Microsoft IoT Hackathon DX

Scenario 2 – Gaming

# Overview

This scenario is designed to show you how to use the Windows Universal Platform background apps running on Windows 10 Core IoT, using different Azure cloud services. The solution encompasses Azure IoT Hubs, Azure Storage, Azure Stream Analytics, Azure SQL Database, Azure API Apps, Azure Data Factory, Azure ML and PowerBI.

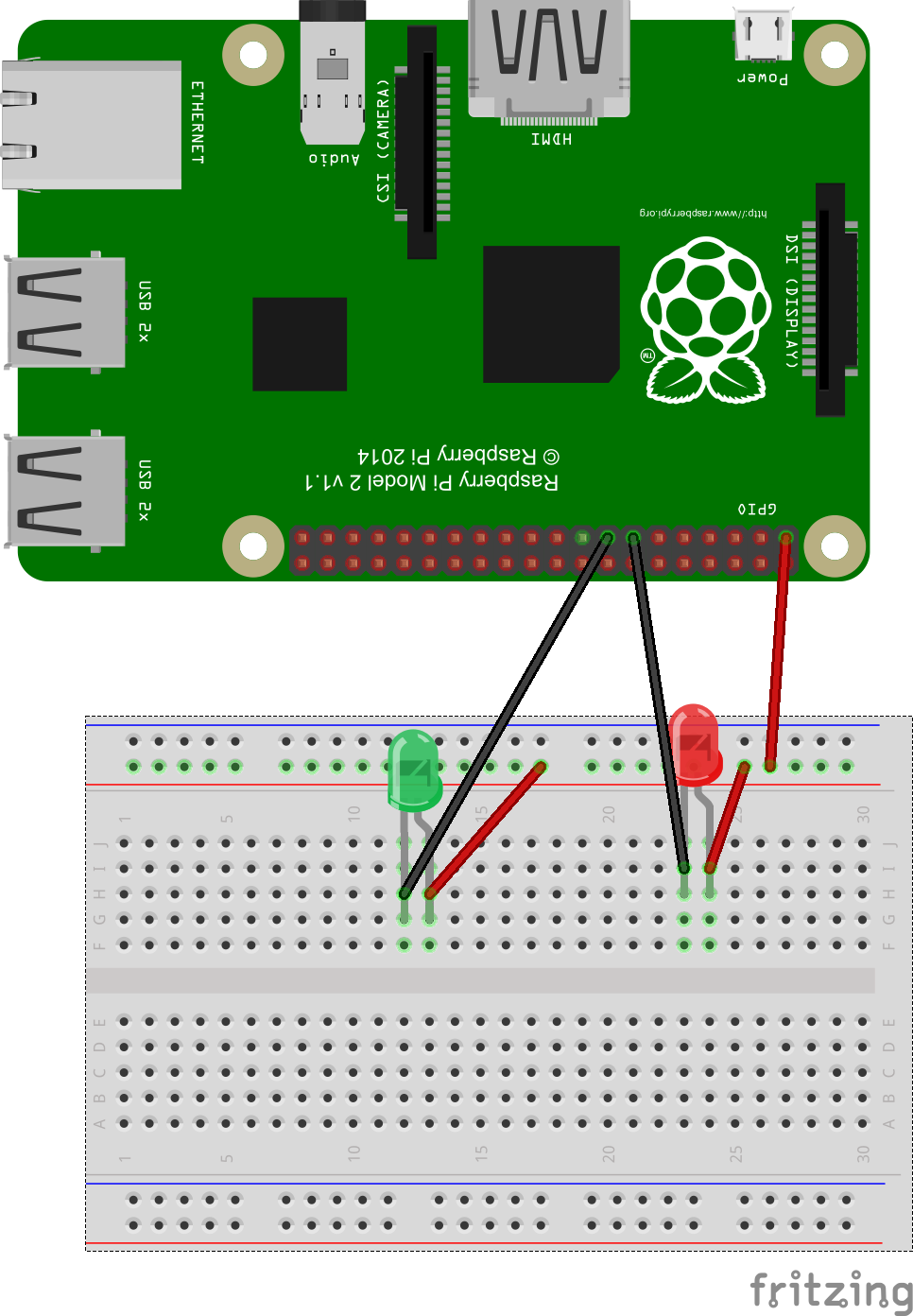
To begin with you will play a short quiz hosted on a communal Raspberry Pi. Use the IP Address you have been given to start playing the quiz.

Once the Quiz has been completed you will be shown **xyz** – need to decide how this is going to be demoed.

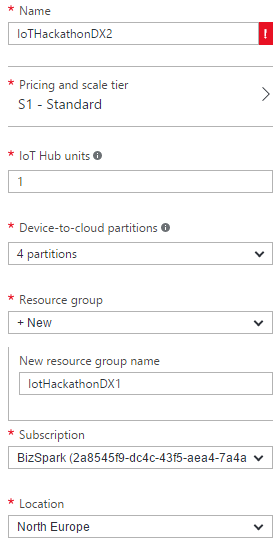
You will now have an opportunity to set up this scenario yourself.

# Circuit Diagram

Wire up your Raspberry Pi by following the circuit diagram.



# Raspberry Pi

Ensure that your Raspberry Pi now has the Windows 10 IoT Core SD card inserted, connect the Raspberry Pi to your computer with a network cable power the Pi on. If you haven’t already **download the Windows 10 IoT Core Dashboard** from <http://ms-iot.github.io/content/en-US/Downloads.htm> and install it.

# Provision the IoT Hub

For this scenario we are going to provision two IoT Hubs. The first hub will be used to send the player profile data to the Stream Analytics task. The second hub will be used to send the player answer data to the Stream Analytics task.

Follow steps 1 through 8 and then repeat. Configure your two hubs with the same name but suffix the second IoT Hub with “-profiles”.

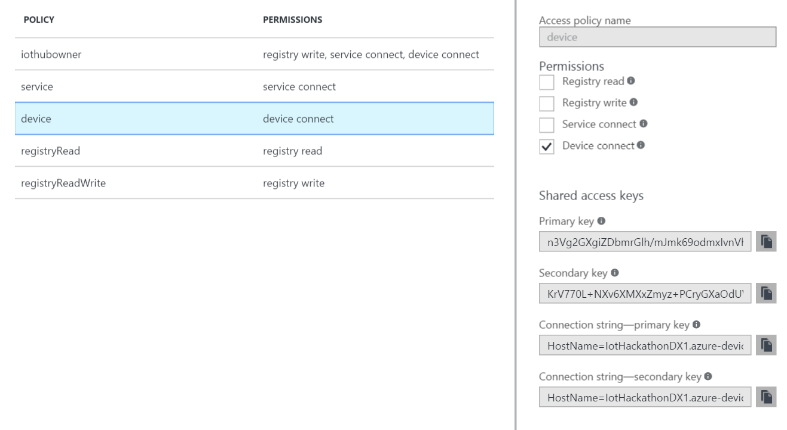
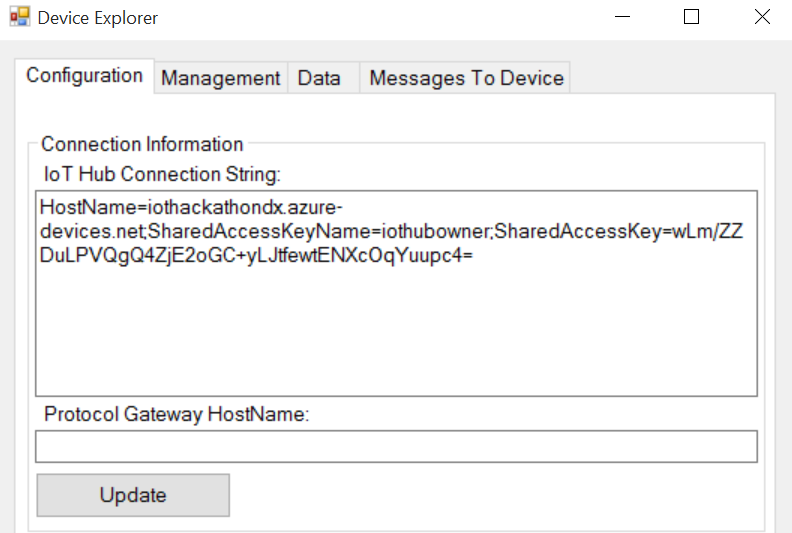
1. On the Preview Portal (remember you need to be in the Preview Portal), click **“+ NEW”** (top left) and select **Internet of Things.**
2. Select **IoT Hub** from the featured apps shown.
3. Enter a name for your hub. In this example I am using “IoTHackathonDX2”.
4. The IoT Hub is charged based on the number of messages per day you want to send to the hub. There is a **“free”** tier in this service that gives you 8,000 messages/day at no charge – select this option unless you are already using the **“free”** tier, in which case select “**S1 – Standard**”.

**Each Azure subscription may only have one free tier IoT Hub provisioned**

1. Leave the IoT Hub units as 1 and the number of partitions as 4.
2. Create a new resource group by clicking **“or create new”** and enter the name you want to call this resource group. We will re-use this resource group for provisioning additional services later on in this lab.
3. For the location select a location local to you where possible.
4. Click the **create button**. You will now be taken back to your Azure dashboard where you will see a new dashboard tile appear showing the status of the provision. IoT Hubs can take a few minutes to provision.

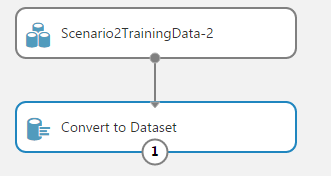
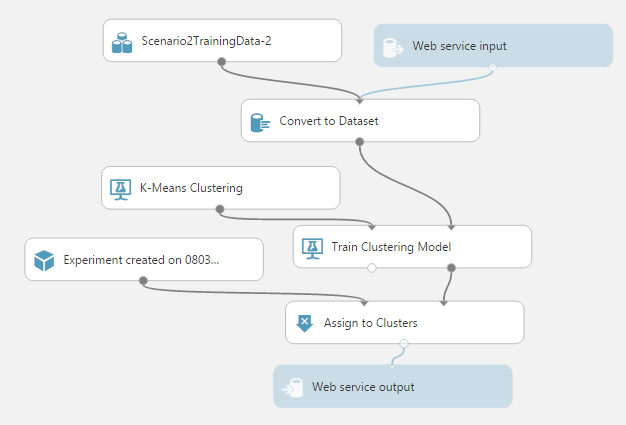
# Using Device Explorer

You can’t configure your IoT Hub devices from the Preview Portal however as part of the Azure IoT SDK there is an application called DeviceExplorer which you can use to manage devices, view messages being sent to your hub and send cloud to device messages. For this part of the lab you will need to have the Azure IoT SDK. For this section please use the first IoT Hub you created for the solution.

1. **Download** or clone the GitHub repo for the **Azure IoT SDK** (<https://github.com/Azure/azure-iot-sdks>). If downloading extract, the zip files to your machine.
2. Navigate to \tools\DeviceExplorer\ in your local repo and **open the DeviceExplorer solution** file in Visual Studio.
3. Run the project.
4. To use Device Explorer, the first thing you need to do is enter the connection string for your IoT Hub. In the Preview Portal click your IoT Hub tile that was created for your earlier.
5. Two panes should now appear for your IoT Hub. The right-hand side pane should be titled **“settings”**
6. Select the **“Shared access policies”** menu option and select the **“iothubowner”** policy in the shared access policies tile.
7. From the **“iothubowner”** tile copy the **“Connection string – primary key”.**
8. Back in Device Explorer paste into the **“IoT Hub connection string”** textbox in Device Explorer.
9. Click the **“Update”** button and a dialog confirming the update of the settings should appear.
10. Select the **management tab** and click the **“Create”** button.
11. Enter a name for your device in the **“device id”** textbox. Copy the primary key value as you will need this in a moment. In this example I’ve called my device **“quiz-server”** and click **“create”**. You should now see the device you created shown in the data grid. You can verify it was created by clicking the **“refresh”** button.

# Configuring Azure ML

In this section you are going to setup an Azure ML experiment. The ML experiment is a basic clustering experiment using k means algorithm.

1. Go to <http://manage.windowsazure.com> and login to your account.
2. Click the bottom **“+”** icon and select **“Data services”** from the menu.
3. Select **“Machine learning”** and then the **“Quick create”** link.
4. In the workspace box enter a name for the workspace. I’m using **“IoTHackathonScenario2”**.
5. Select a location and subscription.
6. Under storage account select **“Create a new storage account”** and enter a name for the storage account.
7. Click the **“Create an ML workspace”** link. A new ML workspace will be created, however this may take a few minutes.
8. When your ML workspace has finished being provisioned select the name of your workspace from the list of ML workspaces under the Machine Learning tab and click the bottom **“Open in studio”** icon.
9. Once in the ML Studio click the bottom **“+”** icon and select **“Dataset”** from the menu. Next select the **“From local file”** option.
10. Click the “choose file” button and select the **“Scenario2TrainingData.csv”** file.
11. In the “Enter a name for the new dataset” textbox enter a name for the dataset. I’ve called mine **“Scenario2TrainingData-2”**.
12. From the **“Select a dataset type…”** drop down select the **“Generic CSV File with a header (.csv)”** option.
13. Finally click the **“tick”** icon. Once the file has uploaded click the **“+”** icon again.
14. This time from the menu select **“Experiment”** and choose the **“Blank experiment”** template. This will create a new experiment and load the ML experiment editor.
15. Your experiment will have a default name of something like “Experiment created….”. Click this text and give your experiment a name.
16. From the left hand menu expand the “Saved datasets” node and then expand the “My Datasets” node. The dataset you just created should appear.
17. Drag and drop the dataset onto the canvas. You will notice the background template disappears and your dataset is now shown on the canvas.
18. In the **“search experiment items”** textbox type **“Convert”**. You will see the list of items changes and a few convert options are shown. Drag and drop the “Convert to Dataset” option onto the canvas below the dataset.
19. On the canvas click back on the dataset and a number 1 should appear at the bottom of the item (output). Click and drag from this to the green dot that appeared on the convert to dataset item (input). The two should now show a line between them.  
    
20. Next again using the **“search experiment items”** feature type **“train clustering”** and drag and drop the **“Train clustering model”** item onto the canvas again below the convert to dataset item.
21. This time drag from the output of convert to dataset to the right hand input of the train clustering model item.
22. Click the train clustering model item on the canvas and on the right hand side of the editor you should see properties. Click the **“Launch column selector”** button.
23. Change the begins with option from **“No columns”** to **“All columns”.**
24. In the “Press enter to enter column name” textbox type **“Age” + Enter, “Gender” + Enter, “TimeToAnswerSeconds” + Enter**. Finally click the **“tick”** icon.
25. Next we need to add a **“K-means clustering”** item so search for **“K-means”** and drop the “K-means clustering” item onto the canvas above and to the left of the train clustering model item.
26. Connect output of the k-means clustering item to the other top input on the train clustering model item.
27. Next search for **“Assign to clusters”** and drop an “Assign to clusters” item onto the canvas below the train clustering model item.
28. From each outlet on the train clustering model item connect these to the two inputs of the assign to clusters item.
29. Select the assign to clusters item and from the properties click **“Launch column selector”.**
30. This time “Begin with” wants to be set to “No Columns” and in the “Press enter to enter column name” textbox type **“Age” + Enter, “Gender” + Enter, “TimeToAnswerSeconds” + Enter**. Finally click the **“tick”** icon.
31. Finally add a **“Convert to dataset”** item connecting the input to the output of the assign to clusters item.
32. Click the **“run”** icon at the bottom of the window to test the experiment runs ok. Once the experiment has run successfully all the items on the canvas should have a green tick.
33. Once the experiment has run successfully we need to convert the experiment into a API which we can consume in other apps.
34. Click the train clustering model item on the canvas and click the bottom **“Set up web service”** button and select **“Predictive web service (recommended) option”**. Azure will now create a web service that can call the ML experiment. The experiment should be similar to the below image. 
35. Save the experiment and click the **“Run”** icon.
36. Once the experiment has run you should now see a **“Deploy web service”**. Once completed your will be redirected to API dashboard for your experiment.

# Configuring the Quiz Server

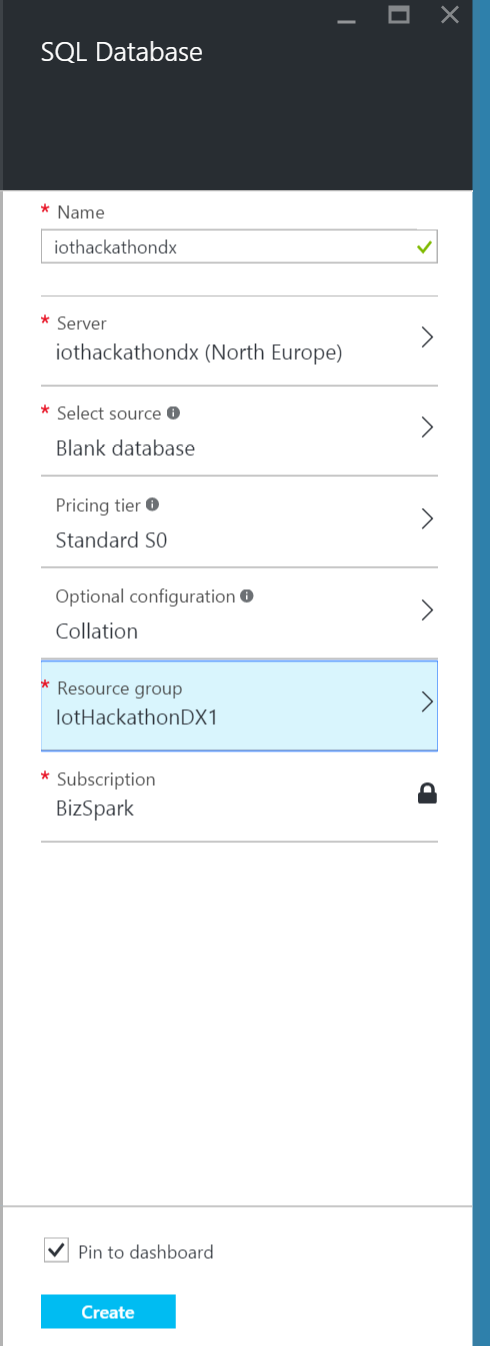
In this section you are going to configure the Quiz Server to run on your Raspberry Pi.

1. Download the solution from **xyz** and open in Visual Studio 2015.
2. Open the class “**HTPServer.cs**” and find the line “private const string IOTHUBCONNECTIONSTRINGANSWERS = …”
3. Replace the “**<IoTHubName>”** text with the **name of the first IoT Hub** you created for this solution.
4. Replace the **“<SharedAccessKey>”** text with primary key you copied in device explorer.
5. Repeat the “Using Device Explorer” instructions but this time use the second “profiles” IoT Hub, using the device name “quiz-server”. Again copy the primary key.
6. Find the line “private const string IOTHUBCONNECTIONSTRINGPLAYERPROFILE = …” and replace the **“<IoTHubName>”** text with the value of the second IoT Hub you created for this solution.
7. Replace the **“<SharedAccessKey>”** text with the primary key you just copied from device explorer.
8. Save the changes to the solution.

# Provisioning Storage

1. On the Preview Portal, click “+ NEW” (top left) and select **Data + Storage**.
2. Select **Storage Account** from the featured apps shown and click the **“Create”** button.
3. Enter a name for your storage account. In this example I am using “iothackathondx2”. The name must in lower case letters.
4. Select the type of storage by clicking on **“Type”**, **locally redundant** is fine for this solution so select this option followed by the **“Select”** button.
5. Chose the resource group you created for your IoT Hubs earlier by clicking the “Select existing” link, then click “Not configured”. A new pane will show with your existing resource groups. Select the correct resource group.
6. For the location select a location local to you where possible.
7. Finally click the create button. You will now be taken back to your Azure dashboard where you will see a new dashboard tile appear showing the status of the provision. Storage accounts can take a few minutes to provision.

# Provision an SQL Database

1. On the Preview Portal, click **“+ NEW”** (top left) and select **DATA AND STORAGE.**
2. Select **SQL Database** from the list of featured apps.
3. Enter the name of your database in the settings tile. In this example I am using IoTHackathonDX2.
4. If you have followed the other scenario’s you can use the same server, you created earlier and skip steps 5 - 7. If not click **“configure server”** followed by **“create new server”**.
5. Enter the server name. In this example I am using IoTHackathonDX. I would suggest calling the server the same as the database to make it easy to remember. Enter a username and password making a note for these for later.
6. Select a suitable region for your server.
7. Click **“Ok”** to confirm the server settings.
8. Select a suitable pricing tier. By default, the pricing tier will be S0 – Standard but you can use the **“B – Basic”.**
9. Select a resource group for the database. I would recommend the same resource group you created for the IoT Hub earlier.
10. Click **“create”**. The SQL Database will now be provisioned. As with the IoT Hub you can see the status of the provisioning process on your dashboard.

## Configure the SQL Database

Next you need to create the table that the data from your IoT Hub will be saved to using Stream Analytics. You will use Visual Studio for this part of the exercise.

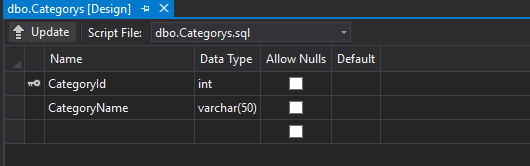
1. Open **Visual Studio 2015**.
2. If you don’t already have **Cloud Explorer** installed download and install it from <https://visualstudiogallery.msdn.microsoft.com/84e83a7c-9606-4f9f-83dd-0f6182f13add>
3. Add your Azure account to cloud explorer by clicking the **“Connect to Microsoft Azure”** link.
4. Enter your credentials in the Visual Studio login popup box.
5. You should now see a list of Azure components. Expand the SQL Databases node and right click on the database you created in the last step. Click the **“Open in SQL Object Explorer”** option and you will be prompted for the password for your database.
6. In the SQL Object Explorer navigate to the table node for your database and right click, selecting **“Add New Table…”**.
7. Enter the following SQL statement into the bottom T-SQL window: -  
   CREATE TABLE [dbo].[Categorys] (

[CategoryId] INT IDENTITY (1, 1) NOT NULL,

[CategoryName] VARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([CategoryId] ASC)

);

1. Click the **“update”** button in the top left hand of the designer window.  
   
2. Repeat steps 7 and 8 for the following table definitions: -

CREATE TABLE [dbo].[Questions] (

[QuestionId] INT IDENTITY (1, 1) NOT NULL,

[QuestionText] VARCHAR (255) NOT NULL,

[CategoryId] INT NOT NULL,

[QuestionDifficulty] INT NOT NULL,

PRIMARY KEY CLUSTERED ([QuestionId] ASC)

);

CREATE TABLE [dbo].[Answers] (

[AnswerId] INT IDENTITY (1, 1) NOT NULL,

[AnswerText] VARCHAR (255) NOT NULL,

[CorrectAnswer] BIT DEFAULT ((0)) NOT NULL,

[QuestionID] INT NOT NULL,

PRIMARY KEY CLUSTERED ([AnswerId] ASC)

);

CREATE TABLE [dbo].[PlayerAnswers] (

[guid] UNIQUEIDENTIFIER NULL,

[category] VARCHAR (50) NULL,

[questionnum] INT NULL,

[answernum] INT NULL,

[timesasked] DATETIME2 (7) NULL,

[timeanswered] DATETIME2 (7) NULL

);

1. Next you are going to create a couple of stored procedures. In SQL Object Explorer expand the **“Programmability”** node and then right click the **“Stored Procedures”** node, selecting **“Add New Stored Procedure”** from the popup menu.
2. Paste the following SQL into the editor window: -  
   CREATE PROCEDURE [dbo].[spGetAnswers]

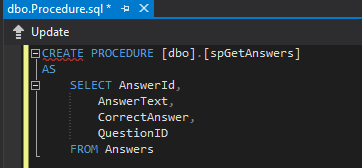
AS

SELECT AnswerId,

AnswerText,

CorrectAnswer,

QuestionID

FROM Answers  
  


1. Click the **“Update”** button in the top left hand of the window.
2. Repeat steps 10 – 12 with the following two stored procedures: -  
   CREATE PROCEDURE [dbo].[spGetCategorys]

AS

SELECT CategoryId,

CategoryName

FROM Categorys  
   
 CREATE PROCEDURE [dbo].[spGetQuestions]

AS

SELECT QuestionId,

QuestionText,

CategoryId,

QuestionDifficulty

FROM Questions

1. The database has now been configured so it’s time to populate the tables with some basic data. On the database node right click and select **“New Query”** from the popup menu. Copy and paste the following SQL into the query window: -  
   SET IDENTITY\_INSERT [dbo].[Categorys] ON

INSERT INTO [dbo].[Categorys] ([CategoryId], [CategoryName]) VALUES (1, N'General Knowledge')

INSERT INTO [dbo].[Categorys] ([CategoryId], [CategoryName]) VALUES (2, N'Geography')

INSERT INTO [dbo].[Categorys] ([CategoryId], [CategoryName]) VALUES (3, N'Film and TV')

INSERT INTO [dbo].[Categorys] ([CategoryId], [CategoryName]) VALUES (4, N'History')

SET IDENTITY\_INSERT [dbo].[Categorys] OFF

SET IDENTITY\_INSERT [dbo].[Categorys] ON

INSERT INTO [dbo].[Categorys] ([CategoryId], [CategoryName]) VALUES (1, N'General Knowledge')

INSERT INTO [dbo].[Categorys] ([CategoryId], [CategoryName]) VALUES (2, N'Geography')

INSERT INTO [dbo].[Categorys] ([CategoryId], [CategoryName]) VALUES (3, N'Film and TV')

INSERT INTO [dbo].[Categorys] ([CategoryId], [CategoryName]) VALUES (4, N'History')

SET IDENTITY\_INSERT [dbo].[Categorys] OFF

SET IDENTITY\_INSERT [dbo].[Answers] ON

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (1, N'Bois de Boulogne', 0, 1)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (2, N'Champ de Mars', 1, 1)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (3, N'Jardin des Plantes', 0, 1)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (4, N'Parc de Belleville', 0, 1)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (5, N'Apollo 7', 0, 2)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (6, N'Apollo 9', 0, 2)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (7, N'Apollo 11', 1, 2)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (8, N'Apollo 13', 0, 2)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (9, N'Charlton Heston', 1, 3)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (10, N'Clark Gable', 0, 3)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (11, N'Errol Flynn', 0, 3)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (12, N'Lee Marvin', 0, 3)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (13, N'HRW', 0, 4)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (14, N'HTR', 0, 4)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (15, N'LHR', 1, 4)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (16, N'LHW', 0, 4)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (17, N'Ukraine', 1, 5)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (18, N'Slovakia', 0, 5)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (19, N'Hungary', 0, 5)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (20, N'Russia', 0, 5)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (21, N'Mount Etna', 0, 6)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (22, N'Mount Stromboli', 0, 6)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (23, N'Mount Vesuvius', 1, 6)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (24, N'Mount Vulture', 0, 6)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (25, N'Angelina Jolie', 1, 7)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (26, N'Minnie Driver', 0, 7)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (27, N'Nell McAndrew', 0, 7)

INSERT INTO [dbo].[Answers] ([AnswerId], [AnswerText], [CorrectAnswer], [QuestionID]) VALUES (28, N'Jennifer Aniston', 0, 7)

SET IDENTITY\_INSERT [dbo].[Answers] OFF

1. Click the **“Update”** button in the top left hand of the window.

# ~~Provisioning an API App~~

~~Your next task is to provision an API App. The app will provide an API that the quiz server uses to retrieve a list of questions and their answers from the database you just provisioned.~~

1. ~~On the Preview Portal (remember you need to be in the Preview Portal), click~~ **~~“+ NEW”~~** ~~(top left) and select~~ **~~Web + Mobile.~~**
2. ~~Select API App from the featured apps shown.~~
3. ~~Enter a name for your API app. In this example I am using IoTHackathon2.~~
4. ~~Select the resource group you have used for the other services you already provisioned.~~
5. ~~Click the~~ **~~“App Service plan/ Location”~~** ~~area and then click the~~ **~~“Create New”~~** ~~button on the panel that appeared.~~
6. ~~Enter a name for your app service plan in the app service plan textbox and then select the region you want to deploy the service to.~~
7. ~~Click~~ **~~“Pricing tier”~~** ~~and change the pricing tier to~~ **~~“B1 Basic”~~** ~~and click the~~ **~~“Select”~~** ~~button.~~
8. ~~Next click the~~ **~~“Ok”~~** ~~button. Finally click the~~ **~~“Create”~~** ~~button. The API app will now be provisioned. As with the other services you can see the status of the provisioning process on your dashboard.~~

# Configuring Web API App

1. Open the **APIApp** solution in Visual Studio.
2. Open the **Web.config** file for the API App project.
3. Under the connectionStrings configuration block you will see a connection string named **“sqldatabase”**.
4. Replace **“<servername>”** with the name of the SQL server you used for your database earlier in this scenario.
5. Replace **“<databasename>”** with the name of the database you created for this scenario.
6. Replace **<userid>** and **<password>** with the login details for your SQL server.
7. Close the web.config file, saving changes.
8. Right click the “APIAPP” project node in the solution explorer and select **“Publish”** from the popup menu.
9. Click on the **“Microsoft Azure API Apps (Preview)”** option under **“Select a publish target”.**
10. Click the **“New”** button.
11. Enter a name for the API App in the **“API App Name”** textbox, making a note of what you called it.
12. Select an existing App service plan or select **“Create new App Service plan”** from the drop down. Enter a name for the App service plan in the textbox.
13. Select the resource group you have used for the other services in this scenario.
14. Make sure Access Level is set to **“Available to Anyone”**.
15. Select a suitable region for the API App.
16. Click the **“Ok”** button to provision the service.
17. Visual Studio will now begin to provision the API App service. This will take a few moments.
18. Once the provisioning has been completed right click on the **“APIAPP”** project node and select **“Publish”** from the popup menu again.
19. Click the **“Publish”** button on the Publish Web dialog. Your API app will now be published. Once published you can close the solution.

# Configuring the Quiz Server part 2

We now need to update the Quiz Server with the URL of the API App you just created.

1. Open the class “**HTPServer.cs**” and find the line “private const string QUESTIONSAPIURL = ". Replace **“<APIAPPName>”** with the name of your API App.
2. Next you need to deploy the solution to your Raspberry Pi. In Visual Studio from the tool bar set the solution platform to **“ARM”** and the target as **“Remote Machine”**.



1. Open the project properties by double clicking the **“Properties”** node in solution explorer and navigate to the **“debug”** settings.
2. In the **“Remote machine:”** textbox you need to enter the IP Address/ Name of your Raspberry Pi. To do this click the “Find” button, wait a few moments for Visual Studio to search for your devices and your Rasbperry Pi should show in the “Auto Detected” section. Click your Raspberry Pi and click the **“Select”** button. If your device doesn’t show your device should be called **“minwinpc”** so enter this value.
3. Press the **“green play”** button next “Remote Machine” or press **F5** to build and deploy the solution to your Raspberry Pi.
4. Finally, once the solution has been deployed open a browser and navigate to **“http://<RaspberryPi IP address or name>/”** to play the quiz.

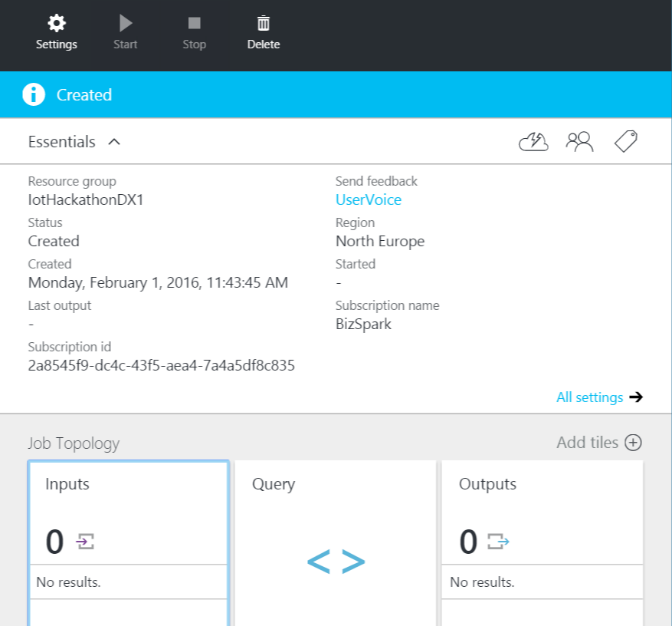
# Provisioning Stream Analytics

You are now going to provision and configure the Azure Stream Analytics service. This will be used to process your data from your IoT Hub and save it into the SQL Database.

1. On the Preview Portal, click **“+ NEW”** (top left) and select **INTERNET OF THINGS.**
2. Select **Stream Analytics Job** from the list of featured apps.
3. Enter a suitable name for the job.
4. Make sure a subscription is selected.
5. Select a resource group for the Job. I would recommend the same resource group you created for the IoT Hub earlier.
6. Select a suitable region for the Job.
7. Click the **“Create”** button. The Stream Analytics Job will now be provisioned. As with the IoT Hub you can see the status of the provisioning process on your dashboard.

## Configuring the Stream Analytics Jobs - Inputs and outputs

Once the job has provisioned the Preview Portal should automatically open up the job. For this Scenario there are two inputs and two outputs. I would recommend opening a second tab in your browser with the Azure portal in both tabs.

1. Under **“Job Topology”** click the “Inputs” box. You will then see a new tile open.
2. On the “Inputs” tile click the top left **“Add”** link.
3. In the **“Input Alias”** textbox enter“answers”.
4. Select **“Data stream”** as the Source Type.
5. Select **“IoT Hub”** as the Source.
6. Next you need the information for the first IoT Hub you created for this scenario. In the second browser tab from your **dashboard** select the IoT Hub you created earlier.
   1. Make a note of the IoT Hub name.
   2. From the settings tile select **“Shared access policies”.**
   3. Select the device policy.
   4. Copy the **“primary key”** for the policy.
7. Back in the previous browser tab enter the name of your IoT Hub in the IoT Hub textbox.
8. In the **“shared access policy name”** textbox enter **“service”**.
9. In the **“shared access policy key”** textbox paste the primary key you copied from the IoT Hub a few moments ago.
10. Leave the consumer group blank.
11. Make sure the Event serialization format is set to JSON and the Encoding is set to UFT-8.
12. Click the **“Create”** button. You will now see under the list on inputs the new input you just created.
13. Repeat steps 1 – 12 but this time using an input alias of **“playerprofile”** and using the **second IoT Hub** you created for this scenario.
14. Back under **“Job Topology”** click the “Outputs” box. You will then see a new tile open.
15. On the “Outputs” tile click the top left **“Add”** link.
16. In the **output alias** textbox enter **“blobstorage”**.
17. Select “Blob storage” from the Sink dropdown.
18. In your second browser tab select the storage account you created earlier on your **dashboard**.
19. Click the **“Access keys”** menu option on the settings pane.
20. Copy the Storage account name and paste this into the Storage account name for the Stream Analytics Output configuration on your first browser window.
21. Copy the KEY1 value from the Access Keys settings and paste this into the **“Storage account key”** textbox for your output configuration.
22. In the Container name textbox enter **“streamoutput”.**
23. In the Path pattern textbox enter it in the following format omitting any spaces “**<yourname>/{date}/{time}”**, replacing <yourname> with your name.
24. The date format should be set to **“YYYY/MM/DD”**.
25. The time format should be set to “**HH”**.
26. Change the Event serialization format to **“CSV”**.
27. Delimiter should be set to comma.
28. The encoding should be set to UTF-8.
29. Finally click the **“Save”** button.

If you have a paid for PowerBi online account, you may follow the next steps, else skip to the Configuring the Stream Analytics Jobs – Queries section.

1. For this task you will need to use the original Azure management portal. Go to <https://manage.windowsazure.com> and navigate to your Stream Analytics job.
2. Click the **“Outputs”** tab and click the bottom **“Add Output”** link.
3. In the **output alias** textbox enter **“powerbi”**.
4. Select **“PowerBi”** from the list of output sources and click the next icon.
5. Now click the **“Authorize Now”** link and sign into your PowerBi account when requested to do so.
6. Once authorisation has been completed you will be asked to enter the details for the output. Enter **“powerbi”** as the Output Alias.
7. Enter **“qanda”** in the dataset name textbox.
8. Enter **“class”** in the table name textbox.
9. Finally click the tick icon to complete the output setup. Switch back to the new Azure portal for the rest of this Scenario.

## Configuring the Stream Analytics Jobs – Queries

1. Back under **“Job Topology”** click the “Query” box. You will then see a new tile open.
2. Enter the following query in the query editor window: -  
   SELECT guid,   
   category,   
   questionnum,   
   answernum,   
   timeasked,   
   timeanswered   
   INTO blobstorage FROM answers TIMESTAMP BY timeanswered
3. If you configured the PowerBi output also enter the following query below the first one, else skip to step 4: -  
   SELECT

playerprofile.age,

MAX(DATEDIFF(second,answers.timeasked, answers.timeanswered)) as maxanswertime,

MIN(DATEDIFF(second,answers.timeasked, answers.timeanswered)) as minanswertime,

AVG(DATEDIFF(second,answers.timeasked, answers.timeanswered)) as avganswertime

INTO powerbi

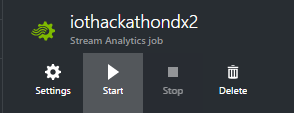
FROM answers TIMESTAMP BY timeanswered

JOIN playerprofile TIMESTAMP BY timecreated

ON DATEDIFF(minute,answers,playerprofile) BETWEEN -15 AND 1

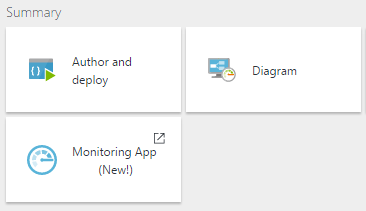
AND answers.guid = playerprofile.guid

GROUP BY playerprofile.age, TumblingWindow(minute, 3)

1. Click the “**save”** button and close the query editor window by clicking on the cross in the top right hand corner.
2. Finally, you need to start your job. Click the start button.

# Provisioning a Data Factory

Your next step in this scenario is to setup an Azure Data Factory to move the data from the blob storage to your SQL Database. The blob storage is populated by the Stream Analytics job you just configured, the Data Factory then transfers this data into the playeranswers table in your database. You could achieve the same result by having an SQL Database output on the Stream Analytics but this is just simply a demonstrate of how the Data Factory service works.

1. On the Preview Portal, click **“+ NEW”** (top left) and select **Data + Analytics.**
2. Select **Data Factory** from the list of featured apps.
3. Enter a suitable name for the Data Factory.
4. Make sure a subscription is selected.
5. Select a resource group for the Data Factory. I would recommend the same resource group you created for this scenario earlier.
6. Select a suitable region for the Data Factory.
7. Click the **“Create”** button.
8. Once the Data Factory has been provisioned we need to setup the configuration of the factory. This is done via JSON files. From your dashboard select the Data Factory you just created.
9. Click the “Author and Deploy” panel in the Summary tile.  
   
10. First we need to create two datasets. Click on the **“New dataset”** icon and from the drop down menu select **“Azure Blob storage”**.
11. If the editor window doesn’t open automatically under the Drafts node of the left hand tree you should see a node called “Draft-1”. Select this node and the editor will appear.
12. In the editor window paste the following text. Make sure you replace **“<yourname>”** with the name you used for the Stream Analytics Blob output.   
    {

"name": "AzureBlobInput",

"properties": {

"published": false,

"type": "AzureBlob",

"linkedServiceName": "AzureStorageLinkedService",

"typeProperties": {

"folderPath": "streamoutput/**<yourname>**/{Year}/{Month}/{Day}/{Hour}",

"format": {

"type": "TextFormat",

"rowDelimiter": "\n",

"columnDelimiter": ","

},

"partitionedBy": [

{

"name": "Year",

"value": {

"type": "DateTime",

"date": "SliceStart",

"format": "yyyy"

}

},

{

"name": "Month",

"value": {

"type": "DateTime",

"date": "SliceStart",

"format": "MM"

}

},

{

"name": "Day",

"value": {

"type": "DateTime",

"date": "SliceStart",

"format": "dd"

}

},

{

"name": "Hour",

"value": {

"type": "DateTime",

"date": "SliceStart",

"format": "hh"

}

}

]

},

"availability": {

"frequency": "Hour",

"interval": 1

},

"external": true,

"policy": {

"externalData": {

"retryInterval": "00:01:00",

"retryTimeout": "00:10:00",

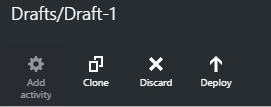
"maximumRetry": 3

}

}

}

}

1. Next click the “Deploy” button at the top of the editor window to save the dataset.  
   
2. Click on the **“New dataset”** icon again and from the drop down menu select **“Azure SQL”**.
3. In the editor window for this dataset paste the following text: -  
   {

"name": "AzureSqlOutput",

"properties": {

"structure": [

{

"name": "guid",

"type": "Guid"

},

{

"name": "category",

"type": "String"

},

{

"name": "questionnum",

"type": "Int32"

},

{

"name": "answernum",

"type": "Int32"

},

{

"name": "timesasked",

"type": "Datetime"

},

{

"name": "timeanswered",

"type": "Datetime"

}

],

"published": false,

"type": "AzureSqlTable",

"linkedServiceName": "AzureSqlLinkedService",

"typeProperties": {

"tableName": "PlayerAnswers"

},

"availability": {

"frequency": "Hour",

"interval": 1

}

}

}

1. Click the **“Deploy”** button to save and deploy this dataset.
2. Now we have our input (Blob storage) and output (SQL Database) datasets configured we need to configure the Pipeline script. Click the “More commands” icon and then click **“New Pipeline”.**

Paste the following text into the editor window. Change the Start and End times to be from when this job will start until start + 2hours. Also change **“<hubName>”** to something that makes sense to you: -  
{

"name": "PipelineTask",

"properties": {

"description": "pipeline with copy activity",

"activities": [

{

"type": "Copy",

"typeProperties": {

"source": {

"type": "BlobSource",

"skipHeaderLineCount": 1

},

"sink": {

"type": "SqlSink",

"writeBatchSize": 0,

"writeBatchTimeout": "00:00:00"

}

},

"inputs": [

{

"name": "AzureBlobInput"

}

],

"outputs": [

{

"name": "AzureSqlOutput"

}

],

"policy": {

"timeout": "01:00:00",

"concurrency": 1

},

"scheduler": {

"frequency": "Hour",

"interval": 1

},

"name": "AzureBlobtoSQL",

"description": "Copy Activity"

}

],

"start": "2016-02-27T11:00:00Z",

"end": "2016-02-28T11:00:00Z",

"isPaused": true,

"hubName": "<hubname>",

"pipelineMode": "Scheduled"

}

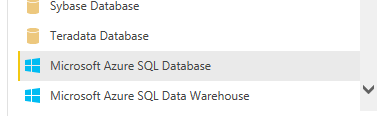
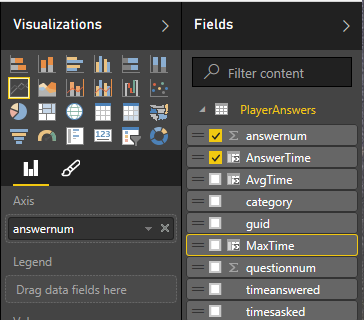
}

1. Finally click the **“Deploy”** button.

Your Azure Data Factory is now configured. It will run every hour moving the data from Blob storage to your SQL Database. You may not see any results in the database until after the first hour has elapsed.

# Configuring PowerBI

In this section you are going to use PowerBi to visualise the data.

1. If you don’t already have PowerBi desktop edition installed on your machine download and install it from <https://powerbi.microsoft.com/en-us/desktop/>.
2. Open PowerBi and click on the **“Get Data”** link on the start-up screen.
3. From the list select **“Microsoft Azure SQL Database”** and click the **“connect”** button.  
   
4. Enter the database name you created earlier in this exercise along with the server name. The server name should be in the format of **“<servername>.database.windows.net”** and click **“Next”**.
5. Select **“Database”** from the side menu and enter the username and password for your database.
6. In the data navigator from the left hand side use the tree to navigate to the **“PlayerAnswers”** table and tick. Finally click the **“Load”** button.
7. From the tool bar click the “**New Column**” button and in the formula field enter **“AvgTime = AVERAGE(PlayerAnswers[AnswerTime])”** followed by the Enter key.
8. Click the **“New Column”** button again and in the formula field enter **“MaxTime = MAX(PlayerAnswers[AnswerTime])”** followed by the Enter key.
9. Finally add a new column with the following formula **“MinTime = Min(PlayerAnswers[AnswerTime])”**.
10. On the right hand side of PowerBi you should see a panel as per the below screen shot: -  
    
11. Next click the **“Card”** icon (4th Icon bottom row) under the Visualization section.
12. From the fields select **“MinTime”**.
13. Repeat steps 11 and 12 selecting **“MaxTime”** and **“AvgTime”**.