**Extending**

**SpotfireDemo App**

**Documentation**

**(Windows Forms)**

**(Cloud Databases)**

**(TIBCO Spotfire Visuals)**

**Table Of Contents**

1. **Introduction……………………………….………….. 2**
   1. **Important Notes………………………………………...…… 2**
2. **Prerequisites…………………………………..….…. 3 - 16**
   1. **Google BigQuery……………………………..………....…. 3 - 4**
   2. **Amazon RDS (Microsoft SQL Server Edition)……........ 5 - 16**
      1. **Setup………………………………………….…....... 5 - 9**
         1. **Creating AWS Database Server……....... 7**
         2. **Creating AWS Database……………........ 8 - 9**
      2. **School WiFi Blockage……………………...…..... 10 - 11**
      3. **Other Potential Bugs………………………..…..... 11 - 16**
         1. **SQL Server Configuration Manager….... 12 - 13**
         2. **Using TCPView…………………………..... 13 - 14**
         3. **Using Windows Defender Firewall…...... 14 - 15**
         4. **Using Services…………………………...... 16**
3. **Google BigQuery Demo……………………..……. 17 - 20**
   1. **Codes……………………………..……………………....…. 17**
   2. **Copy Of Database Table………………………..…...……. 18 - 19**
   3. **How To Use……………………………..…………………... 20**
4. **Google BigQuery Demo……………………..……. 21 - 24**
   1. **Codes……………………………..……………………...…. 22**
   2. **Copy Of Database Table…………………….…..………. 23**
   3. **How To Use……………………………..…………….……. 24**
5. **Combining Both Demos……………………..……. 25**
6. **Improving IoT Demo…..……………………..……. 26 - 28**
7. **References……………………………………..……. 29 - 34**
   1. **General..………………………………………………….…. 29**
   2. **Platforms……………………………….…………..………. 29 - 33**
   3. **Languages……………………………..………………..…. 33**
   4. **Improving IoT Demo………………………………… 33 - 34**

**1. Introduction**

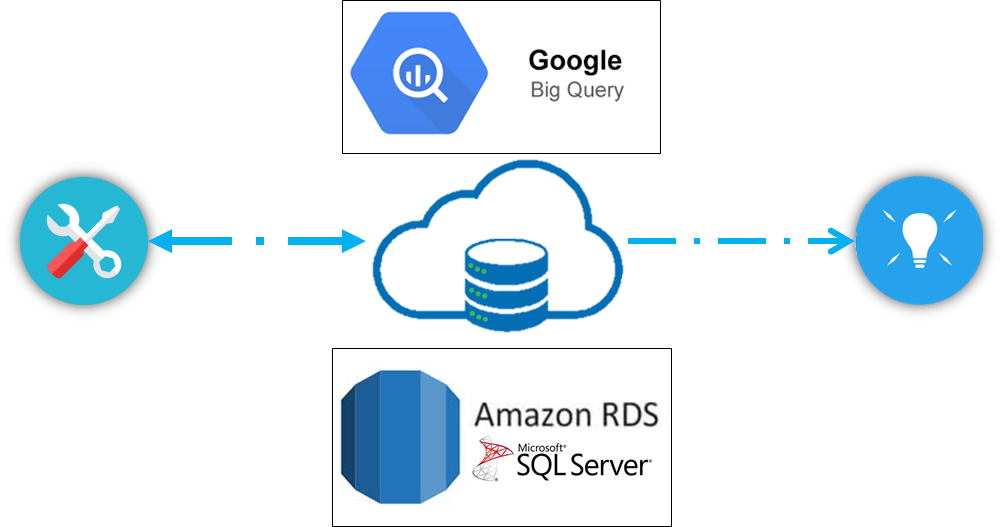
This documentation report is a continuation from the “SpotfireDemo App” Documentation. It covers extending the SpotfireDemo with cloud databases, such as Google BigQuery and Amazon RDS (Microsoft SQL Server Edition).

Before continuing this documentation, please have a read on the “SpotfireDemo App” Documentation to have a basic understanding of how the SpotfireDemo App works.

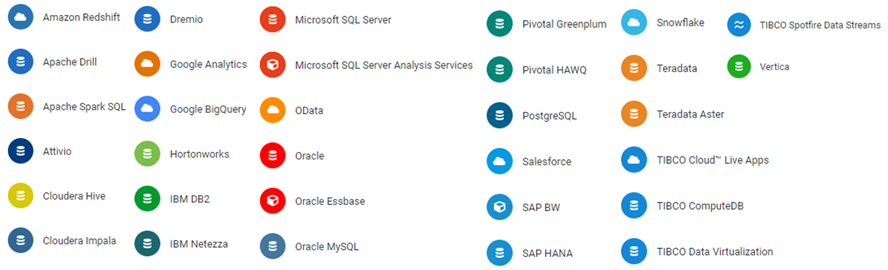
**1.1 Important Notes**

The extension of the SpotfireDemo App with cloud databases will require certain prerequisites before performing the extension.

For this extension, I will be demonstrating how to implement Google BigQuery and Amazon RDS into the SpotfireDemo App, then connecting the either of these cloud databases into Spotfire. Hence, requires a registered Google and Amazon account.



Also, if you plan to use any other cloud database services, please visit TIBCO Spotfire and view the list of available connections eligible with Spotfire cloud database connection. Below is a list of accepted connections as of writing this report.

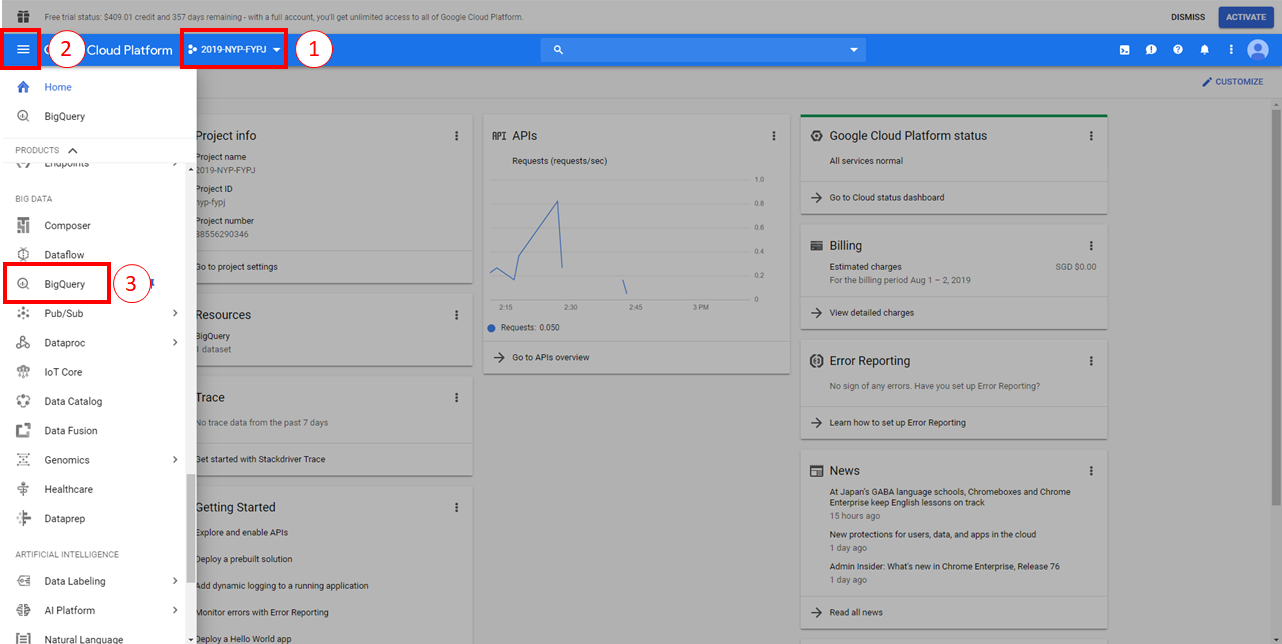


**2. Prerequisites**.

**2.1 Google BigQuery**

As previously mentioned, a Google Account is required to use Google BigQuery. Therefore, please create a Google Account before continuing to attempt any sections of this report regarding Google BigQuery. You can create an account [here](https://cloud.google.com/).

After creating your Google Account, navigate to your Google Cloud Platform Console. Then open up the BigQuery service provided inside the Console. Below I will explain the steps on how to open the BigQuery service, you may skip if you already understand.



**In Step 1:**

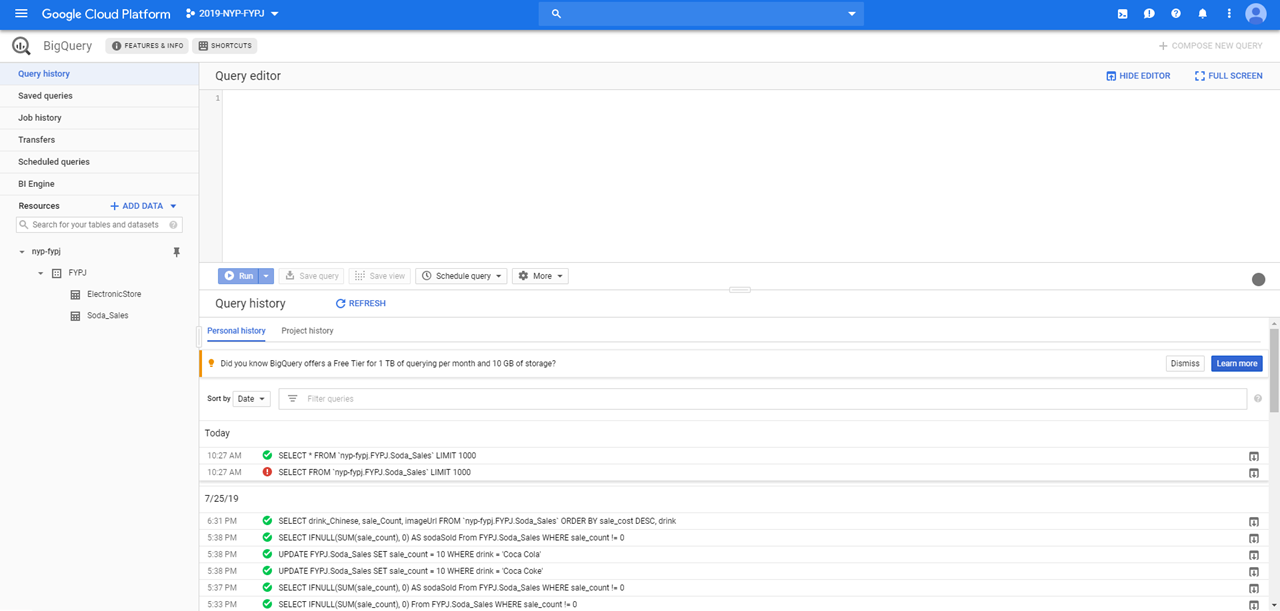
At the very beginning of entering your Console, you must first select a project. Give it any name. Once created, you must associate a billing method (Enable billing for this project). This is essential to gain full access to Google BigQuery.

**In Step 2:**

Once your project is created with billing enabled, open your dashboard and scroll towards the “BIG DATA” section. Locate BigQuery and select it. You should also PIN the tab for future convenience.

**In Step 3:**

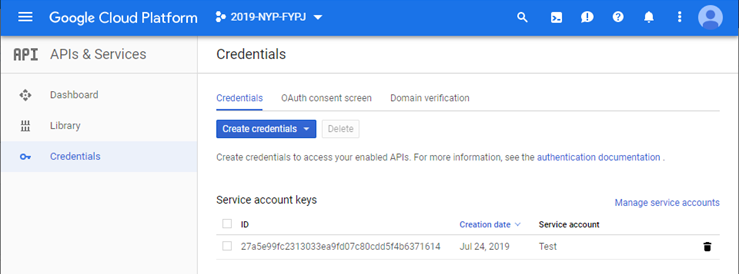
You will be visited with the GUI for Google BigQuery.

****

Create a simple DATASET with a TABLE with the GUI. To retrieve, modify or delete any data, you must use the Query Editor provided. You can also save your queries for future use.

**Finale:**

Once completed, you are ready to use this data table that you have created. However, to use this data you must first create a user credential .json file. The file will be later mentioned again when recreating the project.



Using the navigation menu, go to “APIs & Services > Credentials”. Then click on “Create credentials > Service account key”. Follow the prompts accordingly to create the service account key.

When creating the service account key, download the json file that is associated with the created service account and save it. This json file will act as your Google Credentials to access the BigQuery.

**2.2 Amazon RDS (Microsoft SQL Server Edition)**

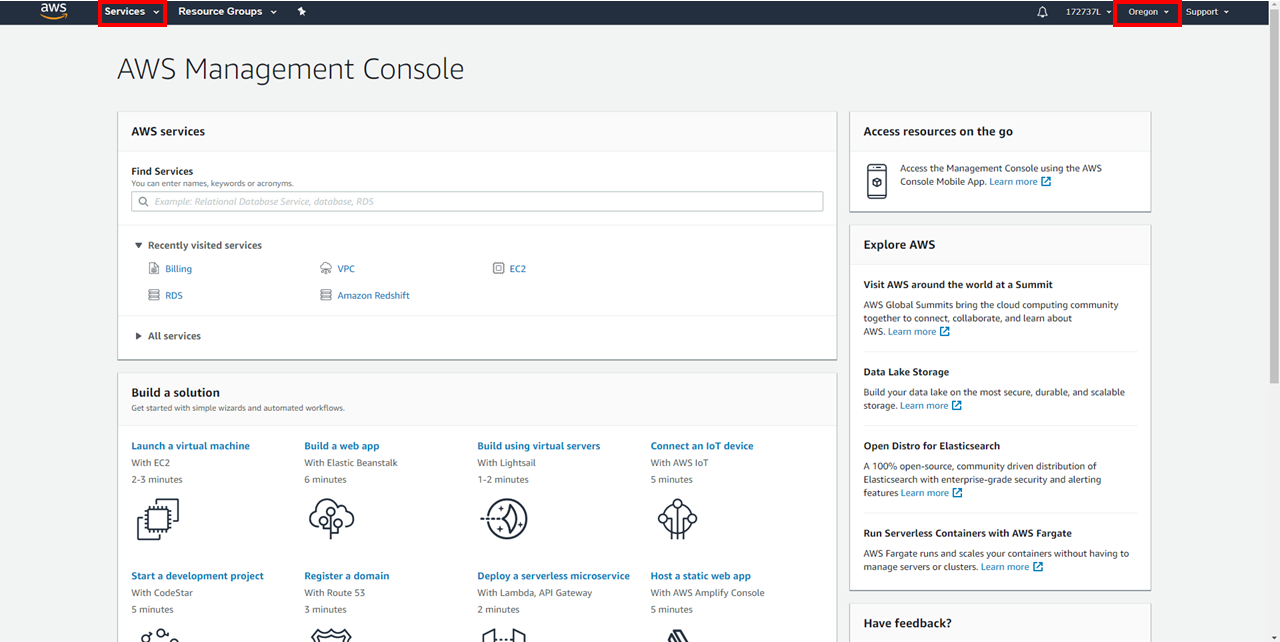
Amazon RSD (AWS RDS) is a lot more complicated in setup due to the many services offered by AWS. It is also very difficult to configure a connection with a database after creation due to potential bugs and the blockage in school WiFi.

Therefore, I will be sharing my exploration findings on how to maneuver over these bugs and an alternate solution to the blockage by school WiFi..

**2.2.1 Setup**

First, I will explain how to setup an AWS RDS - MS SQL database. Enter the AWS website [here](https://aws.amazon.com/) and create an AWS account. It will require you to enter your credit card information.

After you have finished registration, proceed to log into your account and enter the Console for AWS. Inside the console, you may change your region to Singapore or leave it. It has no real effect.



Then click on “Services > Database > RDS”. This will be the main AWS service we will be using as the AWS Cloud Database Server.



**In Section 1:**

You can create a database from either option.

**In Section 2:**

You can create a subnet group to store all subnets into a single group. This subnet group will be used for your database creation.

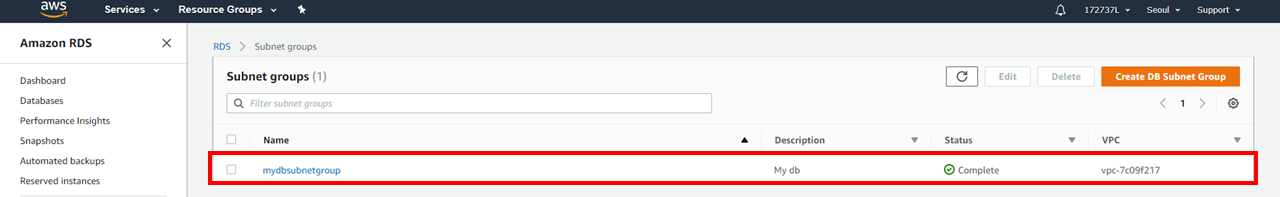
**2.2.1.1 Creating AWS Database Server**

As of writing this report, a new tutorial/method of creating a database is now available, labelled “Easy create”. You may follow the new method by using the official guide [here](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_GettingStarted.CreatingConnecting.SQLServer.html). I will be explaining how to create a Microsoft SQL Database Server with the old method.

Before clicking on “Create Database”, go to the “Subnet groups” tab and create a subnet group. This subnet group will be used for the database server creation. Creating the subnet group, follow the prompts accordingly.

1. Name: Your group name
2. Description: Your group description
3. VPC: Use default
4. Add subnets: Click on “Add all the subnets related to this VPC”

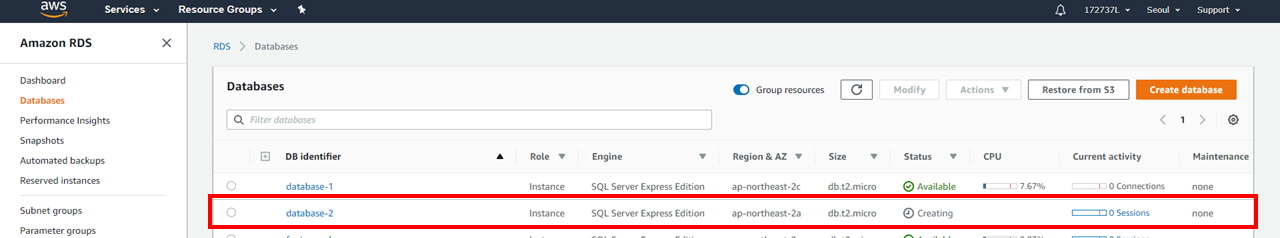
Then proceed to create your subnet. It will be displayed inside your lists of subnet groups.



Now creating a database, uncheck “Easy create” and select the Engine Type “Microsoft SQL Server”. Next select the following options:

1. Edition: SQL Server Express Edition
2. Version: Any version
3. Templates: Free tier
4. DB Instance Identifier: Your DB server name
5. Master Username: Any name
6. Master Password: Any password
7. Under “Additional connectivity configuration”
   1. Subnet group: Use your new subnet group created
   2. Publicly Accessible: Yes
   3. VPC security group: Create new
8. Use the default settings for the rest of the options

Once the following options are filled, proceed to “Create Database”. The database will then be displayed inside the “Databases” tab as follows.



After the database is created, it may not be usable based on the blocked contents by the school WiFi. The school Wifi blocks all of the ports used for an AWS MSSQL Database Server. Hence, I will guide how to go around using the same school WiFi with a VPN to connect your AWS Database Server in the “2.2.2 School WiFi Blockage” section of this report.

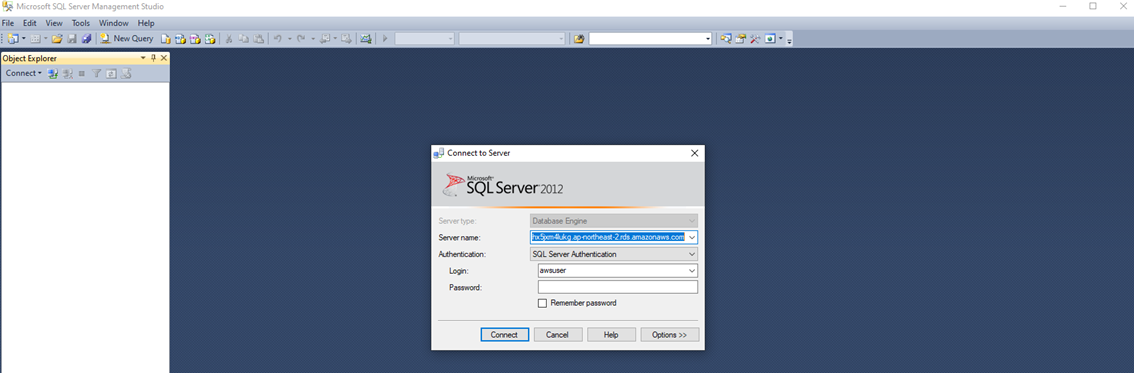
**2.2.1.2 Creating AWS Database**

The next step after creating your AWS Database Server is to create an AWS Database. To create a database, you may follow this video [here](https://www.youtube.com/watch?v=EBTBCBUGR4I&fbclid=IwAR1xZIttetMUBJhBG1zO80DFIe3vjPGQhVArNpXdGrG3D2zvy54_13su1KY) at timestamp 7:40. I will still be sharing on how to create an AWS Database on my part of this report.

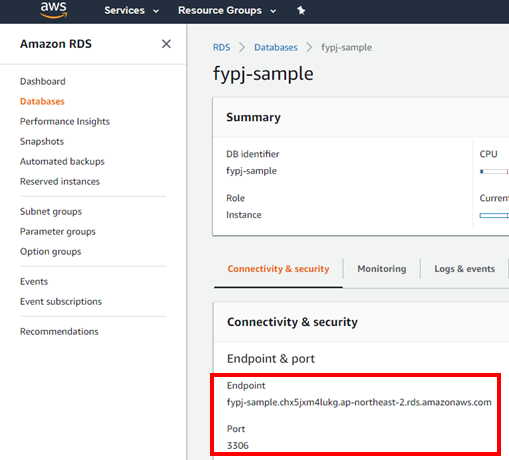
First, ensure that you have SQL Management Studio 2012 Installed. It can be downloaded [here](https://www.microsoft.com/en-sg/download/details.aspx?id=29062). The installation will come with the main SQL Management Studio 2012 and SQL Server Configuration Manager 2012.

Keep both software as Management Studio will be used to create the database, while Server Configuration Manager will be used for later debugging for any potential bugs.

After completing installation, open your Management Studio. On first startup, a popup will appear to prompt for an immediate SQL Server connection.



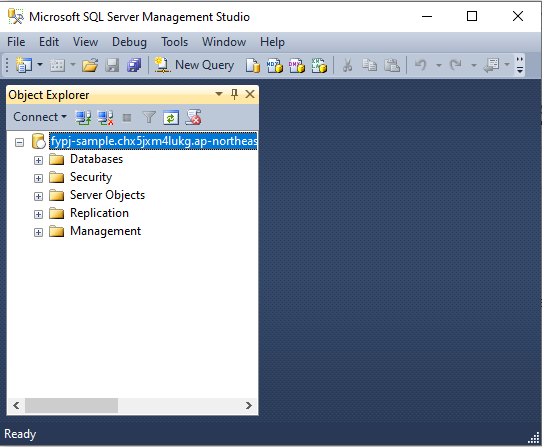
With your AWS RDS console website open, select your database server and copy the endpoint to the SQL Server connection prompt under “Server name”. If your port is other than 1443, the “Server name” must be “Endpoint, Port”.



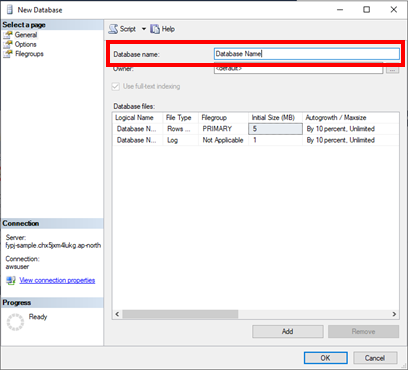
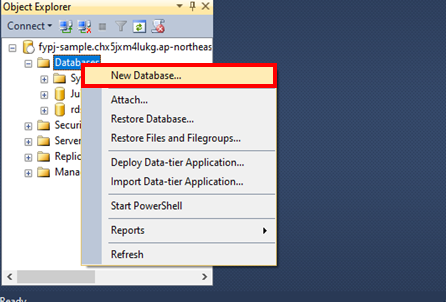
Select SQL authentication and enter your master username and password. Finally, create a connection.

**Note**: A connection may require a VPN using school WiFi.

Once a connection is created, your AWS connection will appear at the left of Management Studio.

****

With the connection established, proceed to create a database. Right click the fold “Databases” and create a new database, then enter a database name and click “OK”. This will wrap up the creation of a database for your future Visual Studio project.



**2.2.2 School WiFi Blockage**

With the school WiFi, it has caused certain ports to be blocked and rendered unusable. These ports are important for our AWS RDS connection. As of writing this report, the following ports 5222, 5223, 5228 and 59234 is able to connect to the AWS database server. However, they may not be available in the future.

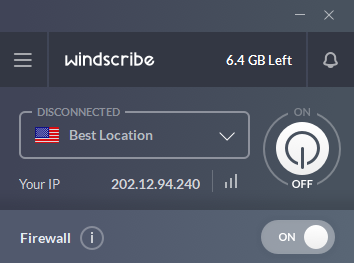
For a backup alternative, a solution was found with VPNs. Hence, I will be sharing how to use a VPN and also recommend a VPN provider to use for this project.

By using a VPN, we can mask the school WiFi to allow the access of the port 1433, the default port used for the AWS RDS database. The desired port used may be different based on how you configured your AWS Cloud Database Server.

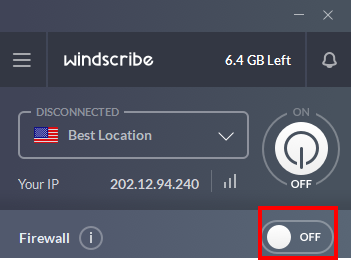
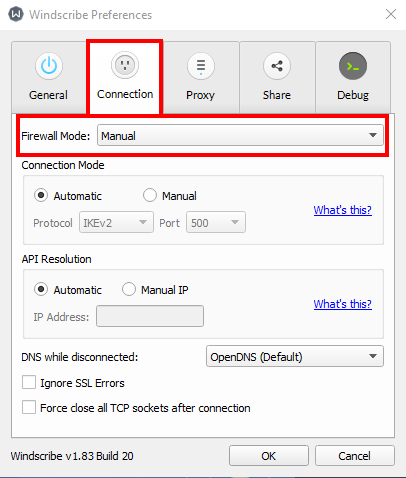
For any VPN services, just enable the VPN then create a connection in Visual Studio’s Server Explorer with your AWS RDS database. Ensure the the AWS RDS database is a MS SQL Database. Now, you are successfully connected to the database.

If you do not have any VPN, I recommend using WindScribe. With WindScribe, you will get free VPN data renewed monthly. It will start with only 4GB of data, but by verifying your Email, it will increase the limit to 10GB. You may also create multiple accounts to use as much data as you want. You can visit their website [here](https://windscribe.com/).

Installing the WindScribe software is required to enable their VPN. The software will appear as a small popup with a simple GUI.



When using the VPN with the school WiFi, go to your “Preferences > Connection” and for “Firewall Mode” select “Manual”, then turn off the firewall. This is essential to unblock the AWS Port that we will be using.



After configuration, just turn on the VPN whenever intending to open a connection to your AWS database server. Hence, I recommend WindScribe as it is user intuitive and easy to use.

**2.2.3 Other Potential Bugs**

In this section, I will be sharing my exploration findings on how to overcome potential bugs and an alternate solution to the blockage by school WiFi..

To fix the bugs, we will require numerous external software and prebuilt troubleshooting programs in Windows PC. We will be using the following lists of software and programs:  
  
Software:

1. SQL Server Configuration Manager
2. TCPView

Default Programs:

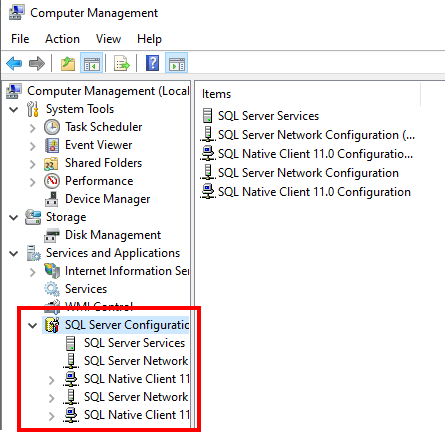
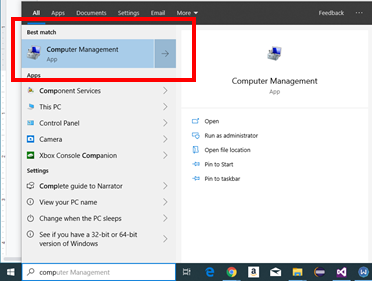
1. Windows Defender Firewall with Advanced Security
2. Services

**2.2.3.1 SQL Server Configuration Manager**

In this section of the Other Potential Bugs, we will be using SQL Server Configuration Manager to change a potential bug in the settings.

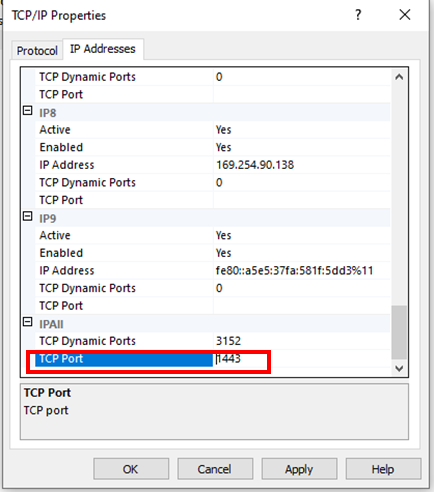
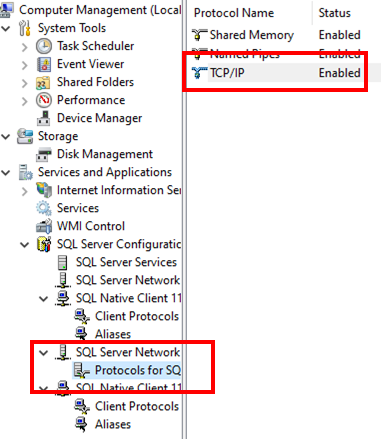
From previous installations, you would have already installed SQL Server Configuration Manager 2012 as it comes together with the SQL Server Management Studio.

To open your SQL Server Configuration Manager, at your windows search bar, type “Computer Management”. Then navigate to “Services and Applications > SQL Server Configuration”.

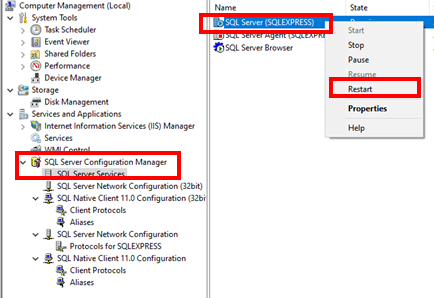


The following items inside “SQL Server Configuration” are the items we will be focusing. For every item labelled “Shared Memory”, “Named Pipes” and “TCP/IP”, change their status to “Enabled”.

Then, for “SQL Server Network Configuration > Protocols for SQLEXPRESS > TCP/IP”, open the properties and change the TCP Port to the same port that is currently used for the AWS connection,



After changing every mentioned setting, navigate to “SQL Server Services > SQL Server (SQLEXPRESS)”. Then right-click and restart the “SQL Server (SQLEXPRESS).



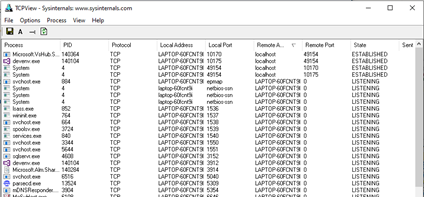
This will reboot with your new settings and hopefully be able to make a connection to your AWS database. If your connection still has problems, refer to the usage of other troubleshooting tools to identify the cause of the error.

**2.2.3.2 Using TCPView**

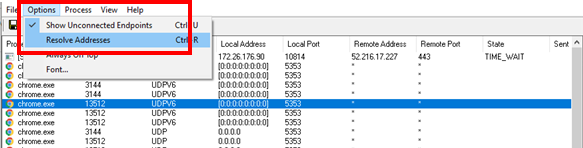
The TCPView tool is used to view all your ports that is being used. This tool is used to mainly help identify which endpoints are being used in both ends of the connection.

It does not directly troubleshoot any errors in connection; however, it helps understand and identify the cause of the error. The download can be found [here](https://docs.microsoft.com/en-us/sysinternals/downloads/tcpview?fbclid=IwAR1w-MGgOCf49Ddlkz-sguGKl_Y0Od_HWrMmduubqMjiTz6Uhq6n7rGIM-U) with documentation.

This is a sample of how the TCPView UI will look like:



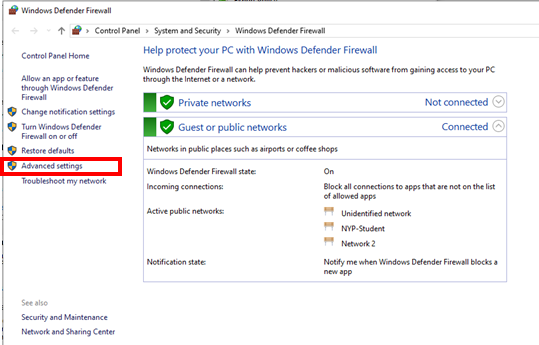
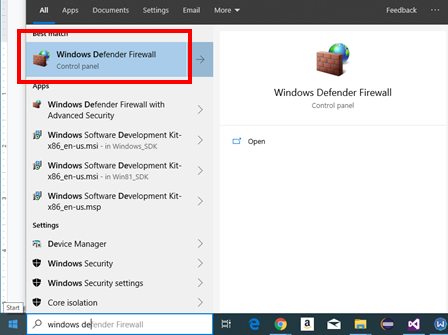
To allow for a readable address in digits, go to “Options > Resolve Addresses” and uncheck it. This will change the IP Address into digits.

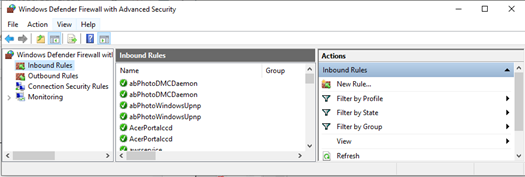


**2.2.3.3 Using Windows Defender Firewall**

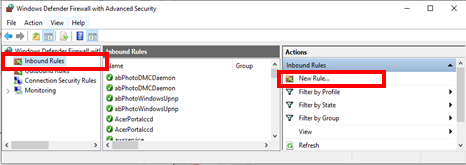
The Windows Defender Firewall may have settings that block the connection to the AWS database. To fix this, I will be guiding how to configure the settings of the firewall.

At your search bar, type “Windows Defender Firewall”, then a popup will appear. In the popup, click on “Advanced settings”. This will open the final popup which is the main page to configure the firewall settings.

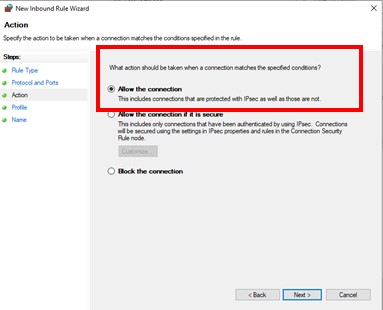
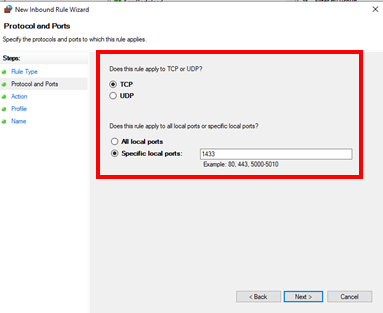
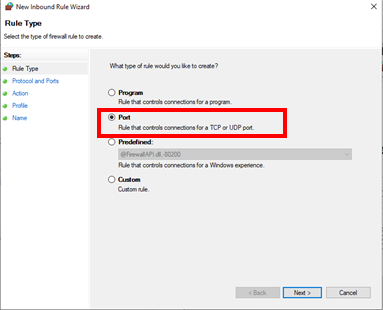




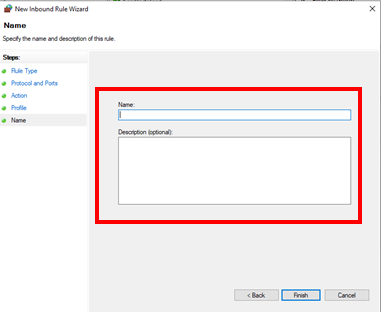
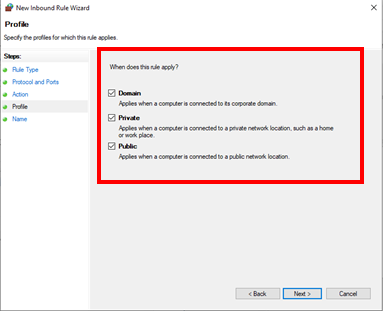
In this page, inside “Inbound Rules”, create a “New Rule”.



This will undergo creating a firewall rule to enable the connection to the AWS servers. In the first section, select “Port”. Then, select “TCP” and enter the desired port to be targeted. Next, select “Allow the connection” to ensure that the connection can be allowed.



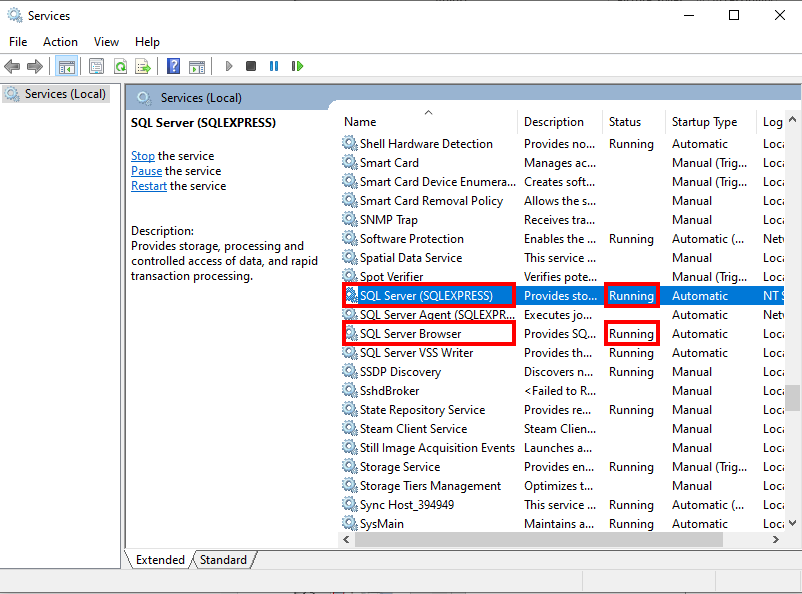
Then, check every checkbox, “Domain”, “Private”, “Public”. And finally, give your new rule a name and a short description.



This will allow your firewall to ignore the port and allow that port to be used to create the AWS connection.

**2.2.3.4 Using Services**

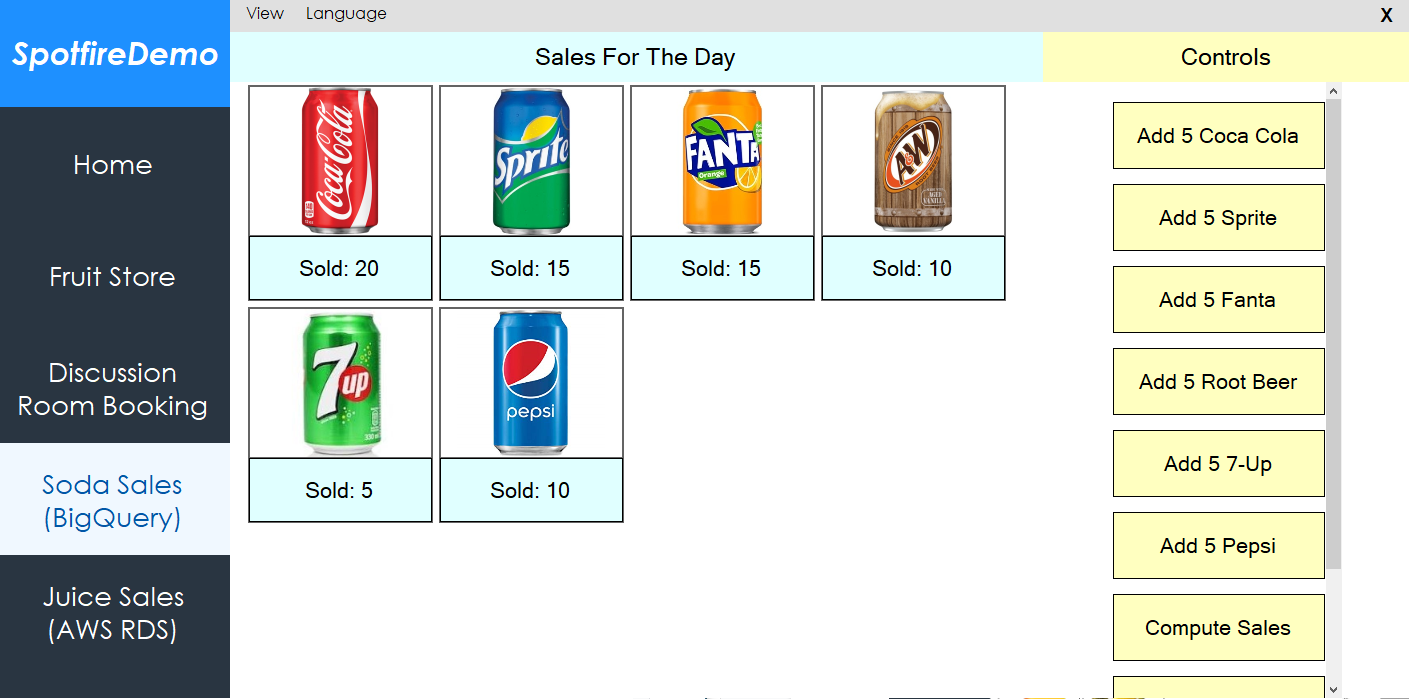
Open your services by the windows search bar. A popup will open. In this popup, search for “SQL Server (SQLEXPRESS)” and “SQL Server Browser”. Ensure that both of these instances are “Running” status.



This is to ensure that your SQL Server connections are enabled to allow for connections to AWS databases.

**3. Google BigQuery Demo**

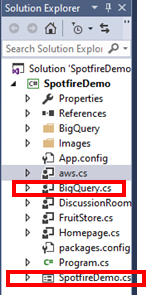
From the original Spotfire Demo App, I have made improvements and added a demo utilizing Google BigQuery as a cloud database source. You can access this demo by running the Spotfire Demo App and selecting the “Soda Sales (BigQuery) to have a glimpse of the demo.



As the demo uses Google BigQuery to store its data, you may only view the data via the Google BigQuery Query Editor.

**3.1 Codes**

By opening the source codes of the project file “SpotfireDemo.cs” and “BigQuery.cs”, you can see the codes to connect to the cloud database and to call/update the cloud database, respectively.

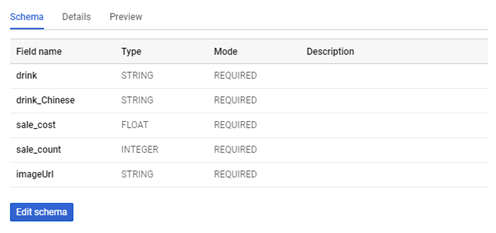


As the demo codes are lengthy and complicated, it is best to explore the source codes yourself. Comments are provided inside the source codes already to further learn how the program works. **Note**: You require to provide your own working account credentials to use the demo.

**3.2 Copy Of Database Table**

As you require to use your own Google BigQuery account, I will provide the exact database used for this table. Copy the database table schema and data onto your own BigQuery Database. You may also use the Saved Queries as well.

**Database Schema:**



**Database Data:**



**Database Saved Queries:**



**Delete Sample:**

DELETE FROM FYPJ.ElectronicStore WHERE stock != 0

**Insert Sample:**

INSERT FYPJ.Soda\_Sales (drink, drink\_Chinese, sale\_cost, sale\_count, imageUrl)

VALUES('Coca Coke', '可口可乐', 1.1, 0, 'coke.jpg'),

('Pepsi', '百事可乐', 0.9, 0, 'pepsi.jpg'),

('Fanta', '芬达', 1.0, 0, 'fanta.jpg'),

('Sprite', '雪碧', 1.1, 0, 'sprite.jpg'),

('7Up', '七喜', 0.9, 0, '7Up.jpg'),

('Root Beer', '根汁汽水', 1.0, 0, 'rootBeer.jpg')

**Select Sample:**

SELECT \* From FYPJ.Soda\_Sales ORDER BY sale\_cost DESC, drink

**Update Sample:**

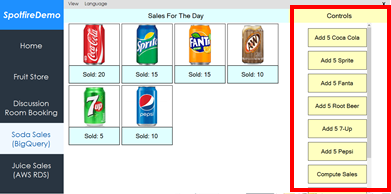
UPDATE FYPJ.Soda\_Sales

SET drink = 'Coca Cola'

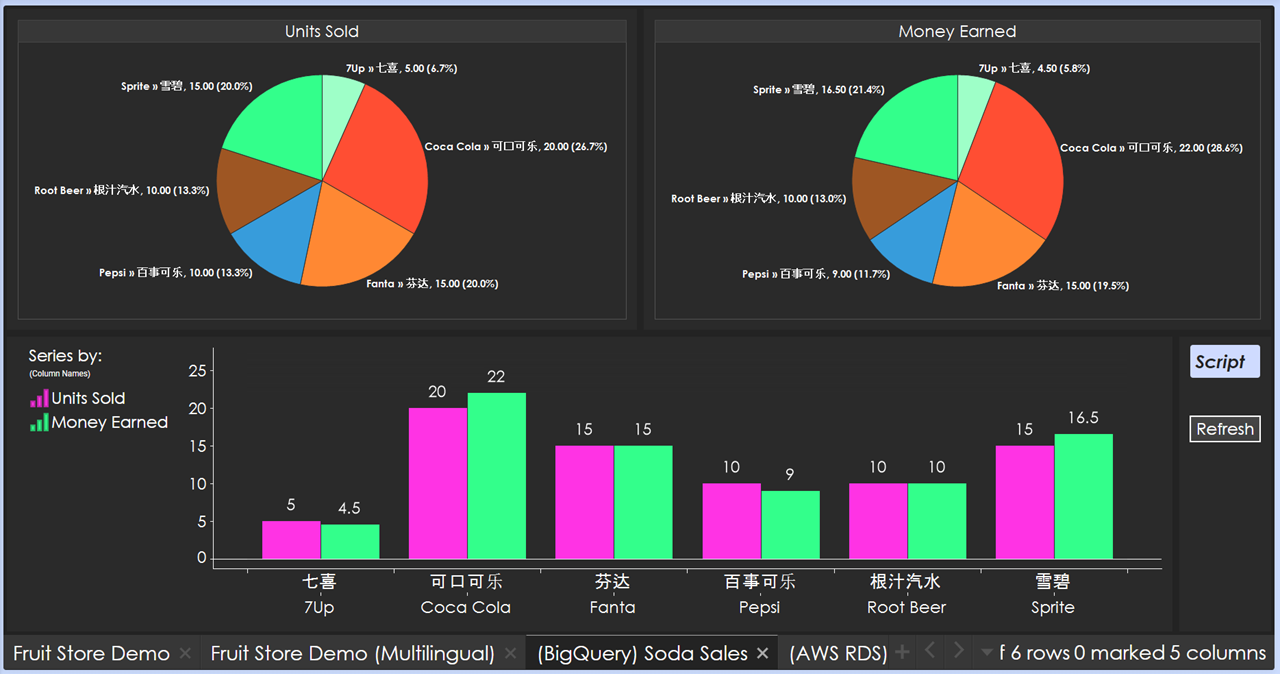
WHERE drink = 'Coca Coke'

**3.3 How To Use**

After opening the main page of the demo, the functionality of this demo works very similar to the FruitStore Demo. At the side, use the controls to add items, reset the items, and use an existing preset.

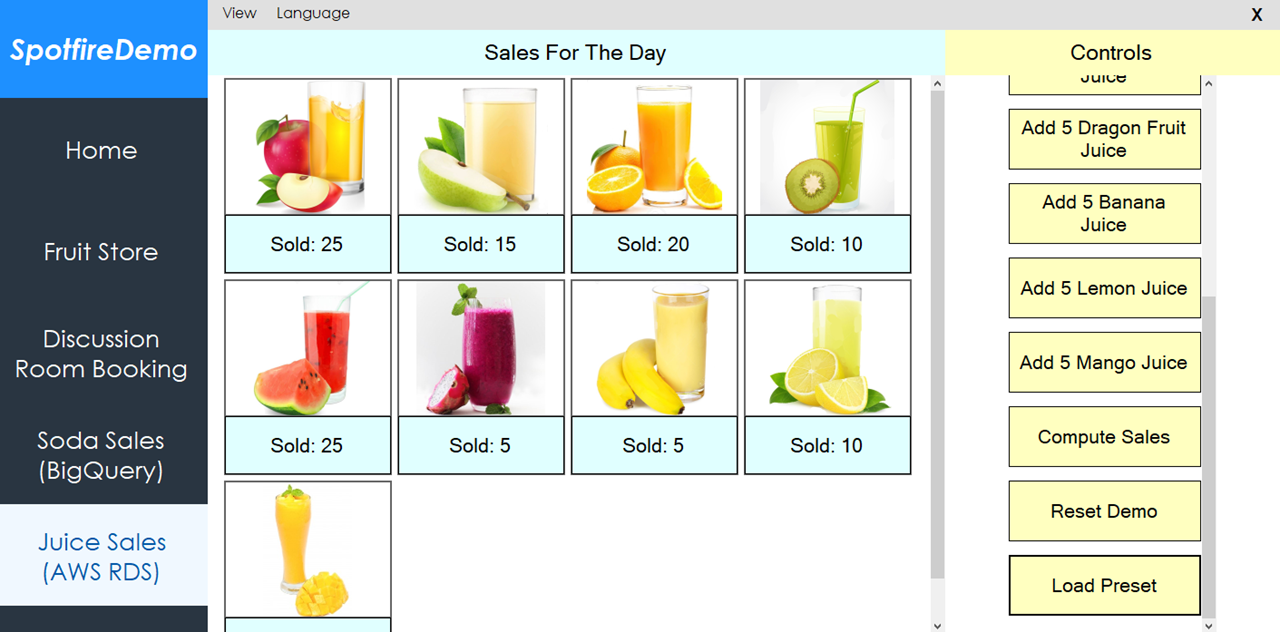


The final step is to link to Spotfire and create visualisations with this demo. An example of my demo is shown below.



**4. AWS RDS Demo**

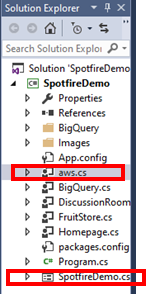
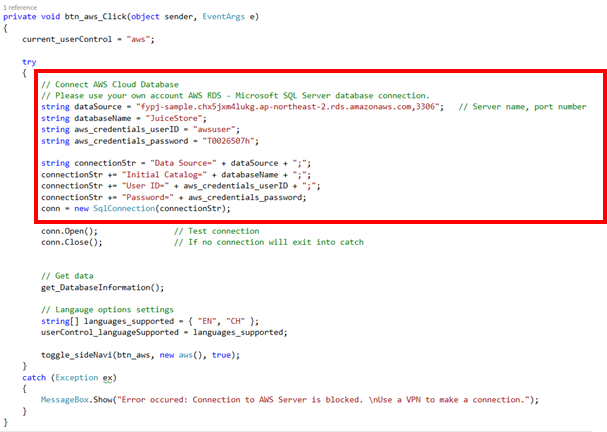
Similar to the Google BigQuery explanation, the demo was made to utilize AWS RDS as a cloud database source. The demo is also accessible via running the Spotfire Demo App, and selecting the “Juice Sales (AWS RDS) tab.



As previously mentioned in setup, you are required to use SQL Management Studio to first create a new database. Then you may view and edit the data via Visual Studio through the server explorer. And depending on the WiFi restriction, a VPN is required.

**4.1 Codes**

By opening the source codes of the project file “SpotfireDemo.cs” and “aws.cs”, you can see the codes to connect to the cloud database and to call/update the cloud database, respectively.



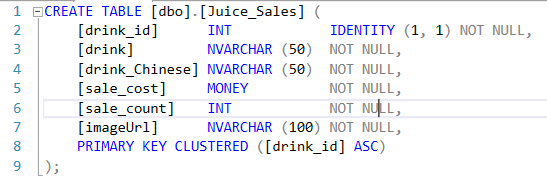
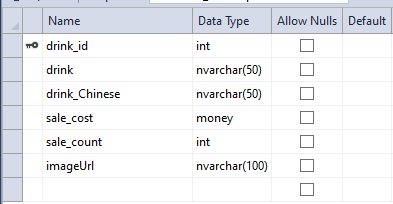
The demo codes for this portion is also very lengthy and complicated, hence, best to explore the source codes by yourself as well. Comments is also provided inside the source codes.

**Note**: When providing your own connection string, use the template in the image above.

**4.2 Copy Of Database Table**

Using your Visual Studio, copy the following database table schema and the data.

**Database Schema:**



CREATE TABLE [dbo].[Juice\_Sales] (

[drink\_id] INT IDENTITY (1, 1) NOT NULL,

[drink] NVARCHAR (50) NOT NULL,

[drink\_Chinese] NVARCHAR (50) NOT NULL,

[sale\_cost] MONEY NOT NULL,

[sale\_count] INT NOT NULL,

[imageUrl] NVARCHAR (100) NOT NULL,

PRIMARY KEY CLUSTERED ([drink\_id] ASC)

);

**Database Data:**



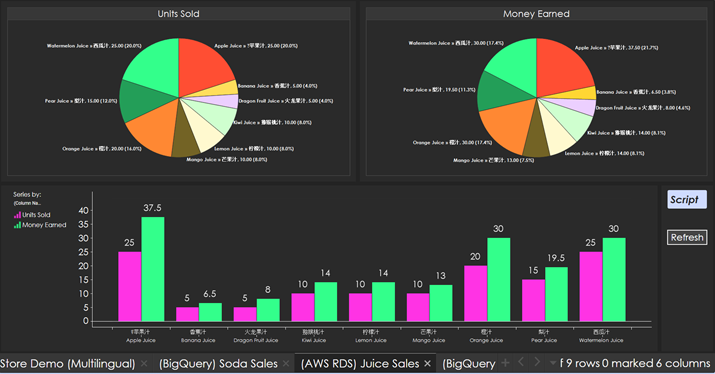
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | Apple Juice | ?苹果汁 | 1.5000 | 25 | appleJuice.jpg |
| 2 | Pear Juice | 梨汁 | 1.3000 | 15 | pearJuice.jpg |
| 3 | Orange Juice | 橙汁 | 1.5000 | 20 | orangeJuice.jpg |
| 4 | Kiwi Juice | 猕猴桃汁 | 1.4000 | 10 | kiwiJuice.jpg |
| 5 | Watermelon Juice | 西瓜汁 | 1.2000 | 25 | watermelonJuice.jpg |
| 6 | Dragon Fruit Juice | 火龙果汁 | 1.6000 | 5 | dragonFruitJuice.jpg |
| 7 | Banana Juice | 香蕉汁 | 1.3000 | 5 | bananaJuice.jpg |
| 8 | Lemon Juice | 柠檬汁 | 1.4000 | 10 | lemonJuice.jpg |
| 9 | Mango Juice | 芒果汁 | 1.3000 | 10 | mangoJuice.jpg |

**4.3 How To Use**

By using the demo app, you can control the with the buttons at the side of the page. The buttons are the main tools used to test the demo app to utilize the usage of AWS RDS database servers.



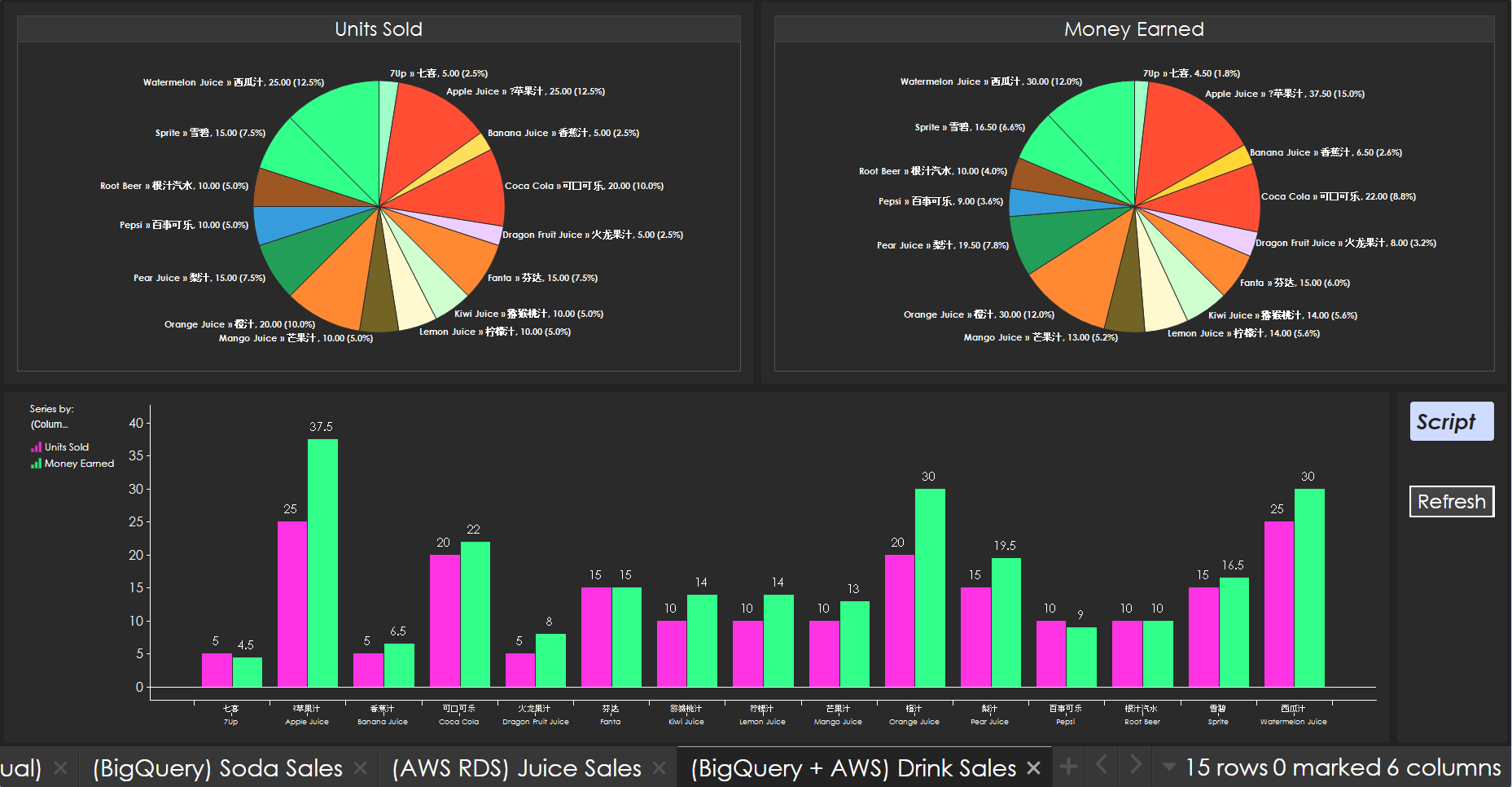
The final step is to link to Spotfire and create visualisations with this demo. An example of my demo is shown below.



**5. Combining Both Demos**

Once you have finished creating a TIBCO Spotfire Visualisation demo with each of the two demo cases, you can combine their data to form one big visualisation.

This is to show that Spotfire allows for combining two different database types together, as long as the database table schemas are similar and supports each other.



**6. Improving IoT Demo**

With the new implementation of cloud services, we are now able to replace the local database with a cloud database server to show better professionalism in the work, and as well as a test to see if all of them are compatible with each other.

The entities used for this portion of the demo will be the existing IoT WinForms Demo, the IoT Background project, a functional AWS RDS database server, and lastly TIBCO Spotfire.

Since it is just an improvement from the previous IoT Demo, I will not be diving into how to create as most of the programming will become very recurring during the improving of the existing project.

I will be sharing useful links and guides found online through development. These links will play a very helpful insight on how to manage, edit, and use DateTime and TimeSpan data-type variables. These variables are especially important for recording the additional details of a discussion room booking, such as when a student enters the room and when a student leaves the room.

They can be found alongside in the reference page.

**Below will be a list of changes from the previous IoT demo:**

1. AWS Cloud database instead of localDB
   1. New database table schema
2. Must now give a date, timeslot, and duration
3. Split the logs into Comment and Developer logs
4. Alarm now changed to:
   1. 3 Warning rings
   2. 1 Final Alarm ring

**How it now works:**

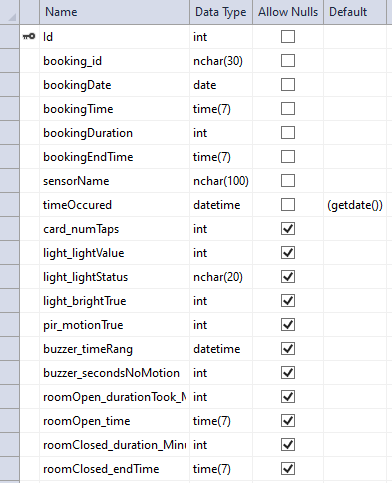
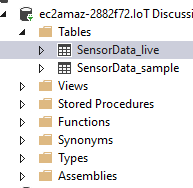
1. Open the demo page
2. Select a date, timeslot and duration
3. Press the confirm button, or cancel button to change input
4. Start the IoT Background Application
   1. The light will turn RED
   2. Simulation has started
5. Tap the correct student card
   1. The light will turn GREEN
   2. Student entered room
6. To End the simulation
   1. Final Alarm Rings
   2. Press the button

Create two database tables.

One for live, the other for sample (dummy data). Both tables have the same table schema.

This is because when applying to TIBCO Spotfire, we want to use live data for the auto-reload, while the sample data for being stored. It will help reduce lag when showing that the data is reloading as TIBCO Spotfire will have to reload lesser live data entries.

**New Database Schema:**

****

**Schema Code:**

CREATE TABLE [dbo].[SensorData\_live] (

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

[booking\_id] NCHAR (30) NOT NULL,

[bookingDate] DATE NOT NULL,

[bookingTime] TIME (7) NOT NULL,

[bookingDuration] INT NOT NULL,

[bookingEndTime] TIME (7) NOT NULL,

[sensorName] NCHAR (100) NOT NULL,

[timeOccured] DATETIME DEFAULT (getdate()) NOT NULL,

[card\_numTaps] INT NULL,

[light\_lightValue] INT NULL,

[light\_lightStatus] NCHAR (20) NULL,

[light\_brightTrue] INT NULL,

[pir\_motionTrue] INT NULL,

[buzzer\_timeRang] DATETIME NULL,

[buzzer\_secondsNoMotion] INT NULL,

[roomOpen\_durationTook\_Minutes] INT NULL,

[roomOpen\_time] TIME (7) NULL,

[roomClosed\_duration\_Minutes] INT NULL,

[roomClosed\_endTime] TIME (7) NULL,

PRIMARY KEY CLUSTERED ([Id] ASC)

);

CREATE TABLE [dbo].[SensorData\_sample] (

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

[booking\_id] NCHAR (30) NOT NULL,

[bookingDate] DATE NOT NULL,

[bookingTime] TIME (7) NOT NULL,

[bookingDuration] INT NOT NULL,

[bookingEndTime] TIME (7) NOT NULL,

[sensorName] NCHAR (100) NOT NULL,

[timeOccured] DATETIME DEFAULT (getdate()) NOT NULL,

[card\_numTaps] INT NULL,

[light\_lightValue] INT NULL,

[light\_lightStatus] NCHAR (20) NULL,

[light\_brightTrue] INT NULL,

[pir\_motionTrue] INT NULL,

[buzzer\_timeRang] DATETIME NULL,

[buzzer\_secondsNoMotion] INT NULL,

[roomOpen\_durationTook\_Minutes] INT NULL,

[roomOpen\_time] TIME (7) NULL,

[roomClosed\_duration\_Minutes] INT NULL,

[roomClosed\_endTime] TIME (7) NULL,

PRIMARY KEY CLUSTERED ([Id] ASC)

);

**7. References**

**7.1 General**

**7.1.1 Prerequisites**

Google Cloud:

<https://cloud.google.com/>

AWS Amazon:

<https://aws.amazon.com/>

Microsoft SQL Server 2012 Express Download:

<https://www.microsoft.com/en-sg/download/details.aspx?id=29062>

**7.2 Platforms**

**7.2.1 Window Forms C#**

Duplicate WinForms form: <https://stackoverflow.com/questions/1268591/how-to-easily-duplicate-a-windows-form-in-visual-studio>

Programmatic Button Click:

<https://stackoverflow.com/questions/16792160/how-to-trigger-a-button-click-in-my-code>

C# ArrayList:

<https://www.geeksforgeeks.org/c-sharp-arraylist-class/>

<https://www.tutorialspoint.com/csharp/csharp_arrays>

ArrayList Check Value Exist:

<https://stackoverflow.com/questions/7867377/checking-if-a-string-array-contains-a-value-and-if-so-getting-its-position>

Array IsNullOrEmpty:

<https://stackoverflow.com/questions/8560106/isnullorempty-equivalent-for-array-c-sharp>

Foreach Loop:

<https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/foreach-in>

Round 2 Decimal:

<https://stackoverflow.com/questions/164926/how-do-i-round-a-decimal-value-to-2-decimal-places-for-output-on-a-page>

Convert String To Float:

<https://stackoverflow.com/questions/11202673/converting-string-to-float-in-c-sharp>

InnerException property of TypeInitializationException:

<https://stackoverflow.com/questions/4398334/the-type-initializer-for-myclass-threw-an-exception>

Get Number Of Rows With SqlDataReader:

<https://stackoverflow.com/questions/1383315/how-to-get-number-of-rows-using-sqldatareader-in-c-sharp>

**7.2.2 Google BigQuery**

**7.2.2.1 Pricing**

Pricing:

<https://cloud.google.com/bigquery-ml/pricing>

**7.2.2.2 Official Notes**

Using BigQuery Sandbox:

<https://cloud.google.com/bigquery/docs/sandbox?hl=en_US&_ga=2.46468704.-610591827.1563784934&_gac=1.150467908.1563784965.Cj0KCQjwvdXpBRCoARIsAMJSKqJ3Kd73OClcWE4Q04qraecFAQ9bWDTJoOArYWU0l6QCZIIviJm5mT4aAjnEEALw_wcB>

Wildcard Tables:

<https://cloud.google.com/bigquery/docs/reference/standard-sql/wildcard-table-reference>

Data Manipulation Language (DML) Syntax:

<https://cloud.google.com/bigquery/docs/reference/standard-sql/dml-syntax>

BigQuery Classic Web UI:

<https://cloud.google.com/bigquery/docs/bigquery-classic-ui>

Google.Cloud.BigQuery.V2 Demo Codes:

<https://googleapis.github.io/google-cloud-dotnet/docs/Google.Cloud.BigQuery.V2/index.html>

BigQuery Client Library Setup:

<https://cloud.google.com/bigquery/docs/reference/libraries>

**7.2.2.3 Community Notes**

SQL Cheat Sheet:

<http://www.sqltutorial.org/sql-cheat-sheet/>

C# .NET Google Application Credentials JSON file:

<https://stackoverflow.com/questions/37626200/c-sharp-net-mvc-set-path-to-google-application-credentials-json-file>

BigQuery w/ C# .NET Simple Installation Guide:

<https://medium.com/@gabriel.faraday.barros/asp-net-core-api-google-bigquery-bd69c4fde45b>

Using BigQuery with C# Video:

<https://www.youtube.com/watch?v=zOjOg_Lp5Nc>

SELECT return default: <https://stackoverflow.com/questions/11503463/how-to-return-default-value-from-sql-query>

INSERT statement:

<https://www.w3schools.com/sql/sql_insert.asp>

**7.2.3 AWS RDS (Microsoft SQL Edition)**

**7.2.3.1 Official Notes**

Pricing:

<https://aws.amazon.com/rds/sqlserver/pricing/>

**7.2.3.2 Database SQL Queries**

Sql Null Functions:

<https://www.w3schools.com/sql/sql_isnull.asp>

Sql Where Clause:

<https://www.w3schools.com/sql/sql_where.asp>

Sql Aliases:

<https://www.w3schools.com/sql/sql_alias.asp>

Sql Count(), Avg(), Sum() Functions:

<https://www.w3schools.com/sql/sql_count_avg_sum.asp>

Adding Parameters to Commands:

<https://csharp-station.com/Tutorial/AdoDotNet/Lesson06>

**7.2.3.3 Setup Help**

Setting Up for Amazon RDS:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_SettingUp.html#CHAP_SettingUp.SecurityGroup>

Amazon Virtual Private Cloud VPCs and Amazon RDS:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_VPC.WorkingWithRDSInstanceinaVPC.html#USER_VPC.CreatingVPC>

**7.2.3.4 Additional Setup Help**

AWS Connection With C# tutorial:

<https://www.youtube.com/watch?v=EBTBCBUGR4I&fbclid=IwAR1xZIttetMUBJhBG1zO80DFIe3vjPGQhVArNpXdGrG3D2zvy54_13su1KY>

How to install SQL Server Management Studio 2012:

<https://www.hivelocity.net/kb/how-to-install-sql-server-management-studio-2012-on-windows-server-2012/>

SQL Server Database connection error on Amazon RDS:

<https://saaction.blogspot.com/2017/05/sql-server-database-connection-error-on.html?fbclid=IwAR2kA3IPkaBQXYri7Wtv8CUvZT1JNsBQrmdeiu-Bv4K9wU_dEN7tmZX09Ps>

Cannot connect to my RDS database:

<https://forums.aws.amazon.com/thread.jspa?threadID=115067>

Connect to SQL Server Error:

<https://www.experts-exchange.com/questions/28746979/Connect-to-SQL-Server-Error.html?fbclid=IwAR3DTBRdnStikbw-VVCExGYucivBtdFGzVZaeCLRVBehHC0rfph3MjDGrlw>

Fix SQL Server Error 53: Could not open connection on SQL Server:

<https://techyaz.com/sql-server/troubleshooting/fix-error-53-not-open-connection-sql-server/>

“A network- related or instance-specific error occurred”: <https://www.lansweeper.com/knowledgebase/a-network-related-or-instance-specific-error-occurred/>

How to fix error ‘Named Pipes Provider, error 40 - Could not open a connection to SQL Server’:

<https://stackoverflow.com/questions/9945409/how-do-i-fix-the-error-named-pipes-provider-error-40-could-not-open-a-connec?fbclid=IwAR2CVje64W609EYfBLfbIFbAwXCzzZzlJ0whAVmn8dpStcoettt8YRnjf14>

How to fix SQL connection in SQL Server Management Studio 2017:

<https://www.youtube.com/watch?v=9O4-2QjoEVo>

MSSQLSERVER service on Local Computer, started then stopped:

<https://stackoverflow.com/questions/35080354/the-mssqlserver-service-on-local-computer-started-and-then-stopped-some-service>

**7.2.3.5 Potential Bugs Solutions**

How to open ports on the Windows Firewall:

<https://serverfault.com/questions/221075/how-to-know-currently-open-ports-on-the-windows-firewall>

TCPView Download And Guide:

<https://docs.microsoft.com/en-us/sysinternals/downloads/tcpview?fbclid=IwAR3ZkBndUp837A5juzWaect4FeWRA4oCCtPEK4COzgEa3CbOJ4Hed5PMH64>

SQL Server TCP and UDP Ports:

<https://www.itprotoday.com/sql-server/sql-server-tcp-and-udp-ports?fbclid=IwAR0EyxoOjOFFyGJFmtNKYw93nlHmuHaHY-nCxFly6L9OrDviVMkITdrLN5s>

Overview of common TCP and UDP default ports:

<https://www.examcollection.com/certification-training/network-plus-overview-of-common-tcp-and-udp-default-ports.html?fbclid=IwAR2nAMZXxjqPke-gN_9XNyoTev1YVn71du-QVgbTyFbjjbLh34HHOjGGqr8>

How to specify a Port Number to Connect to SQL Server Instance:

<http://zarez.net/?p=3305>

Guide To Change AWS Port:

<https://www.cloudconformity.com/conformity-rules/RDS/rds-default-port.html>

**7.2.3.6 Others**

**7.2.3.6.1 VPN Solution**

WindScribe:

<https://windscribe.com/>

**7.2.3.6.2 SQL Server Configuration Manager Guide**

SQL Server Configuration Manager:

<http://lexisnexis.custhelp.com/app/answers/answer_view/a_id/1095989/~/sql-server-configuration-manager-general-information>

**7.2.3.6.3 TPCView Guide**

TPCView:

<https://docs.microsoft.com/en-us/sysinternals/downloads/tcpview?fbclid=IwAR1w-MGgOCf49Ddlkz-sguGKl_Y0Od_HWrMmduubqMjiTz6Uhq6n7rGIM-U>

**7.3 Language**

Drinks in Chinese:

<https://blogs.transparent.com/chinese/chinese-vocabulary-drinks/>

<https://www.fluentu.com/blog/chinese/2013/03/08/essential-chinese-vocabulary-word-list-drinks/>

<https://blogs.transparent.com/chinese/get-your-drink-on/>

**7.4 Improving IoT Demo**

**7.4.1 Date, DateTime, Time, Timespan Replated**

DateTime Format:

<https://www.c-sharpcorner.com/blogs/date-and-time-format-in-c-sharp-programming1>

Date And Time Data Types and Functions (Transact-SQL)

<https://docs.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql?view=sql-server-2017>

SQL Server GETDATE Function:

<http://www.sqlservertutorial.net/sql-server-date-functions/sql-server-getdate-function/>

Get Current Time:

<https://stackoverflow.com/questions/296920/how-do-you-get-the-current-time-of-day>

Convert String To Time:

<https://stackoverflow.com/questions/18919530/convert-string-to-time>

Convert String To DateTime:

<http://net-informations.com/q/faq/stringdate.html>

Calculate Minutes Between Two Dates:

<https://www.tutorialspoint.com/calculate-minutes-between-two-dates-in-chash>

DateTime.Add() Method:

<https://www.geeksforgeeks.org/datetime-add-method-in-c-sharp/>

DateTime.AddMinutes(Double) Method:

<https://docs.microsoft.com/en-us/dotnet/api/system.datetime.addminutes?view=netframework-4.8>

Custom TimeSpan format:

<https://docs.microsoft.com/en-us/dotnet/standard/base-types/custom-timespan-format-strings#mmSpecifier>

**7.4.2 Additional C# Related**

Array Reverse Method:

<https://docs.microsoft.com/en-us/dotnet/api/system.array.reverse?view=netframework-4.8>

ListBox Word Wrap alternative:

<https://social.msdn.microsoft.com/Forums/vstudio/en-US/9b69fcd9-e9af-4f06-96e9-f164ee6e5428/list-box-word-wrap?forum=vbgeneral>

String.Replace() Method:

<https://www.geeksforgeeks.org/c-sharp-replace-method/>

Foreach Loop, Find Last Iteration:

<https://stackoverflow.com/questions/7476174/foreach-loop-determine-which-is-the-last-iteration-of-the-loop>

Parse Strings Using String.Split:

<https://docs.microsoft.com/en-us/dotnet/csharp/how-to/parse-strings-using-split>

Convert Double To Nearest Integer:

<https://stackoverflow.com/questions/633335/how-might-i-convert-a-double-to-the-nearest-integer-value/633340>

Generate Random Int Number:

<https://stackoverflow.com/questions/2706500/how-do-i-generate-a-random-int-number>

Generate Random Alphanumeric String:

<https://stackoverflow.com/questions/1344221/how-can-i-generate-random-alphanumeric-strings>