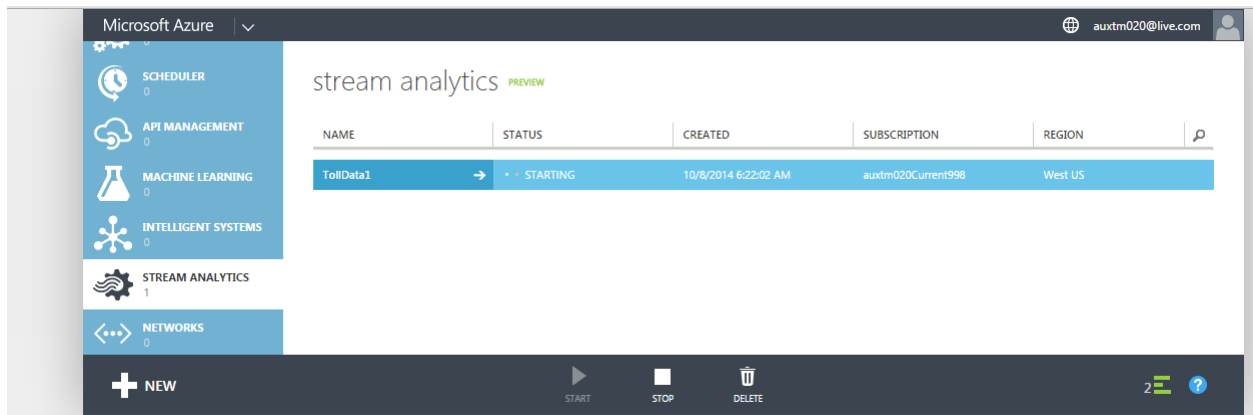
START OUTPUT



:
 :

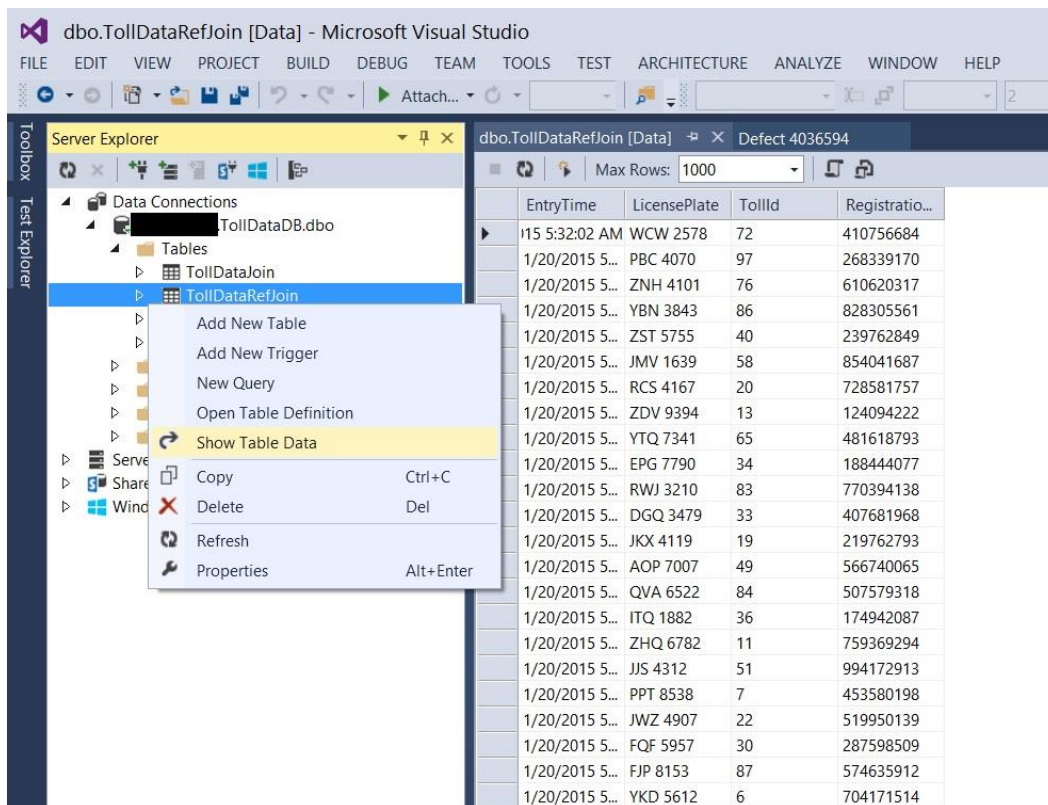
LOCAL TIME (UTC-08:00)

Starting the job can take a few minutes. You will be able to see the status on the top-level page for Stream Analytics.



CHECK RESULTS IN VISUAL STUDIO

Open Visual Studio Server Explorer and right click TollDataRefJoin table. Select “Show Table Data” to see the output of your job.

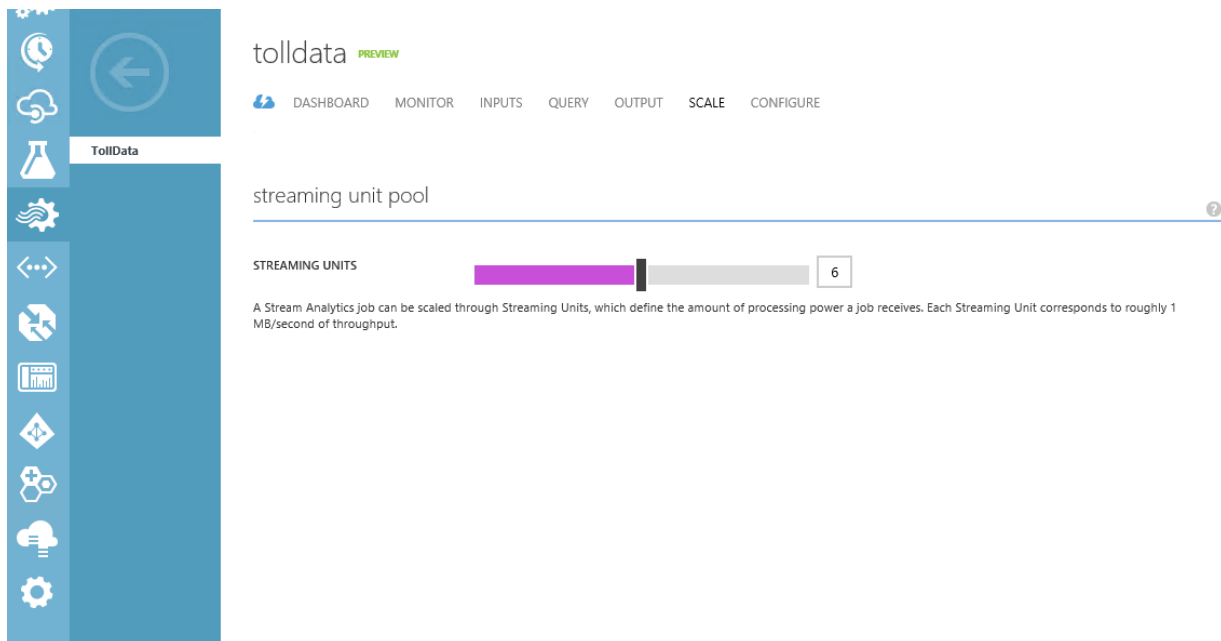


9. SCALING OUT AZURE STREAM ANALYTICS JOBS

Azure Stream Analytics is designed to elastically scale and be able to handle high load of data. The Azure Stream Analytics query can use a **PARTITION BY** clause to tell the system that this step will scale out. PartitionId is a special column added by the system that matches the partition id of the input (Event Hub)

```
SELECT TollId, System.Timestamp AS WindowEnd, COUNT(*) AS Count
FROM EntryStream TIMESTAMP BY EntryTime PARTITION BY
PartitionId
GROUP BY TUMBLINGWINDOW(minute,3), TollId
```

Stop the current job, update the query in Query tab and open Scale tab. Streaming units define the amount of compute power the job can receive. Move the slider to 6.



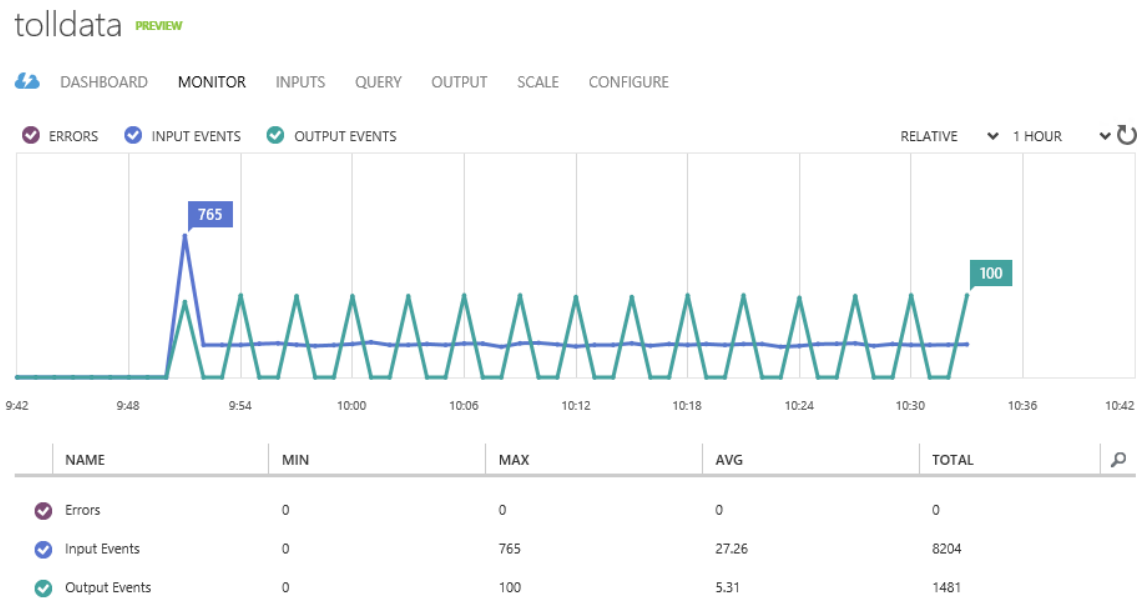
The screenshot shows the Azure Stream Analytics job configuration interface for a job named 'tolldata'. The 'SCALE' tab is selected, displaying the 'streaming unit pool' configuration. A slider is shown with the value set to 6. Below the slider, a note states: 'A Stream Analytics job can be scaled through Streaming Units, which define the amount of processing power a job receives. Each Streaming Unit corresponds to roughly 1 MB/second of throughput.'

Please go to the output tab and change SQL table name to “TollDataTumblingCountPartitioned”

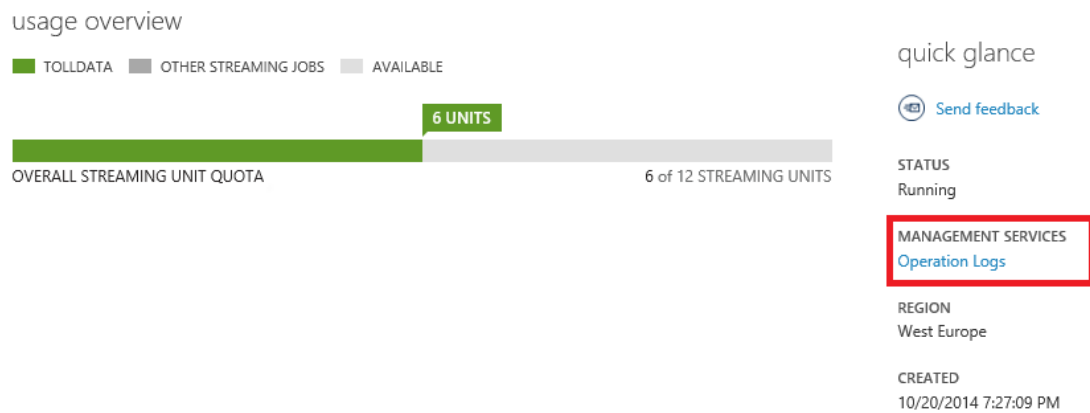
Now, if you start the job, Azure Stream Analytics will be able to distribute work across more compute resources and achieve better throughput. Please note that TollApp application is also sending events partitioned by TollId.

10. MONITORING

Monitoring tab contains statistics about the running job.



You can access Operation Logs from the Dashboard tab.



Microsoft Azure | Subscriptions

management services

ALERT OPERATION LOGS

SUBSCRIPTION StreamAnalyticsPublicPr... FROM 2014-10-19 11:00 PM TO 2014-10-20

STATUS All TYPE Stream Analytics SERVICE NAME TollData

Click the checkmark button to execute the query.

TIME STAMP	OPERATION	STATUS	SERVICE NAME	TYPE
10/20/2014 11:04:44 PM	Start job 'TollData'	Failed	TollData	Microsoft.StreamAn
10/20/2014 11:04:43 PM	Start streaming job 'TollData'	Started	TollData	Microsoft.StreamAn
10/20/2014 11:04:43 PM	Start streaming job 'TollData'	Completed	TollData	Microsoft.StreamAn
10/20/2014 11:04:42 PM	Microsoft.StreamAnalytics/StreamingJobs/start/action	Accepted	TollData	Microsoft.StreamAn
10/20/2014 11:04:42 PM	Microsoft.StreamAnalytics/StreamingJobs/start/action	Started	TollData	Microsoft.StreamAn
10/20/2014 11:04:38 PM	Start streaming job 'TollData'	Completed	TollData	Microsoft.StreamAn
10/20/2014 11:04:38 PM	Microsoft.StreamAnalytics/StreamingJobs/start/action	Accepted	TollData	Microsoft.StreamAn

+ NEW

DETAILS

To see additional information about a particular event, select the event and click “Details” button.

OPERATION DETAILS

```

JobFailedMessage:
Stream Analytics job has validation errors: The input alias 'entrystreammissing',
JobId:
a9fcbbf5-d113-4e06-9c3b-2b9fc9987125
JobRunCreatedDateTime:
2014-10-20 23:04:38Z
JobRunId:
3b6ac8d8-8b1e-44ff-974d-39f5d1b367a9
JobRunLastUpdateDateTime:
2014-10-20 23:04:43Z
JobRunStatus:
Failed
Microsoft.Resources/EventNameV2:
Start job 'TollData'
Microsoft.Resources/Operation:
Start job 'TollData'

```

COPY TO CLIPBOARD

11. ORDERING EVENTS

The order of the events is very important for the Stream Analytics query to produce correct answers. Since all the events are transmitted over the network, there is no guarantee that they will arrive in the same order as they were sent. There are two types of policies that can be used to specify how out of order events should be handled. They are Adjust (default) and Drop.

ADJUST POLICY (DEFAULT)

The Adjust Policy is the default policy setting for an Azure Stream Analytics job. In this mode, the *timestamp* of any late arriving out of order event is changed and adjusted to the *timestamp* of the last received event.

DROP POLICY

Setting the policy as Drop instructs Stream Analytics to drop all events that occur out of order. This is especially useful if the job needs to prioritize low latency over absolute correctness.

TOLERANCE WINDOW

There are cases where the amount of expected disorder can be estimated by the user. For example if you think your events are not going to be delayed for more than 30 sec, you can set that as your Tolerance Window and provide a leeway for your incoming events. In this case all events arriving within 30 seconds will be sent to the engine in proper order. It is important to understand that you are introducing a latency for the amount of time you set in the tolerance window and if you have a high throughput of events, you are also adding memory pressure.

The screenshot shows the 'CONFIGURE' tab of an Azure Stream Analytics job named 'sampleexample'. The left sidebar contains a navigation menu with '114test' and 'SampleExample' (selected), and a list of demo jobs: 'Demo', 'fdsjakl', 'mondemo', 'CSVtoJSON', and 'BlobEgress'. The main configuration area is titled 'out of order policy' and includes the following settings:

- TOLERANCE WINDOW (MM:SS):** Set to 00 : 00.
- ACTION:** Two buttons, 'ADJUST' (selected) and 'DROP'.
- locale:** A section header.
- LOCALE:** A dropdown menu set to 'en-US'.
- regional monitoring storage account:** A section header.
- STORAGE ACCOUNT:** A dropdown menu set to 'nrtstorage2'.

A warning message states: 'All Stream Analytics jobs in the same region share the same monitoring storage account. Changing it will affect all the jobs in the same region.'

12. CONCLUSION

In this lab we provided introduction to the Azure Stream Analytics service. We demonstrated how to configure inputs and outputs for the Stream Analytics job. Using the Toll Data sample scenario we explained common types of problems that arise in the space of data in motion and how they can be solved with simple SQL-like queries in Azure Stream Analytics. We described SQL extension constructs for working with temporal data. We showed how to join data streams, how to enrich the data stream with static reference data. We explained how to scale out a query to achieve higher throughput.

While this lab provides good introductory coverage, it is not complete by any means. Please refer to documentation on MSDN to learn more about Azure Stream Analytics.

13. CLEANUP YOUR AZURE ACCOUNT

Please stop the Stream Analytics Job from the Azure Portal.

Setup.ps1 script creates 2 Azure Event Hubs and Azure SQL database server. The following instructions will help you to cleanup resources at the end of the lab.

In PowerShell window type “.\Cleanup.ps1” This will start the script that deletes resources used in the lab.

Please note, that resources are identified by the name. Make sure you carefully review each item before confirming removal.

```
PS C:\TollApp\LabSetup> .\Cleanup.ps1
WARNING: This script is going to delete resources that match resource names used in the lab. Please carefully review
names of the resources before confirming delete operation
Remove Service Bus namespaces starting with 'TollData'
```

PROVISIONING EVENT HUB INPUT SOURCES

We will use Azure Event Hub to receive input data streams with the toll data.

Log in to the Azure portal (<http://manage.windowsazure.com>) and create a new Service Bus namespace in West Europe region that is unique. Specify that it is for Messaging and select Standard Tier.

Note: Event Hub is not present in all Data Centers. If you pick a DC for the namespace where EventHub currently not available, you will not be able to create an Event Hub.

CREATE A NAMESPACE

Add a new namespace

NAMESPACE NAME

TollData12345 ✓

.servicebus.windows.net

REGION

West Europe ▼

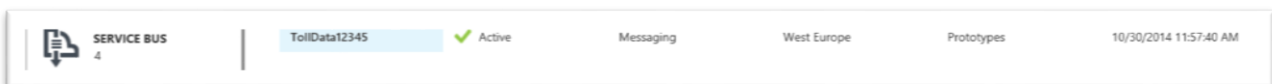
TYPE ?

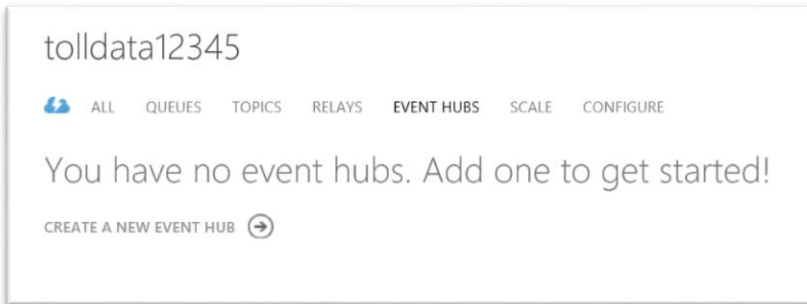
MESSAGING NOTIFICATION HUB

MESSAGING TIER ?

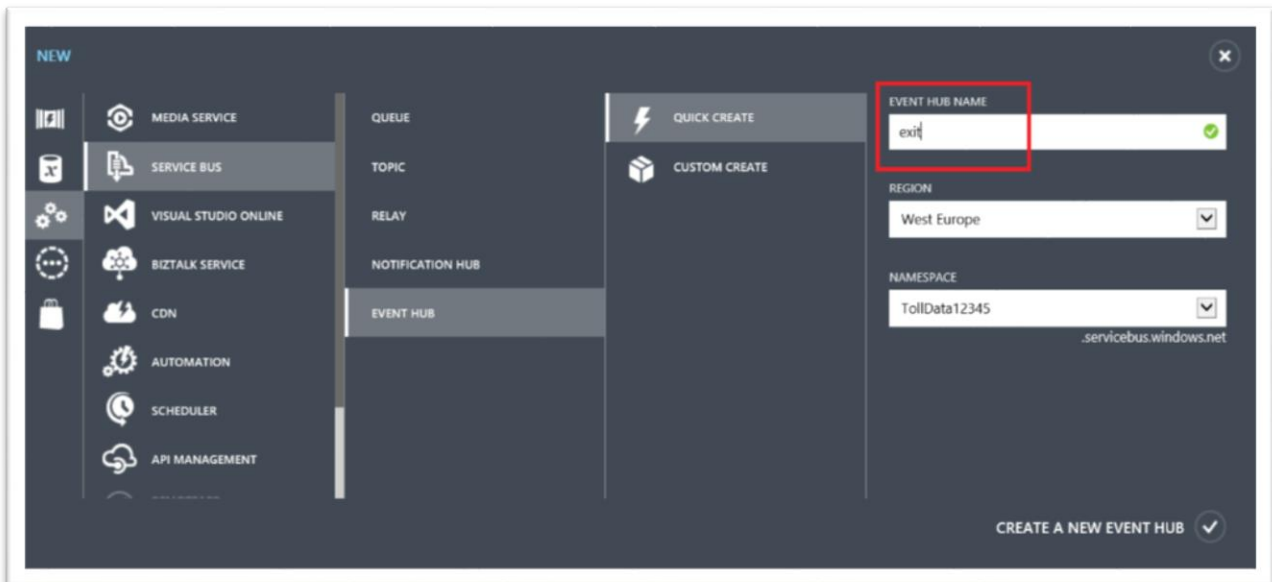
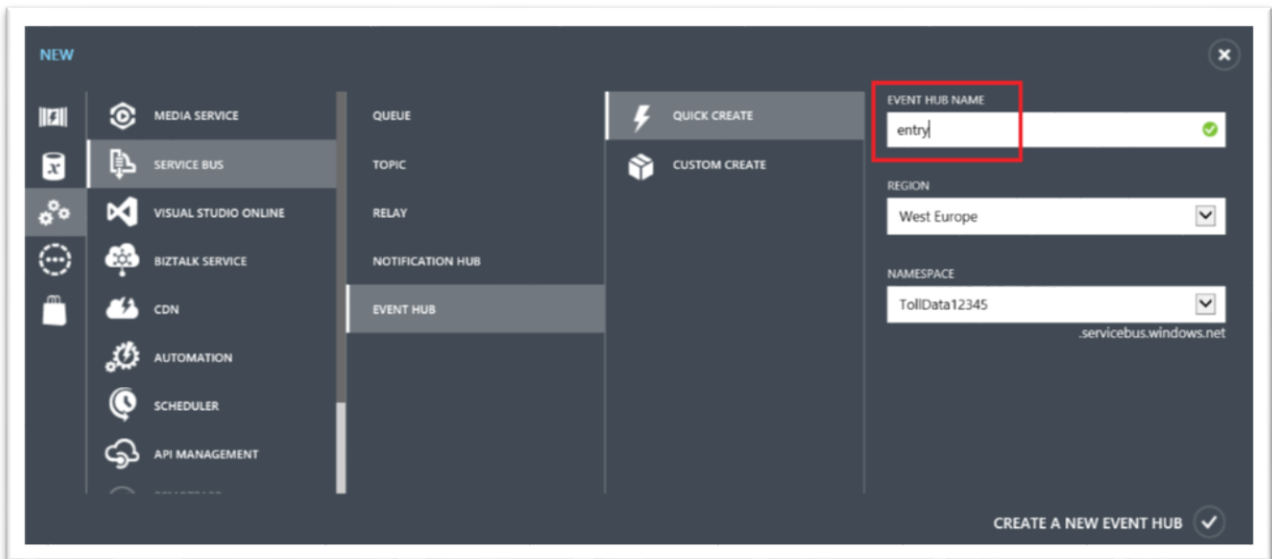
BASIC STANDARD

With the Service Bus namespace selected, click on it and go to the Event Hub tab to now create the two Event Hubs.





Now create two Event Hubs with the name entry and exit.



Now you should see both your event hubs created. Go into each event hub and create its shared access policy.

tolldata12345		
ALL QUEUES TOPICS RELAYS EVENT HUBS SCALE CONFIGURE		
NAME	STATUS	PARTITION COUNT
entry	→ ✓ Active	16
exit	→ ✓ Active	16

You should select “Manage” for the shared access policy.

entry

DASHBOARD CONFIGURE

entry

exit

general

MESSAGE RETENTION 1 days

EVENT HUB STATE Enabled

PARTITION COUNT 16 Partitions

shared access policies

NAME	PERMISSIONS
All	Manage, Send, Listen
NEW POLICY NAME	<input checked="" type="checkbox"/> Manage <input type="checkbox"/> Send <input type="checkbox"/> Listen

NEW

SAVE DISCARD

Repeat the same process to create the shared access policy for the second event hub.

PROVISIONING A WINDOWS AZURE SQL DATABASE

Stream Analytics job will process the input data and produce the output stream. We will be outputting data into Sql Azure Database in this lab.

In the Azure Management Portal click Sql Databases and click New at the bottom of the page.

Enter a name for the database as TollDataDB and select your subscription if asked.

Choose New SQL database server as the Server and set Region to “West Europe”.

Enter “tolladmin” as login name and “streamanalytics2014!” as the password.

The screenshot shows the Azure portal's 'NEW' page. On the left, there's a navigation pane with categories like COMPUTE, DATA SERVICES, APP SERVICES, NETWORK SERVICES, and STORE. The 'SQL DATABASE' section is selected. In the center, there are options for STORAGE, HDINSIGHT, RECOVERY SERVICES, MACHINE LEARNING, and STORSIMPLE MANAGER. On the right, there's a form to create a new SQL database. The form fields are: DATABASE NAME (TollDataDB), SUBSCRIPTION (StreamingTestPool_655453 (d4701118-d4b7-4a...)), SERVER (New SQL database server), REGION (West Europe), LOGIN NAME (tolldata), and CONFIRM PASSWORD (masked with dots). At the bottom right, there's a 'CREATE SQL DATABASE' button with a checkmark icon.

Click on the TollDataDB database you just created. You are now in the database overview page for your new database.

tolladatdb

 DASHBOARD  MONITOR  SCALE  CONFIGURE  GEO-REPLICATION  AUDITING & SECURITY PREVIEW



You created a new SQL database

Here are a few options to get you started

☐ Skip Quick Start the next time I visit



Get Microsoft database design tools 


[Install Microsoft SQL Server Data Tools](#)



Design your SQL database 

[Download a starter project for your SQL database](#) [Set up Windows Azure firewall rules for this IP address](#) [Download the Elastic Scale APIs preview](#)



Connect to your database 

[Design your SQL database](#) [Run Transact-SQL queries against your SQL database](#) [View SQL Database connection strings for ADO .Net, ODBC, PHP, and JDBC](#)

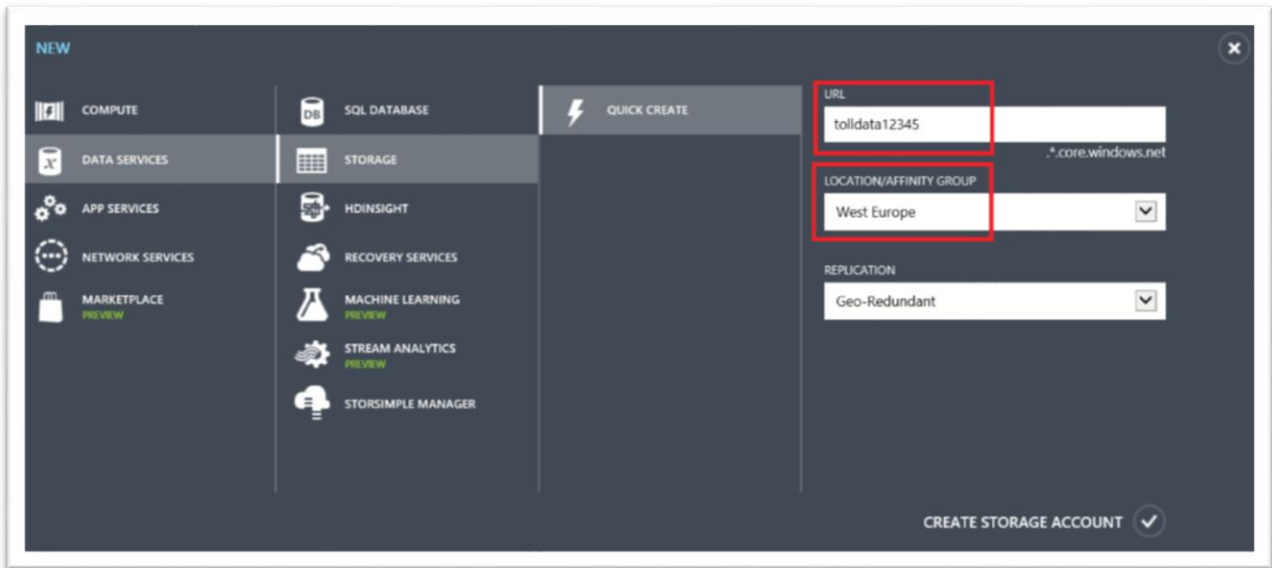
Server: `acs4y0uuq9.database.windows.net,1433`

In order for your computer to access the SQL Database, you need to open a firewall connection.

Click on the Set up Windows Azure firewall rules for this IP address link and add a rule with IP range 0.0.0.0 - 255.255.255.255

PROVISION A WINDOWS AZURE STORAGE ACCOUNT

Go to the Storage service and create a new Storage account.

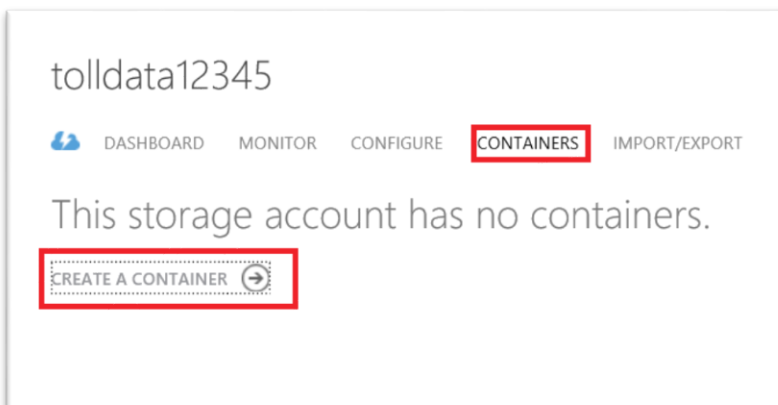


The screenshot shows the Azure portal's 'NEW' page. On the left, the 'STORAGE' service is selected under 'DATA SERVICES'. On the right, the 'QUICK CREATE' form is displayed with the following fields:

- URL:** tolldata12345
- LOCATION/AFFINITY GROUP:** West Europe
- REPLICATION:** Geo-Redundant

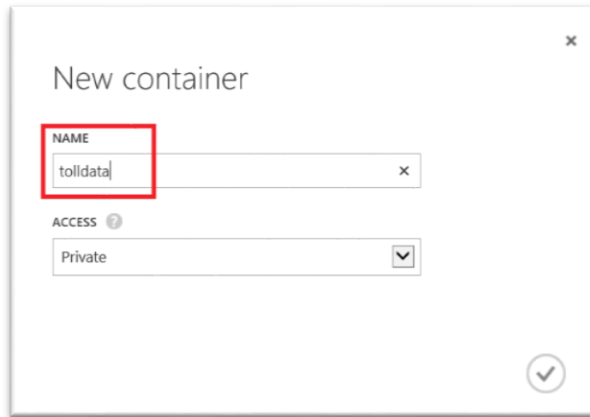
The 'CREATE STORAGE ACCOUNT' button is located at the bottom right of the form.

Now click on the Storage account and go to Containers tab to create a new container.



The screenshot shows the 'Containers' tab for the 'tolldata12345' storage account. The navigation bar includes 'DASHBOARD', 'MONITOR', 'CONFIGURE', 'CONTAINERS' (which is highlighted), and 'IMPORT/EXPORT'. The main content area displays the message 'This storage account has no containers.' and a 'CREATE A CONTAINER' button with a right-pointing arrow icon.

Create a container named tolldata.



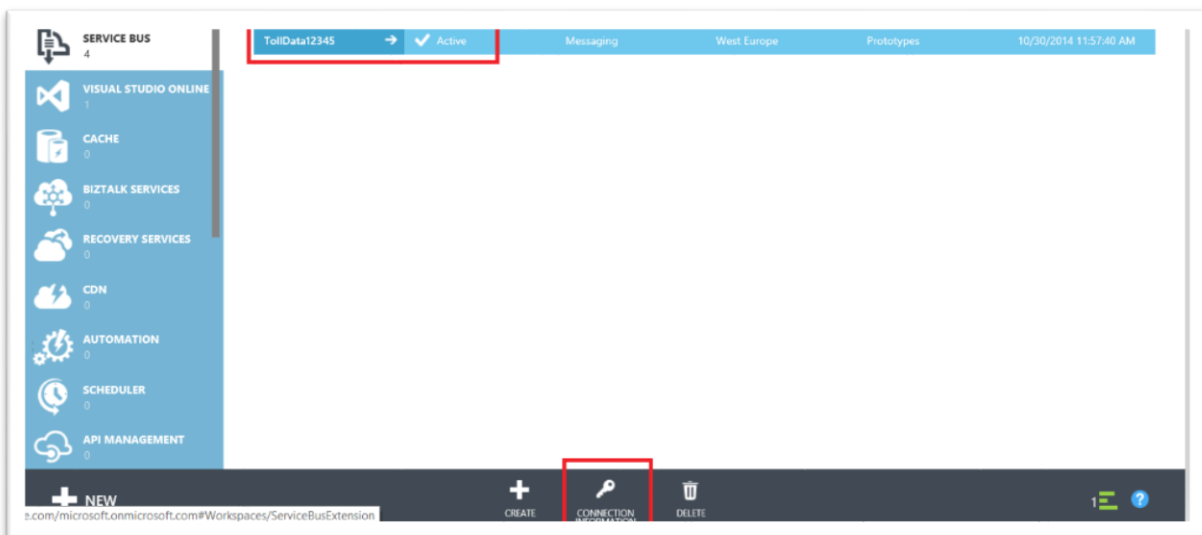
A dialog box titled "New container" with a close button (X) in the top right corner. It contains two input fields: "NAME" and "ACCESS". The "NAME" field has a red border and contains the text "tolldata". The "ACCESS" field is a dropdown menu with "Private" selected. A checkmark icon is in the bottom right corner.

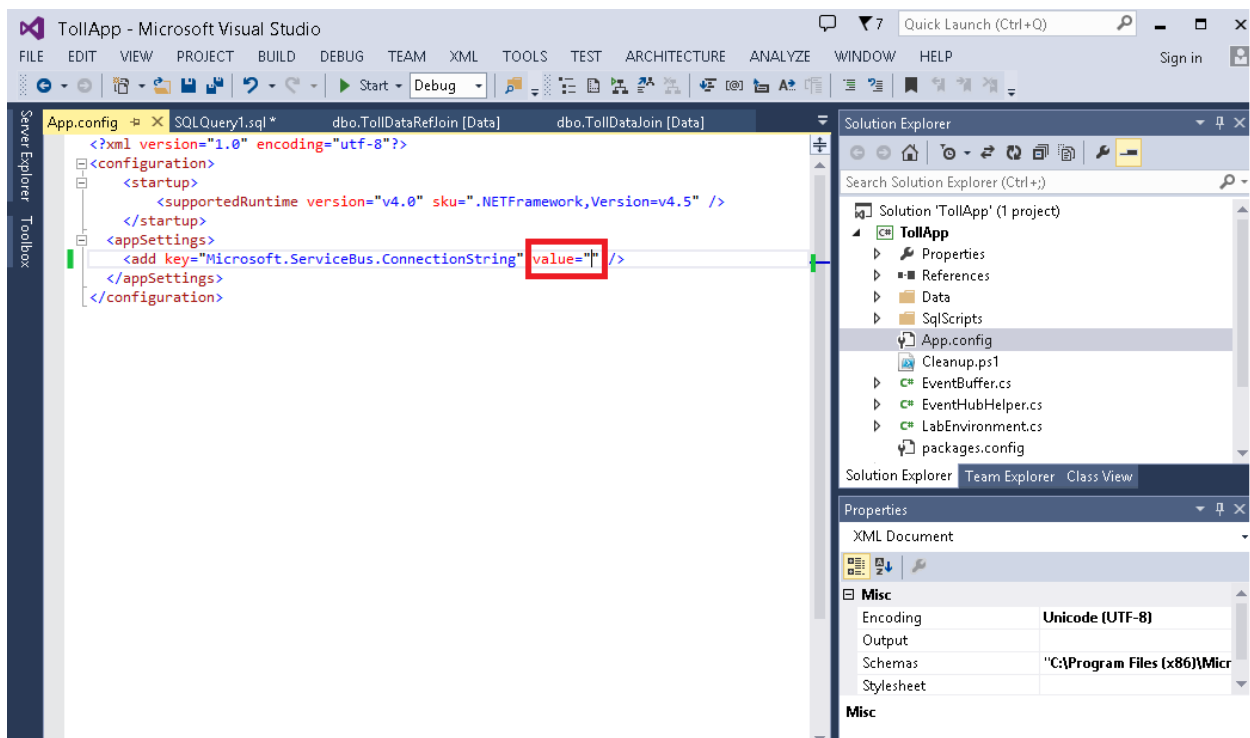
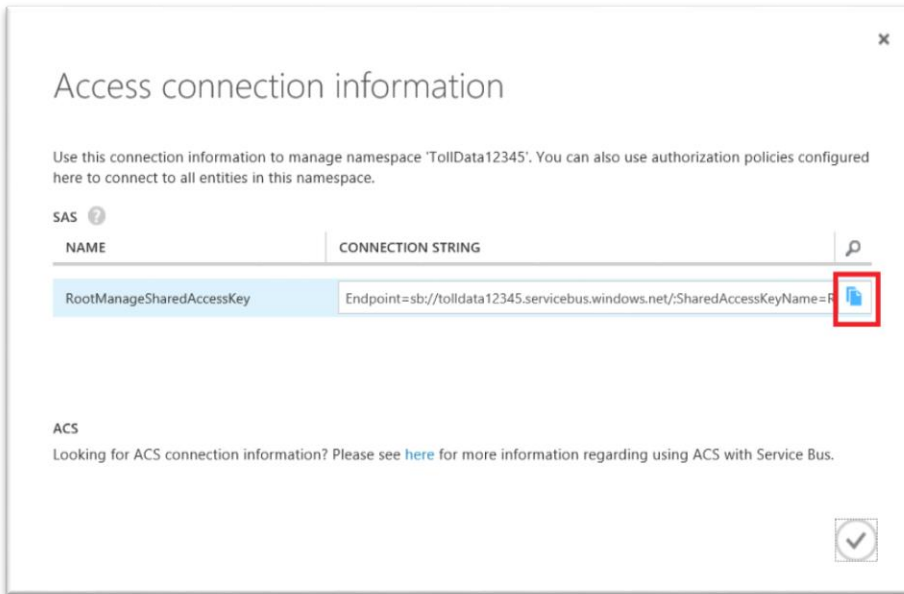
Use your favorite tool like [Azure Storage Explorer](#) to upload the registration.csv file to this container.

STARTING THE TOLL BOOTH EVENT GENERATOR

Open your Visual Studio solution for the TollApp project and go to the App.config file.

Copy Service Bus Root connection string and copy it in the App.config file.





Now start the application to start pushing Toll Booth events to Event Hub.