SQL Server vNext on Linux Ubuntu - Part 2

In Part 1 of this article, we have seen

- How to install SQL Server vNext on Ubuntu
- How to install SQLCMD / BCP tools and ODBC drivers
- Run basic queries and create our own new databases on Linux
- Restore a database from SQL Server on Windows to SQL Server vNext on Linux
- Limitation of current SQL Server vNext CTP 1.1 (December 2016)

In Part 2 of this article we will see how to:

- Install Visual Studio Code on Linux Ubuntu VM
- Install MSSQL extension for Visual Studio Code
- Connect to SQL Server vNext using VS code and run T-SQL codes against a new database
- Install .NET core and C# extension for VS Code, and also build a sample project in C# to connect and retrieve data from SQL Server vNext
- Connect to SQL Server vNext from windows server using SQL Server Management Studio

1- Install Visual Studio on Linux Ubuntu VM

Go to the following link
https://code.visualstudio.com/Download

And select .deb for Debian-based distributions such as Ubuntu (for Red Hat-based distributions select .rpm file)

From Ubuntu terminal console (or Putty) run

$ sudo dpkg -i <file>.deb
$ sudo apt-get install -f

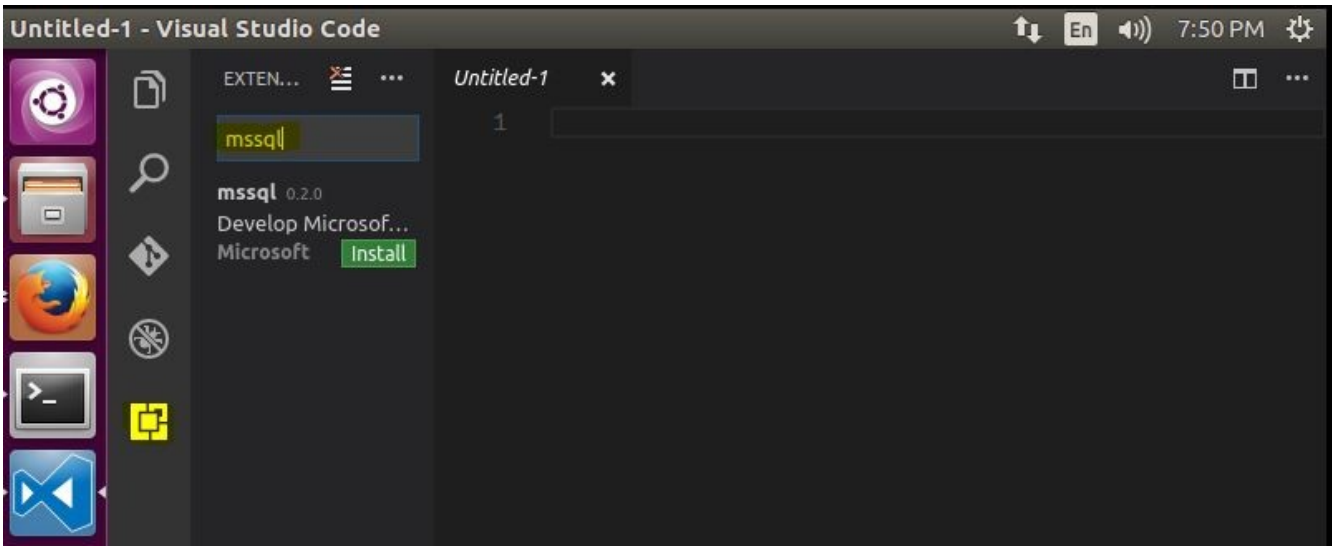
In our case we have downloaded .deb file in the Desktop path, so the command is
$ sudo dpkg -i /home/mbouarroudj/Downloads/code_1.8.1-1482158209_amd64.deb

```
mbouarroudj@Ubuntu1604:~$ dir /home/mbouarroudj/Downloads/
code_1.8.1-1482158209_amd64.deb
mbouarroudj@Ubuntu1604:~$ sudo dpkg -i /home/mbouarroudj/Downloads/code_1.8.1-14
82158209_amd64.deb
Selecting previously unselected package code.
(Reading database ... 205740 files and directories currently installed.)
Preparing to unpack .../code_1.8.1-1482158209_amd64.deb ...
Unpacking code (1.8.1-1482158209) ...
Setting up code (1.8.1-1482158209) ...
Processing triggers for gnome-menus (3.13.3-6ubuntu3.1) ...
Processing triggers for desktop-file-utils (0.22-1ubuntu5) ...
Processing triggers for bamfdaemon (0.5.3~bzr0+16.04.20160701-0ubuntu1) ...
Rebuilding /usr/share/applications/bamf-2.index...
Processing triggers for mime-support (3.59ubuntu1) ...
mbouarroudj@Ubuntu1604:~$ sudo apt-get install -f
Reading package lists... Done
Building dependency tree
Reading state information... Done
0 upgraded, 0 newly installed, 0 to remove and 184 not upgraded.
mbouarroudj@Ubuntu1604:~$
```

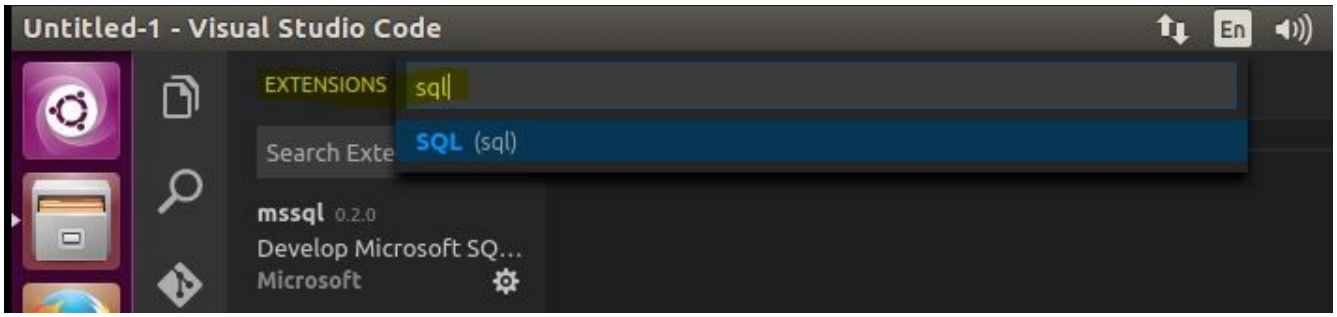
Once the installation is complete, in Ubuntu Launcher bar click the Search icon and type “visual” word and then select Visual Studio Code (to create a shortcut drag the icon in Ubuntu launcher)

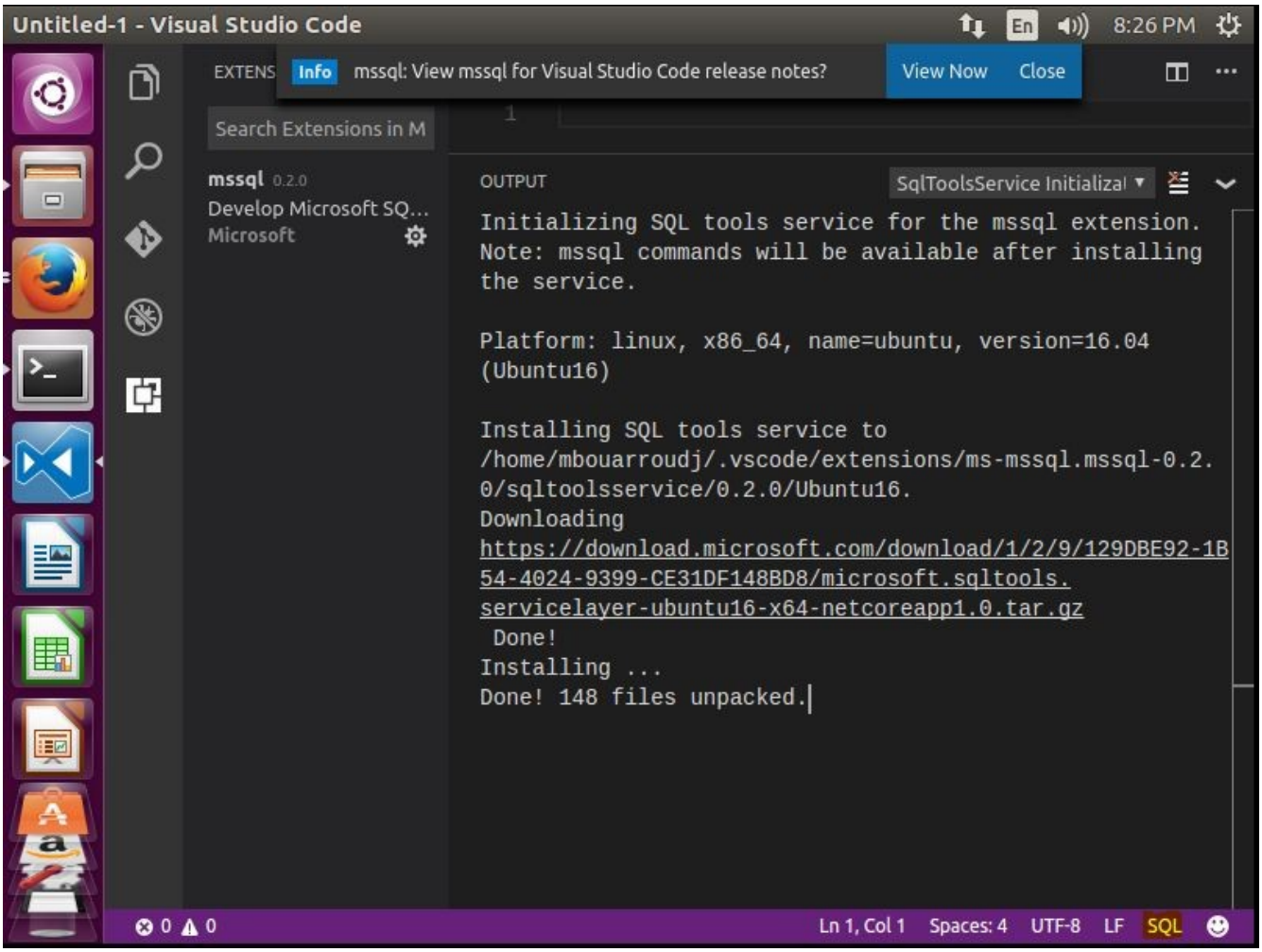
2- Install MSSQL extension for Visual Studio Code

Click extension icon and type “mssql” as shown in this print screen and hit Install button



Wait a couple of minutes for installation to be completed, and in VS Code bar click Plain Text and then enter SQL in the Extension field to switch to SQL mode





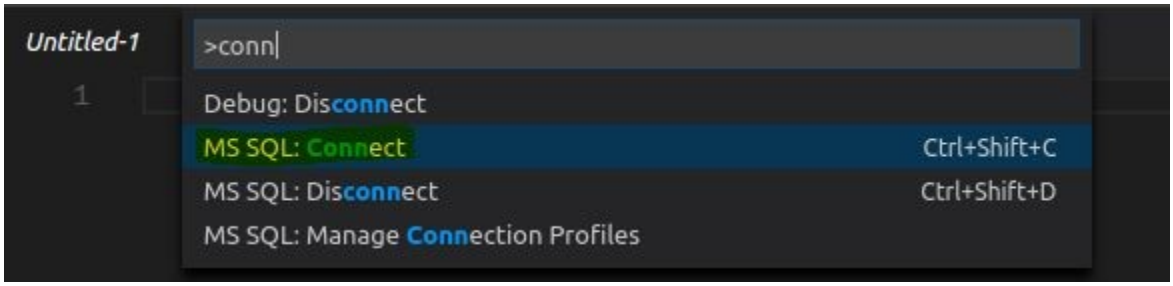
3- Connect to SQL Server vNext using VS code and run T-SQL codes against a new database

To connect to FirstDbOnlinux database we have create on Part 1, follow these steps

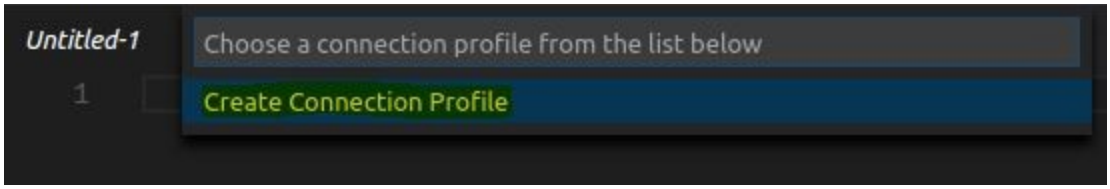
- Click File / New Window, and swich from “Plan Text” to “SQL” language mode

- Press CTRL+SHIFT+P or F1 to open the Command Palette and then type “sql” to display mssql commands.

Select the MS SQL: Connect command and press ENTER



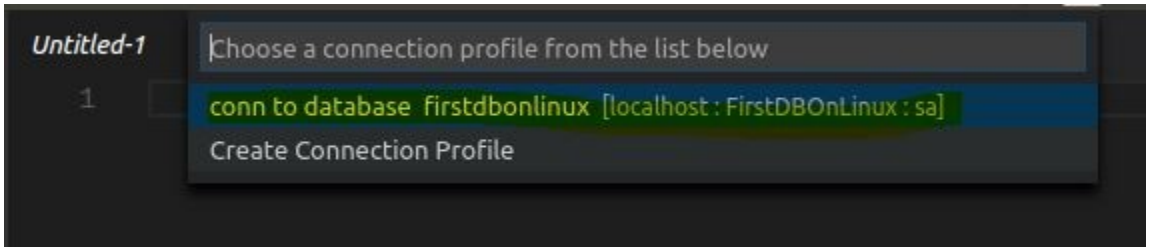
- Select Create Connection Profile to create a connection profile for our SQL Server instance, this is an optional step, you can connect without creating a profile



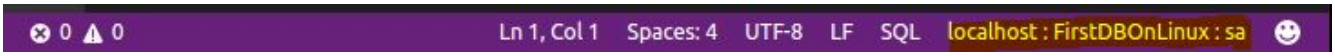
- Enter the following values

HostName: hostname
Database: firstdbonlinux -- this is a new database we created in Part 1
User: SA
Password: your password
Name for this profile (optional)

- Press CTRL+SHIFT+C and select the profile you created in the previous step

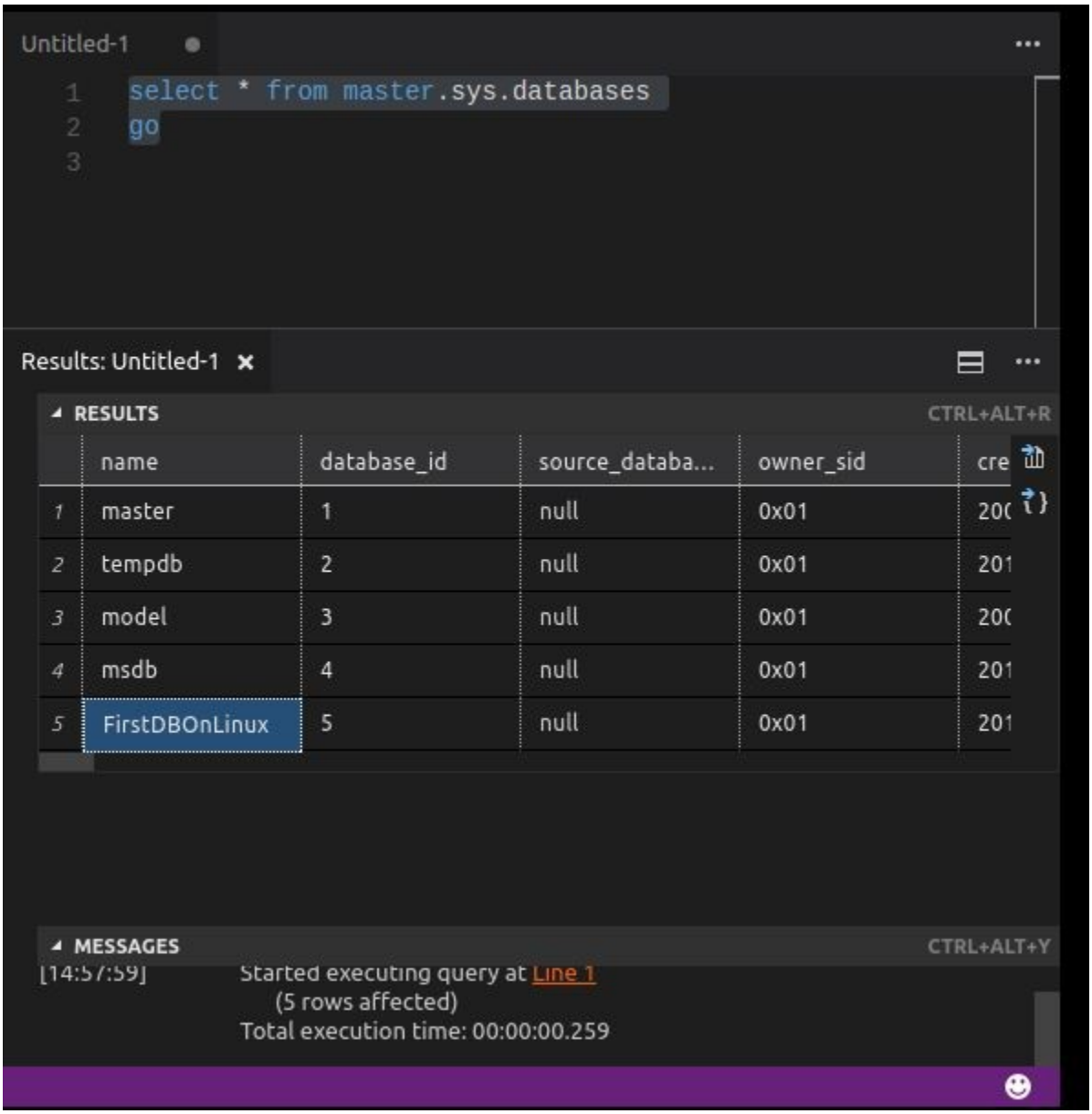


Verify your connection in the status bar.



- At this level, you are ready to start writing queries and execute them against your database with CTRL+SHIFT+E shortcut

You will notice that IntelliSense, Keyword completion are available. You can Select Toggle Editor Group Layout from the menu to switch to vertical or horizontal split layout.



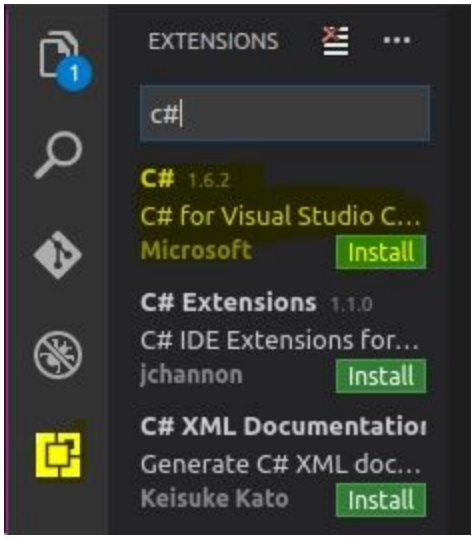
Unlike SQL Server Management Studio you can save the result in .csv or Jason format

4- Write a sample code in C# to connect and retrieve data from SQL Server vNext

Before you start to write our first C# project we have to install C# code extension and .Net,
please note that VS code with C# extension is not required and you can use the editor of your choice, but you will not have IntelliSense, Keyword completion, and debugger

4.1. Installation of C# code extension

In Visual code, click on extension icon and type C#, select C# for Visual Code and hit the install button



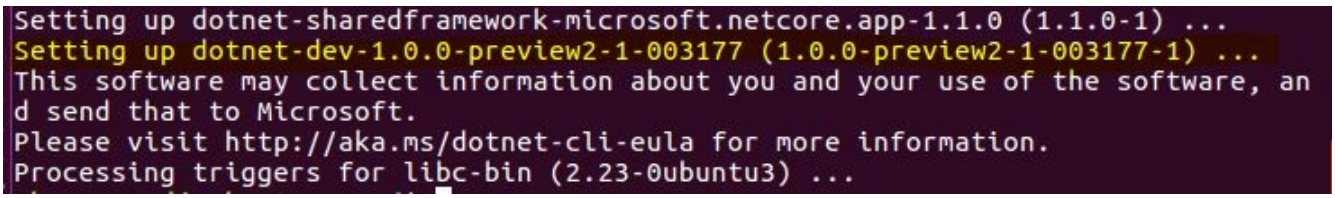
Once the installation is complete, in VS Code bar you can hit Select Language Mode to switch between C# and SQL modes

4.2. Installation of .NET core

- In Ubuntu Terminal or Visual Studio Code integrated terminal, run the following commands

sudo sh -c 'echo "deb [arch=amd64] https://apt-mo.trafficmanager.net/repos/dotnet-release/ xenial main" > /etc/apt/sources.list.d/dotnetdev.list'
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 417A0893
sudo apt-get update

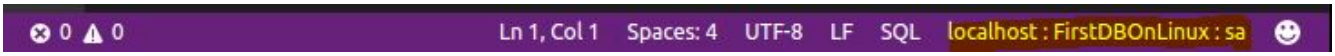
and then run the installation
sudo apt-get install dotnet-dev-1.0.0-preview2.1-003177



Now we are ready to write our C# application to retrieve data from SQL Server vNext hosted on Linux Ubuntu

4.3. Create a new Project with C# to connect and retrieve data to/from SQL Server vNext

- Before we create our C# project in Visual Studio Code run the following T-SQL to create a table dbo.Country and new stored procedure dbo.GetCountryById, make sure you are connected to FirstDBOnLinux, the database we have created in Part 1 of this article




```
create table dbo.country
(
id           int identity(1,1) primary key,
country_name varchar(32) not null,
country_code varchar(16) not null,
)

insert into dbo.country(country_name, country_code) values('Algeria', '213'),('Brazil', '55'),
('Canada', '1')
go

create proc dbo.GetCountryById
(
@id int
)

as
set nocount on

select *
from dbo.country
where id = @id
go

exec dbo.GetCountryById @id = 2
go
```

- In Ubuntu or VS Code Terminal console, enter the following commands to create our first project in dotnetprojects\countryget subdirectory

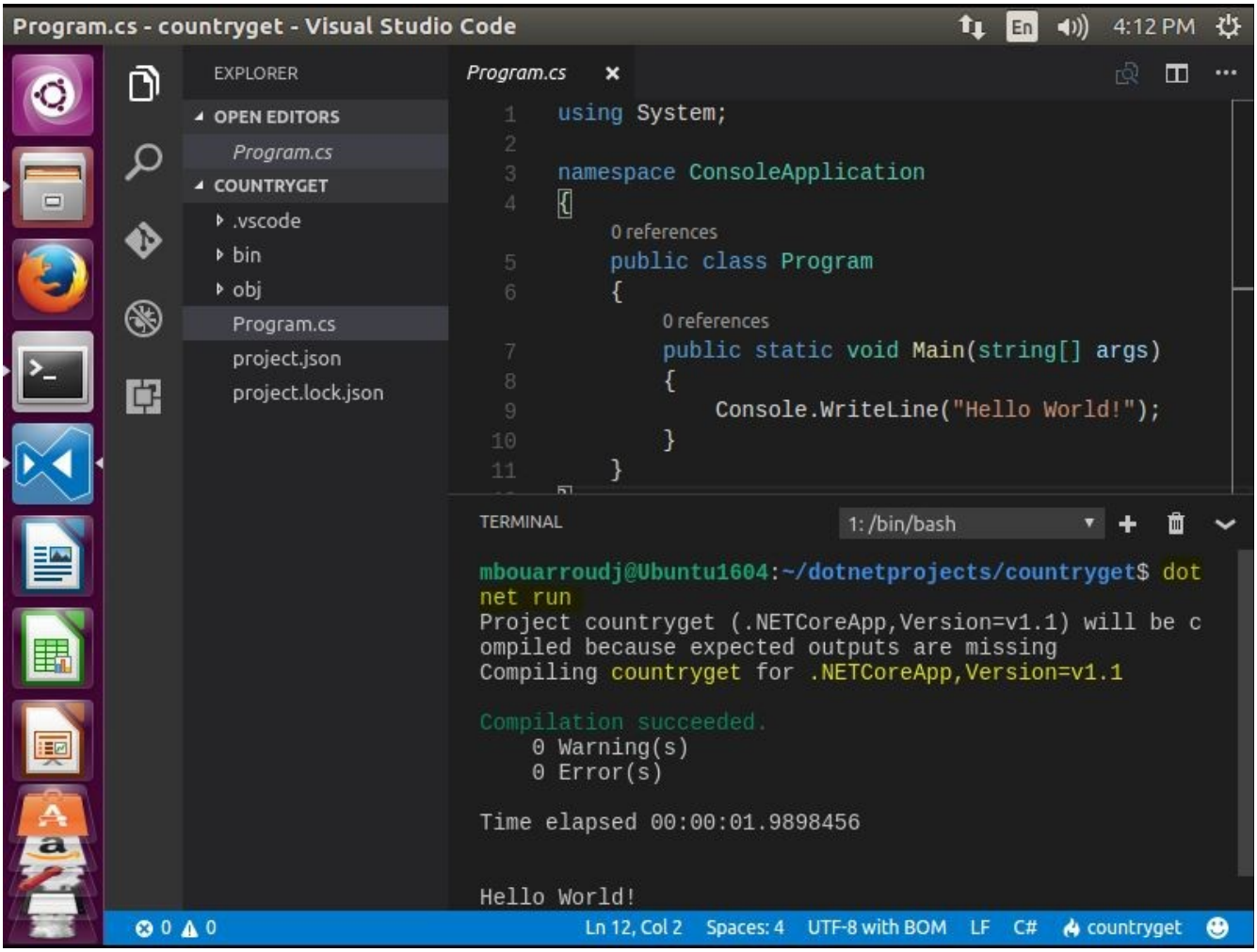
$ mkdir dotnetprojects
$ cd dotnetprojects
$ mkdir countryget
$ cd countryget
$ dotnet new

```
mbouarroudj@Ubuntu1604:~$ mkdir dotnetprojects
mbouarroudj@Ubuntu1604:~$ cd dotnetprojects
mbouarroudj@Ubuntu1604:~/dotnetprojects$ mkdir countryget
mbouarroudj@Ubuntu1604:~/dotnetprojects$ cd countryget
mbouarroudj@Ubuntu1604:~/dotnetprojects/countryget$ dotnet new

Welcome to .NET Core!
---------------------
Learn more about .NET Core @ https://aka.ms/dotnet-docs. Use dotnet --help to se
e available commands or go to https://aka.ms/dotnet-cli-docs.
Telemetry
-------------
The .NET Core tools collect usage data in order to improve your experience. The
data is anonymous and does not include commandline arguments. The data is collec
ted by Microsoft and shared with the community.
You can opt out of telemetry by setting a DOTNET_CLI_TELEMETRY_OPTOUT environmen
t variable to 1 using your favorite shell.
You can read more about .NET Core tools telemetry @ https://aka.ms/dotnet-cli-te
lemetry.
Configuring...
```

- In VS Code, open countryget subdirectory and hit “Yes” for messages related to restore packages and missing assets. At the end, our project will be opened and ready to run, click View Menu and select Integrated Terminal to open a terminal window and enter the following command to build and run the project, this will display: Hello word! message

$ dotnet run



Let's now change the code in our project to connect to SQL Server vNext and retrieve data from FirstDBOnLinux database

Open Program.cs file and change the code to this:

```
using System;
using System.Text;
using System.Data;
using System.Data.SqlClient;

namespace ConsoleApplication
{
    public class Program
    {
        public static void Main(string[] args)
        {
            try
            {
                // Build connection string, in real word the connection string is in separate
config file to ease the deployment process
                SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
                builder.DataSource = "localhost";   // your vNext instance
                builder.UserID = "sa";              // in real word you have to use an
application user with no sysadmin access
```

```
                builder.Password = "a4t18288-";      // in real word your pwd should be
encrypted
                builder.InitialCatalog = "FirstDBOnLinux";

                // Connect to SQL Server vNext
                Console.WriteLine("Connecting to database " + builder.DataSource + "." +
builder.InitialCatalog + "...");
                using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
                {
                    connection.Open();
                    Console.WriteLine("Connection succeeded");

                    SqlCommand cmd = new SqlCommand();
                    cmd.Connection = connection;
                    cmd.CommandText = "dbo.GetCountryById";
                    cmd.CommandType = CommandType.StoredProcedure;

                    int id = 2;  // for purposes of this demo we used a hard-coded value,
otherwise you should read from Console.ReadKey(true)
                    cmd.Parameters.Add("@Id", SqlDbType.Int, 4).Value = id;

                    SqlDataReader sqlReader = cmd.ExecuteReader();
                    while (sqlReader.Read())
                    {
                        Console.WriteLine(string.Format("Country Id: {0}, Country Name: {1},
Country Code: {2}", sqlReader.GetInt32(0), sqlReader.GetSqlString(1),
sqlReader.GetSqlString(2)));
                    }
                }
            }
            catch (SqlException e)
            {
                Console.WriteLine(e.Message.ToString());
            }
            finally
            {
                // check and close the connection
            }


            Console.ReadKey(true);
        }
    }

}
```

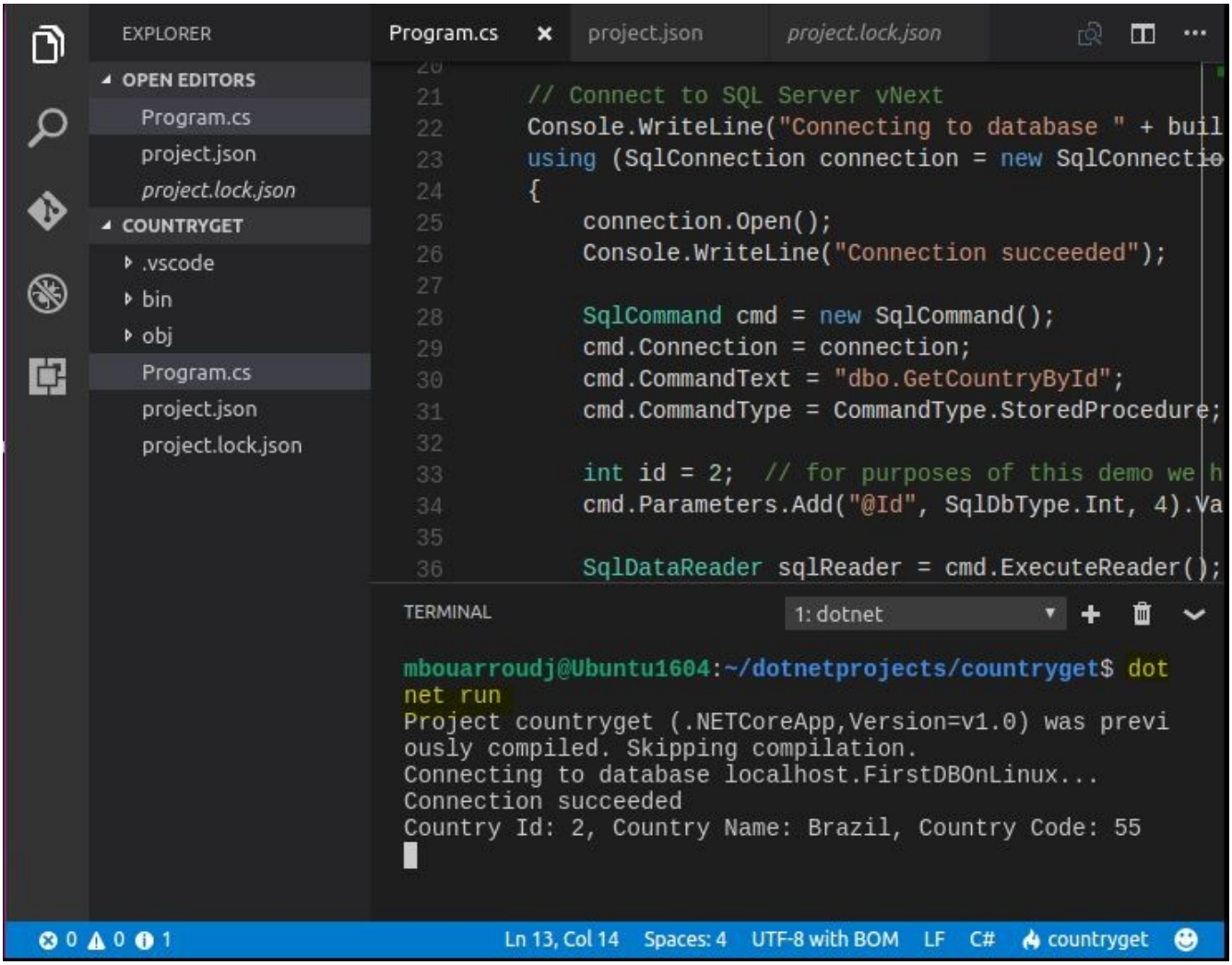
Open project.json file and change the code to this:

```
{
  "version": "1.0.0-*",
  "buildOptions": {
    "debugType": "portable",
    "emitEntryPoint": true
  },
  "dependencies": {
    "System.Data.SqlClient": "4.3.0"
  },
  "frameworks": {
    "netcoreapp1.0": {
      "dependencies": {
```

```
        "Microsoft.NETCore.App": {
          "type": "platform",
          "version": "1.1.0"
        }
      },
      "imports": "dnxcore50"
    }
  }
}
```

Restore the dependencies in project.json by running the following commands in VS Code Terminal, otherwise, you will face this error: The type or namespace 'Data' does not exist
$ dotnet restore

Run the following command to rebuild and rerun the project
$ dotnet run



Please note that DataAdapters/DataSet/DataTable/DataRow classes are not available in .NET core at the moment of our tests (Version 1.1.0)

5- Connect to SQL Server vNext on Linux from windows using SQL Server Management Studio

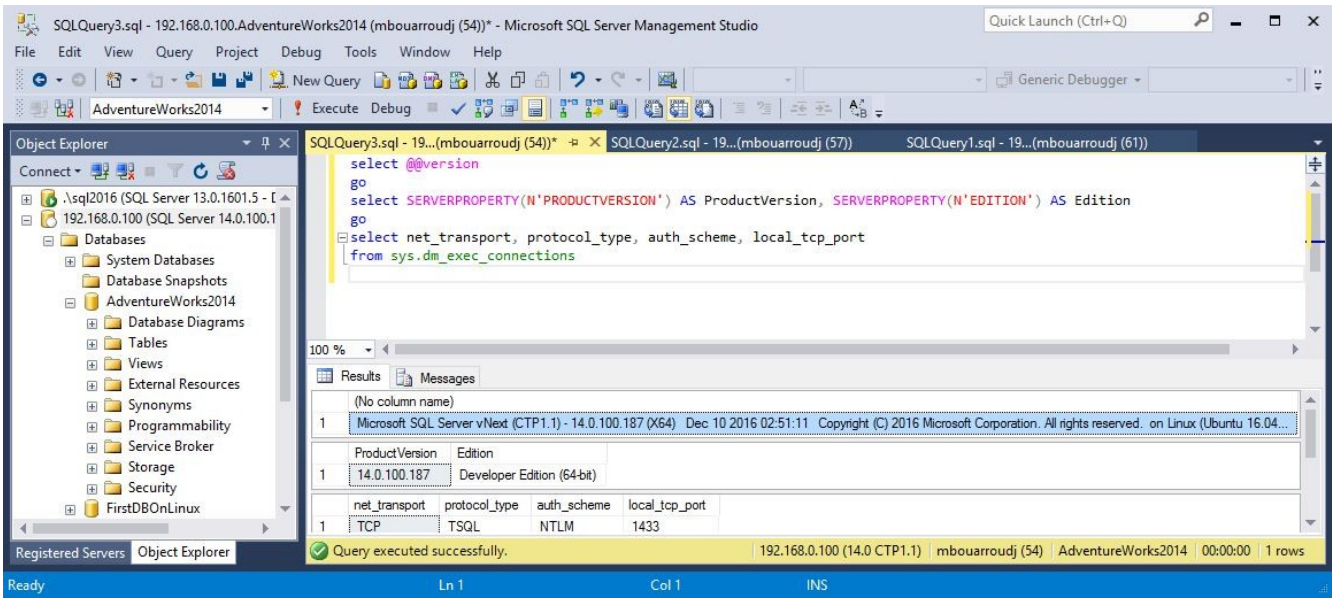
We have seen how to connect to SQL Server vNext from Linux server using SQL Cmd, Visual Studio Code, and C# project, you can also connect to SQL Server vNext from windows servers using SSMS (SQL Server Management Studio) or SQL Server Data Tools for Visual Studio

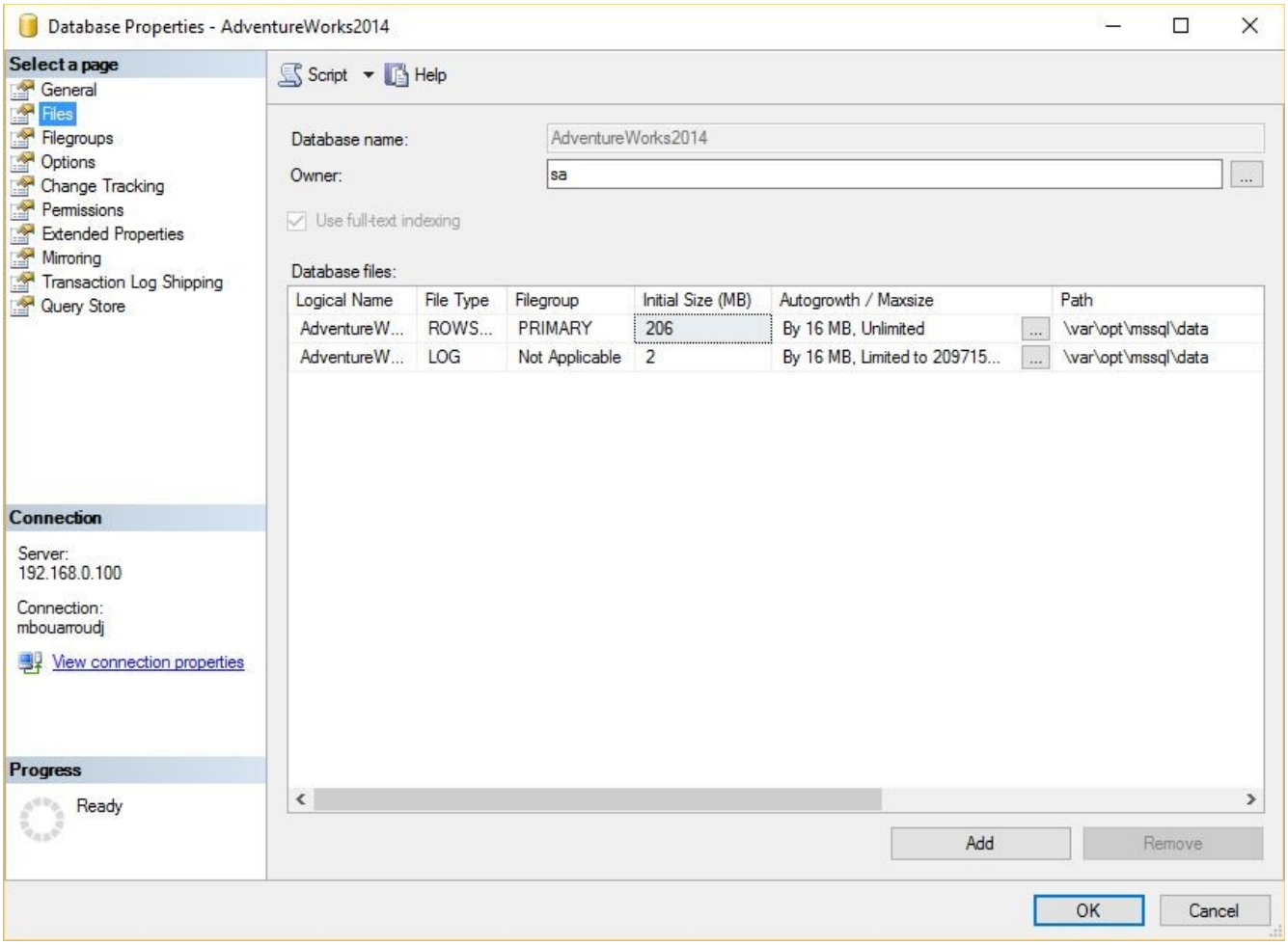
Based on Microsoft documentation here is the supported client tools

Tool	**Minimum version**
SQL Server Management Studio (SSMS) for Windows - Release Candidate 1	17.0
SQL Server Data Tools for Visual Studio - Release Candidate 1	17.0
Visual Studio Code with the mssql extension	Latest (0.2)

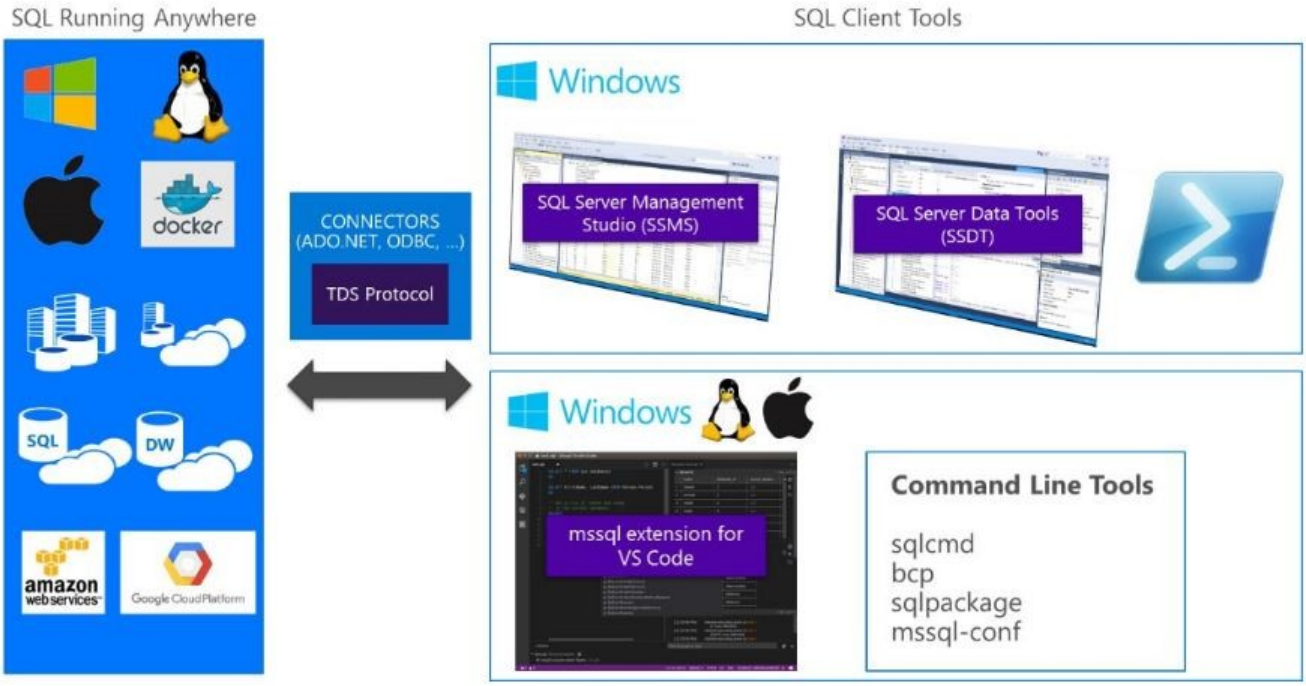
I was able to connect to SQL Server vNext with SSMS V16.5 (build 13.0.1600) and V17.0 RC1 (build 14.0.1600) without any issue.

You can connect to SQL Server vNext from SSMS as the same way you connect to SQL Server on-premises or on the cloud, you only need to enter the IP Adr and SQL login




Database Properties - AdventureWorks2014
Select a page
General
Files
Filegroups
Options
Change Tracking
Permissions
Extended Properties
Mirroring
Transaction Log Shipping
Query Store
Script
Help
Database name:
AdventureWorks2014
Owner:
sa
Use full-text indexing
Database files:
Logical Name
File Type
Filegroup
Initial Size (MB)
Autogrowth / Maxsize
Path
AdventureW...
ROWS...
PRIMARY
206
By 16 MB, Unlimited
\var\opt\mssql\data
AdventureW...
LOG
Not Applicable
2
By 16 MB, Limited to 209715...
\var\opt\mssql\data
Add
Remove
Connection
Server:
192.168.0.100
Connection:
mbouarroudj
View connection properties
Progress
Ready
OK
Cancel

The picture below from Microsoft site summarizes the expanded SQL tools portfolio



Conclusion

Here are some Microsoft references I used to write this article

https://www.microsoft.com/en-us/sql-server/developer-get-started
https://www.microsoft.com/en-us/sql-server/sql-server-vnext-including-Linux
https://www.microsoft.com/nett/core
https://code.visualstudio.com

It is recommneded you install and test SQL Server vNext on a virtual machine, do not use a physical or production server for your tests.

Some print screen you have seen in this article may different from your installations or other documentations, the product and client tools are constantly changing, and new builds are released frequently.

If you are wondering how Microsoft have accomplished this tremendous and great work, visit this link:

https://blogs.technet.microsoft.com/dataplatforminsider/2016/12/16/sql-server-on-linux-how-introduction

About me

My name is Mohamed Bouarroudj and I am SQL Server DBA working at IBM Canada, I am certified MCSA - SQL Server 2012/2014, I have started working with SQL Server since 6.5 version and my favourite subjects are High Availability and Performances. I am also the author of SQLDBDiff a freeware and shareware tool that compares the structure and the data of SQL Server databases and generates the sync scripts, link: www.sqldbtools.com

You can contact me on mbouarroudj@sqldbtools.com

Montreal December 2016