

Application Note for MSSQL Database Operations in u-create IoT

Abstract:

This document is about Microsoft SQL databases. It is shown how to install, setup and configure a database server for the Microsoft SQL technology. An important step for controlling the database is user creation and user roles, which will be explained within this document. Furthermore, it is detailed how to access a database manually (by SQL queries) as well as in an automated way using the node-red-contrib-mssql node in Node-RED. Examples are provided to show how to create and delete databases, create and delete tables, and insert values into tables.

Hardware reference

No.	Component name	Article No.	Hardware / Firmware version
1	UC20-WL2000-AC	1334950000	FW: 1.13.0 (or higher)
2	UC20-WL2000-IOT	1334990000	FW: 1.13.0 (or higher)
3	UC20-SL2000-OLAC-EC	2638920000	-
4	UC20-SL2000-OLAC-EC-CAN	2655590000	-

Software reference

No.	Software name	Article No.	Software version
1	u-create studio with the software unit "Node-RED"	2660130000	The latest
2	node-red-contrib-mssql	-	0.7.3
3	Microsoft SQL Server Developer Edition 2019	-	-
4	Microsoft SQL Server Management Studio	-	SSMS 2019 (or higher)

File reference

No.	Name	Description	Version
1	AN0038-UC20-WL2000 MSSQL Database Operations in u-create IoT.zip	The file contains a Node-RED flow, which was used in examples 1 and 2	-

Contact

Weidmüller Interface GmbH & Co. KG
Klingenbergstraße 26
32758 Detmold, Germany
www.weidmueller.com

For any further support please contact your local sales representative:
<https://www.weidmueller.com/countries>

Content

1	Warning and Disclaimer.....	4
2	Requirements	5
3	Install and Configure Microsoft SQL Server 2019 Developer Edition	6
3.1	Download	6
3.2	Installation	6
3.3	Configuration	8
3.3.1.	User Setup (Option 1).....	14
3.3.2.	User Setup (Option 2).....	16
4	Node-RED	18
4.1	Description of the Microsoft SQL node (node-red-contrib-mssql).....	18
4.2	Setup of the Microsoft SQL node (cloud configuration screen)	18
4.3	Simple database communication (Example 1)	19
4.4	Advanced database communication (Example 2)	23
5	Troubleshooting.....	29

1 Warning and Disclaimer

Warning

Controls may fail in unsafe operating conditions, causing uncontrolled operation of the controlled devices. Such hazardous events can result in death and / or serious injury and / or property damage. Therefore, there must be safety equipment provided / electrical safety design or other redundant safety features that are independent from the automation system.

Disclaimer

This Application Note / Quick Start Guide / Example Program does not relieve you of the obligation to handle it safely during use, installation, operation and maintenance. Each user is responsible for the correct operation of his control system. By using this Application Note / Quick Start Guide / Example Program prepared by Weidmüller, you accept that Weidmüller cannot be held liable for any damage to property and / or personal injury that may occur because of the use.

Note

The given descriptions and examples do not represent any customer-specific solutions, they are simply intended to help for typical tasks. The user is responsible for the proper operation of the described products. Application notes / Quick Start Guides / Example Programs are not binding and do not claim to be complete in terms of configuration as well as any contingencies. By using this Application Note / Quick Start Guide / Example Program, you acknowledge that we cannot be held liable for any damages beyond the described liability regime. We reserve the right to make changes to this application note / quick start guide / example at any time without notice. In case of discrepancies between the proposals Application Notes / Quick Start Guides / Program Examples and other Weidmüller publications, like manuals, such contents have always more priority to the examples. We assume no liability for the information contained in this document. Our liability, for whatever legal reason, for damages caused using the examples, instructions, programs, project planning and performance data, etc. described in this Application Note / Quick Start Guide / Example is excluded.

Security notes

In order to protect equipment, systems, machines and networks against cyber threats, it is necessary to implement (and maintain) a complete state-of-the-art industrial security concept. The customer is responsible for preventing unauthorized access to his equipment, systems, machines and networks. Systems, machines and components should only be connected to the corporate network or the Internet if necessary and appropriate safeguards (such as firewalls and network segmentation) have been taken.

2 Requirements

To understand the document, basic knowledge about Node-RED, SQL and JavaScript is required. If a Microsoft SQL Server application is available, including all information (credentials, server address, database name, ...), all relevant chapters about installation and configuration of Microsoft SQL Server Tools can be skipped.

This application requires additional hardware, such as u-control web / IoT or u-control studio with the Software Unit "Node-RED", PC or Laptop with administrative rights (as database server), and the software, such as installed node-red-contrib-mssql node (for u-control web / IoT). Microsoft SQL Server Management Tools and Microsoft SQL Server Developer Edition 2019.

3 Install and Configure Microsoft SQL Server 2019 Developer Edition

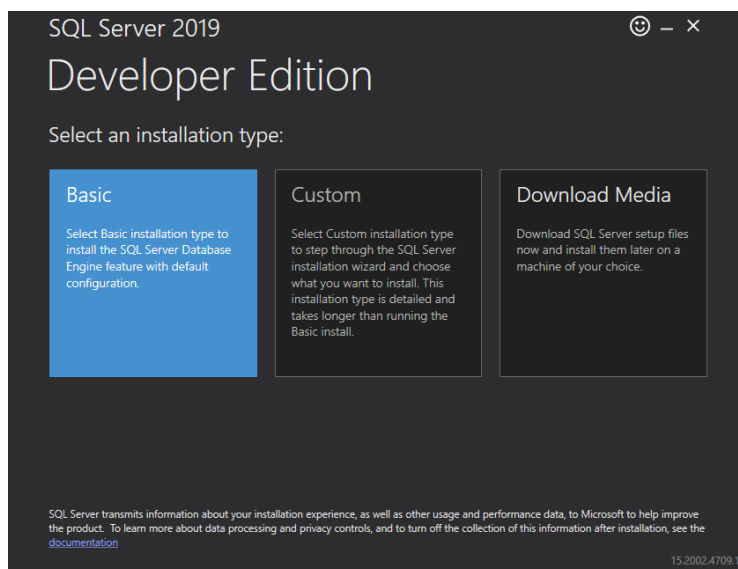
3.1 Download

Microsoft SQL Server Developer Edition is available at the [official Microsoft webpage](#).

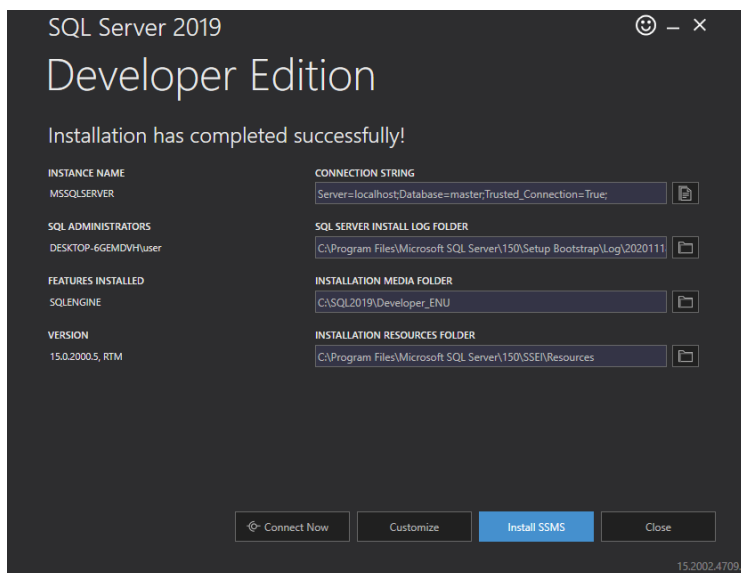
3.2 Installation

To install Microsoft SQL Server Developer Edition, proceed with the following steps:

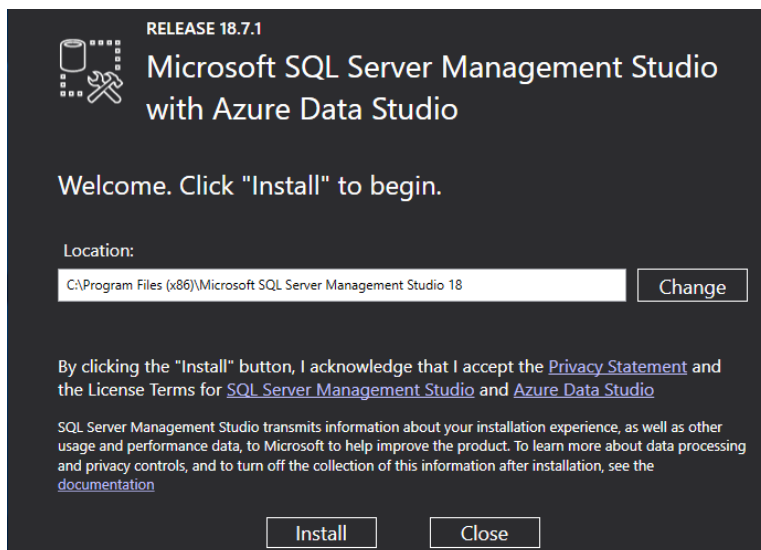
1. Start the Installer
2. Choose **Basic** installation type



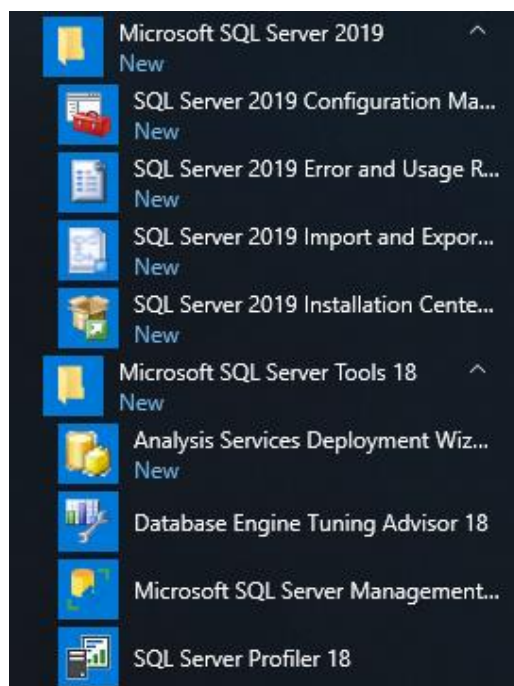
3. Define an installation path and click **Install**
4. Once the installation has been completed, the following window appears. Click **Install SSMS**



5. Install Microsoft SQL Server Management Studio (alternatively it can be downloaded [here](#))
 - Define an installation path



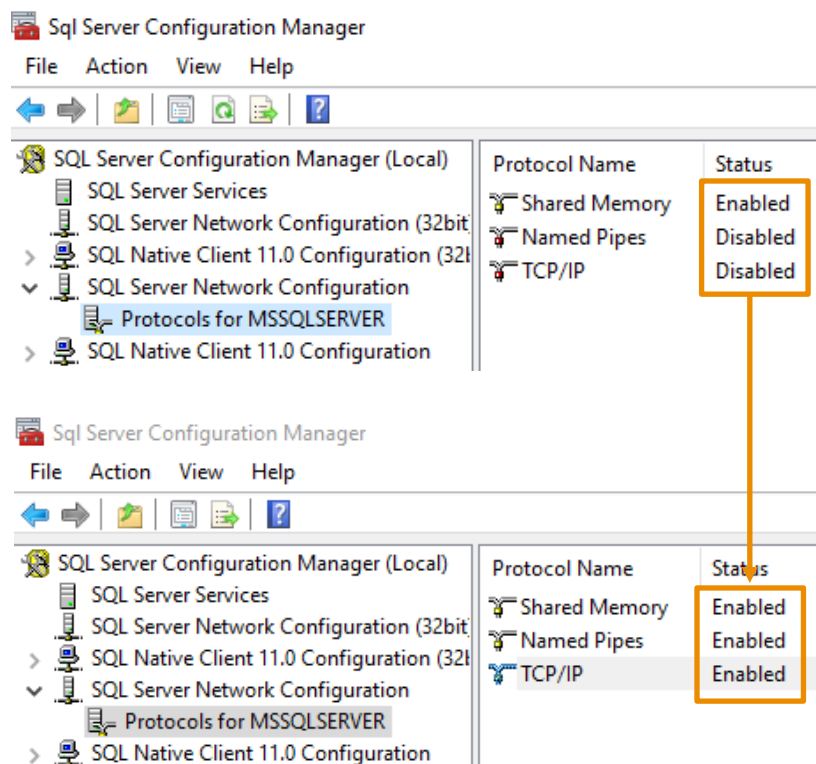
- Click **Install** and wait for the software to be installed.
- After successful installation the following application icons will appear in Start menu



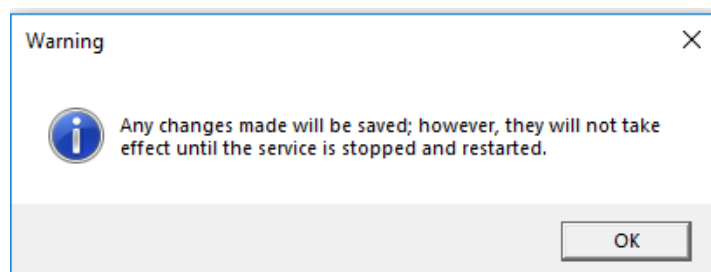
3.3 Configuration

To complete the configuration, proceed with the following steps.

- ▶ Start SQLServer 2019 Configuration Manager
- ▶ Go to SQL Server Network Configuration
- ▶ Enable all the protocols by Right click – **Enable**.

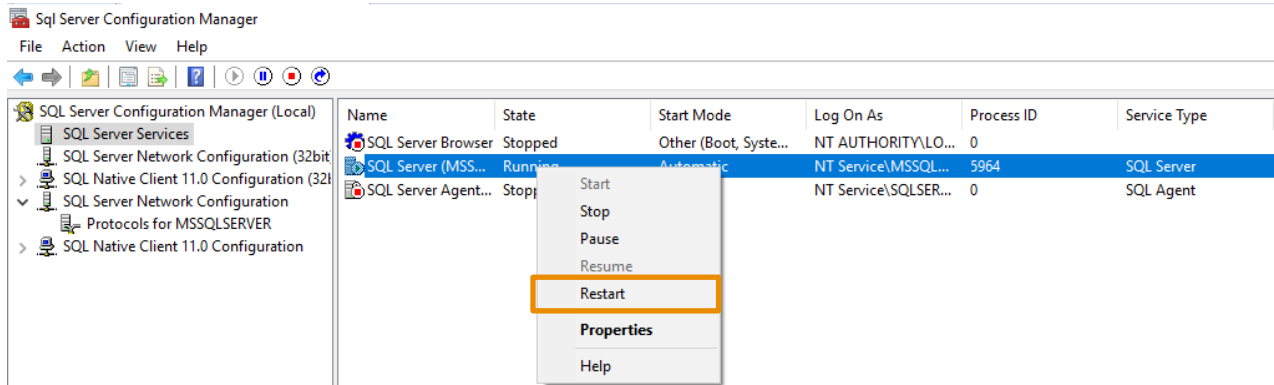


- ▶ The following warning will appear on the screen

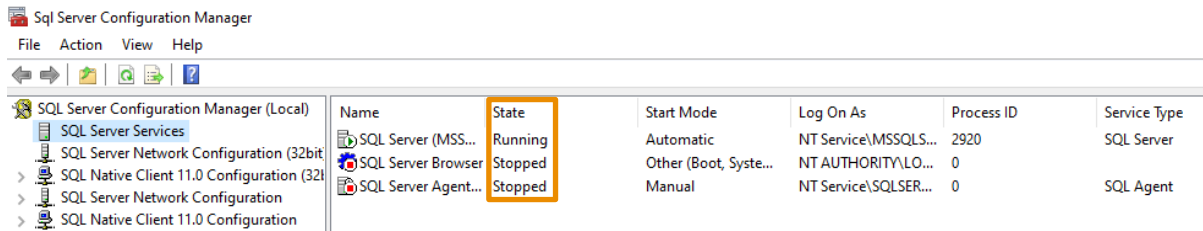


Application Note for MSSQL Database Operations in u-create IoT

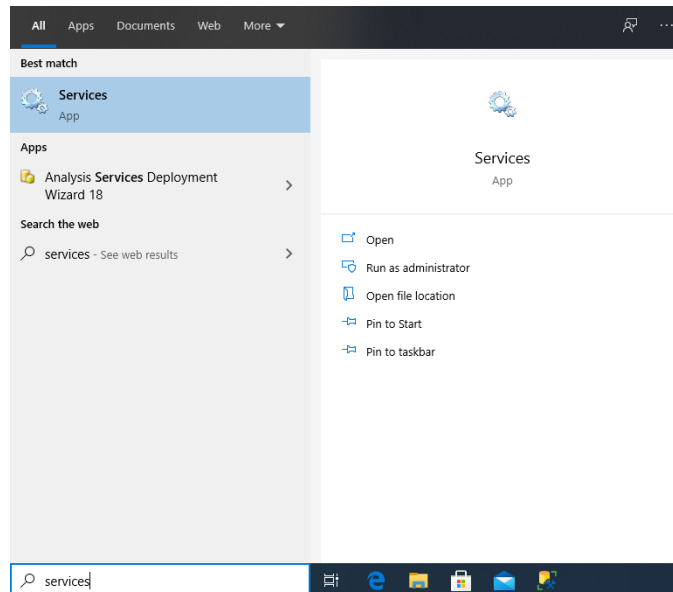
- In order to restart the server, go to **SQL Server Services**, Right-click **SQL Server** and click **Restart**






- If SQL Server Browser or Agent are **Stopped** as well, open **Services**



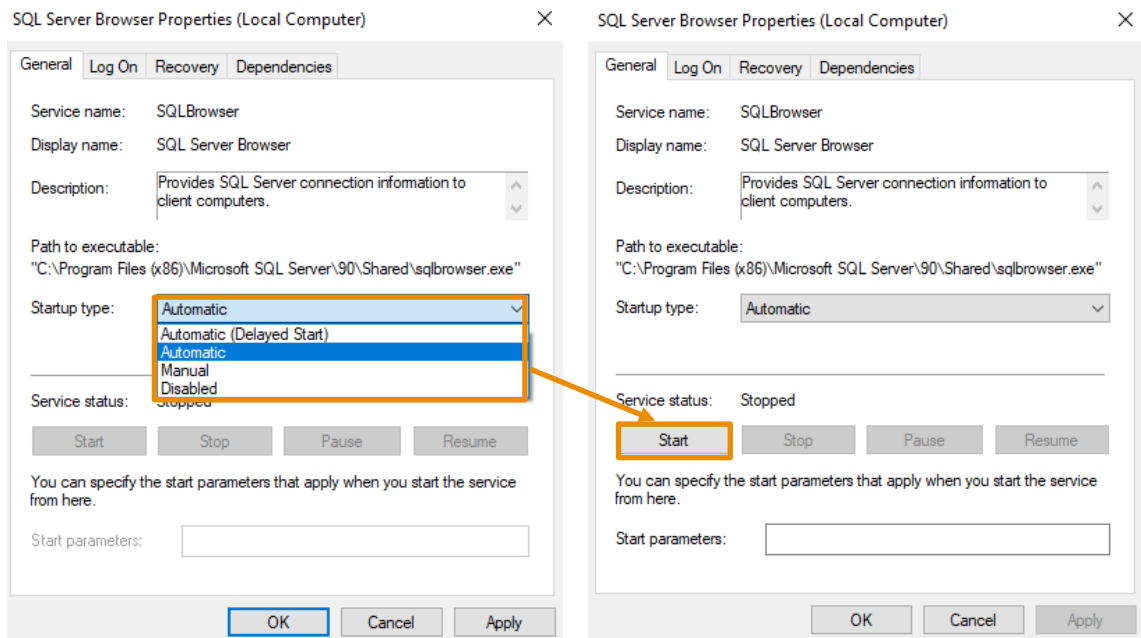
- Search for **Services** in start menu and press **ENTER**






- Search for **SQL Server**, **SQL Server Agent** and **SQL Server Browser**

	SQL Server (MSSQLSERVER)	Provides sto...	Running	Automatic	NT Service...
	SQL Server Agent (MSSQLSERVER)	Executes jo...		Manual	NT Service...
	SQL Server Browser	Provides SQ...		Disabled	Local Service

- Right click on **SQL Server Agent** and go to **Properties**
- Change **Startup type** to **Automatic**. And then **Apply**. Click **Start** to start the service
- Proceed with the **same steps** for **SQL Server Bowser**

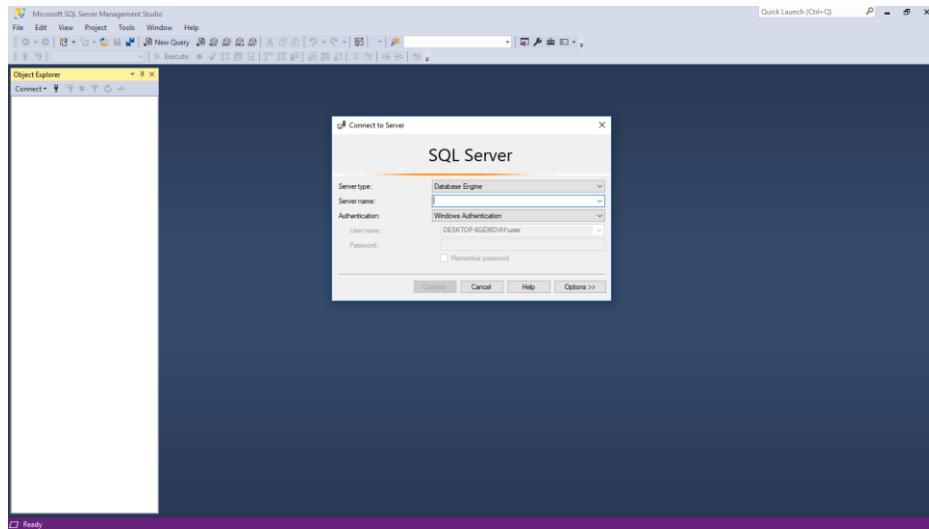


- All the services should now run

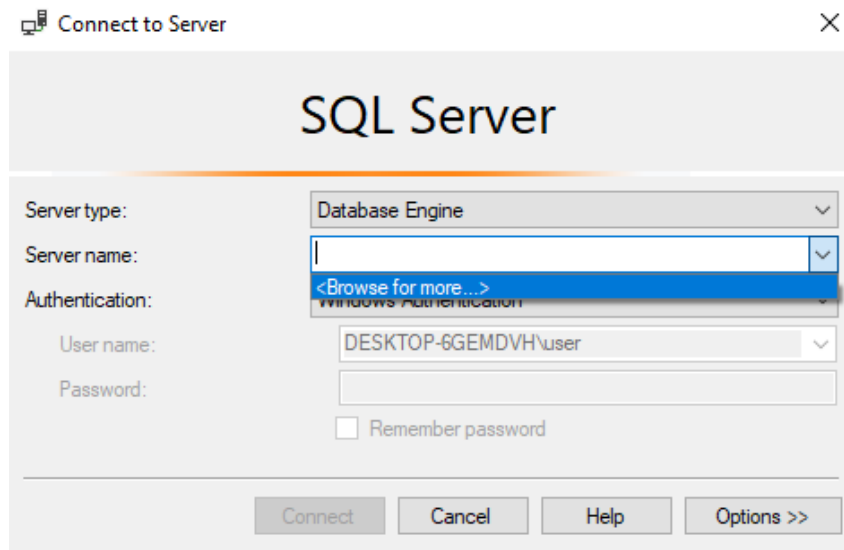
	SQL Server (MSSQLSERVER)	Provides sto...	Running	Automatic	NT Service...
	SQL Server Agent (MSSQLSERVER)	Executes jo...	Running	Automatic	NT Service...
	SQL Server Browser	Provides SQ...	Running	Automatic	Local Service

- Close **SQLServer 2019 Configuration Manager**

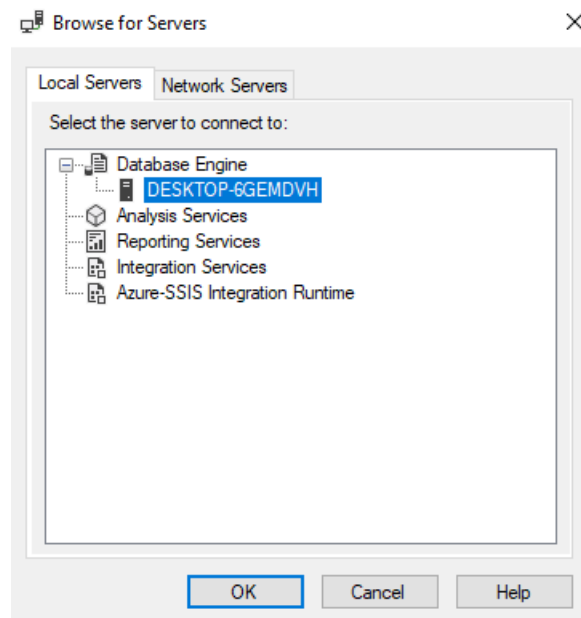
► Start **Microsoft SQL Server Management Studio**



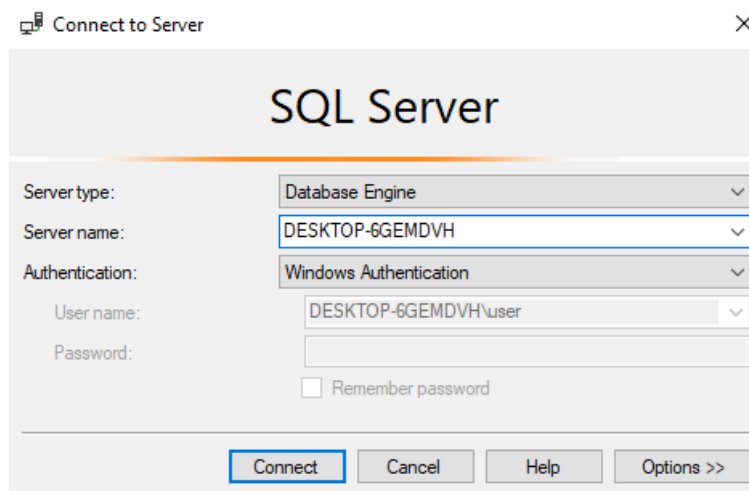
► Find your local server instance. Click on **<Browse for more...>**



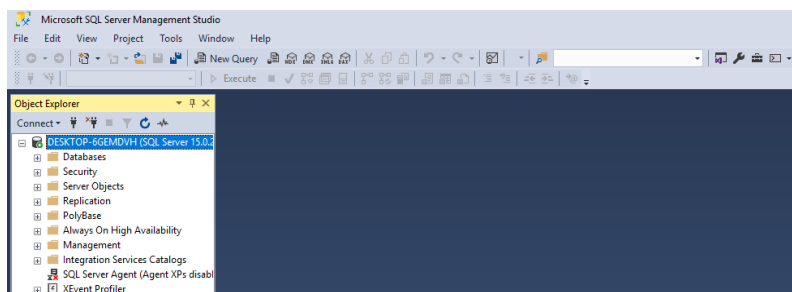
- Select the local **Database Engine** and press **OK**.



- In the following window click **Connect**

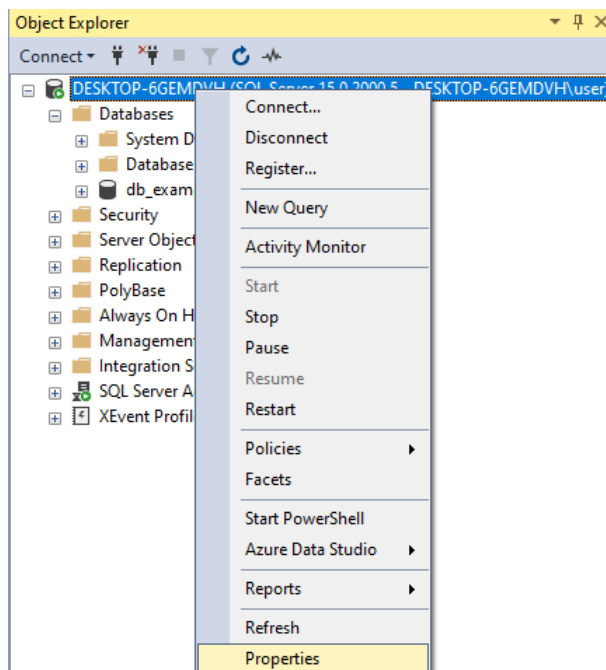


- The starting page should appear on the screen

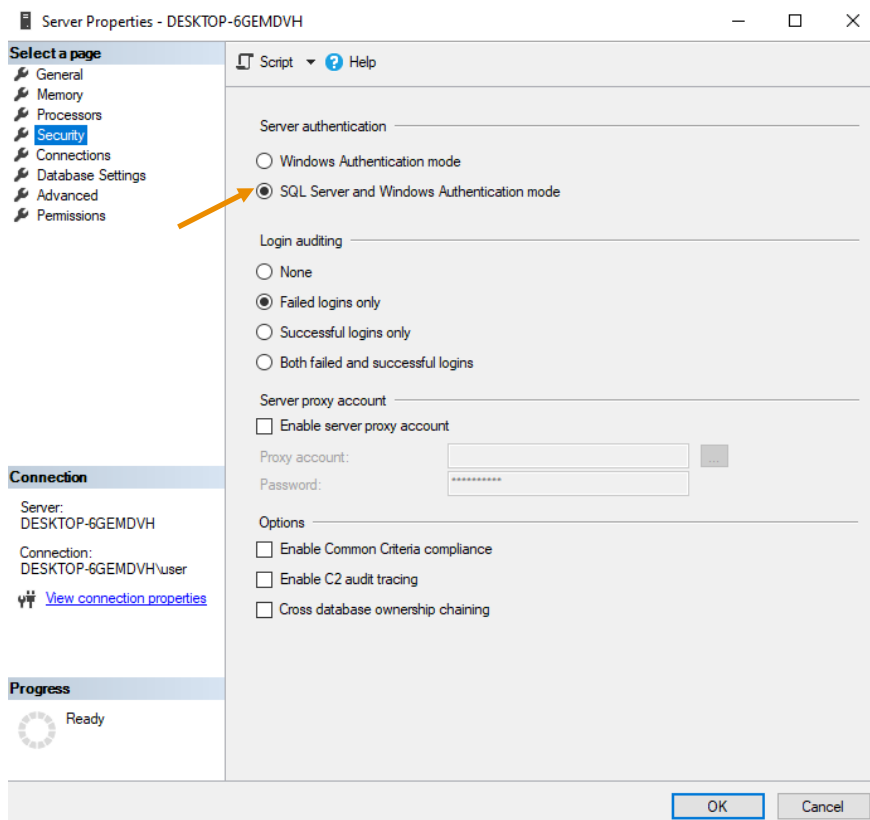


Application Note for MSSQL Database Operations in u-create IoT

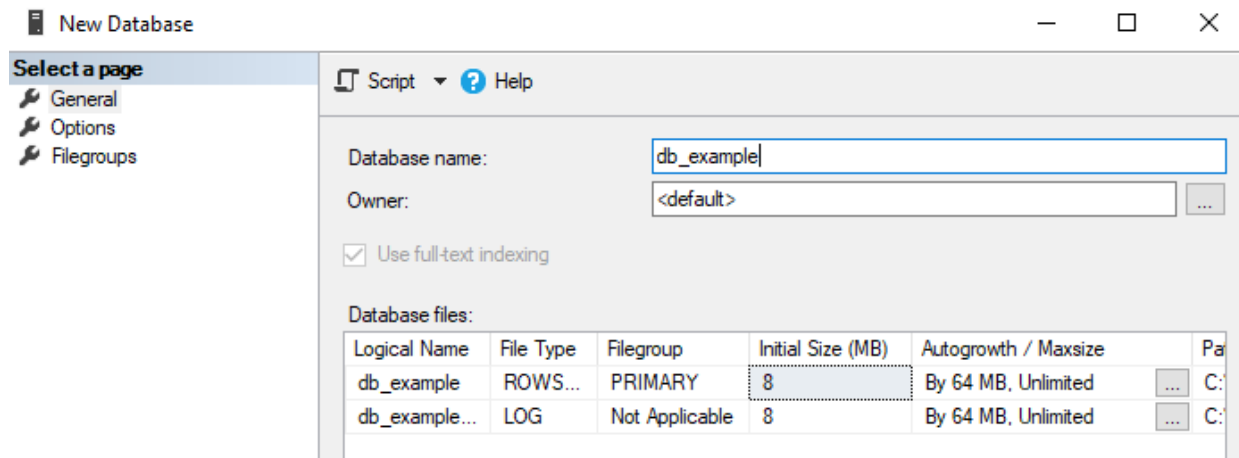
- ▶ Right click on your server instance, here: “DESKTOP-6GEMVDH (SQL Server...)” and go to **Properties**



- ▶ Go to Security and select SQL Server and Windows Authentication mode
- ▶ Press **OK**

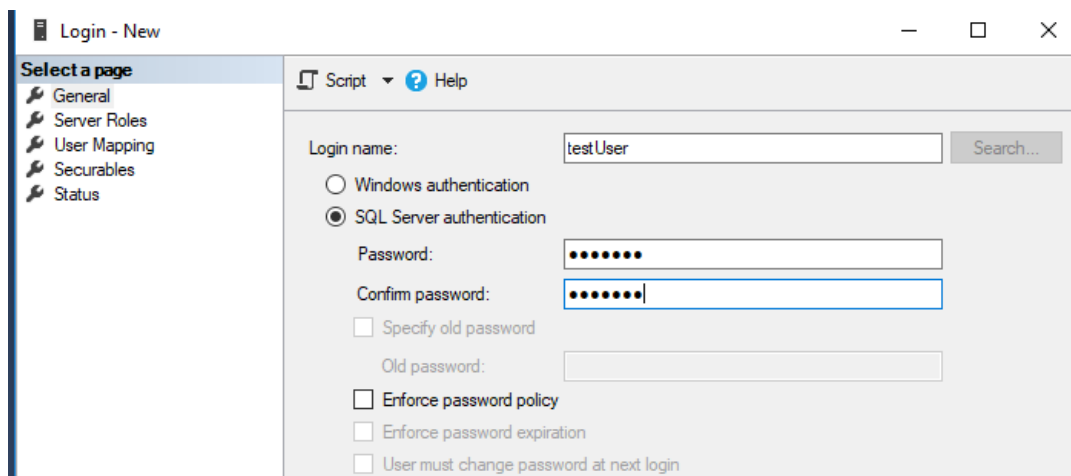


- ▶ Right click on **Database** → **New Database...**
- ▶ Define a database name and press **OK**

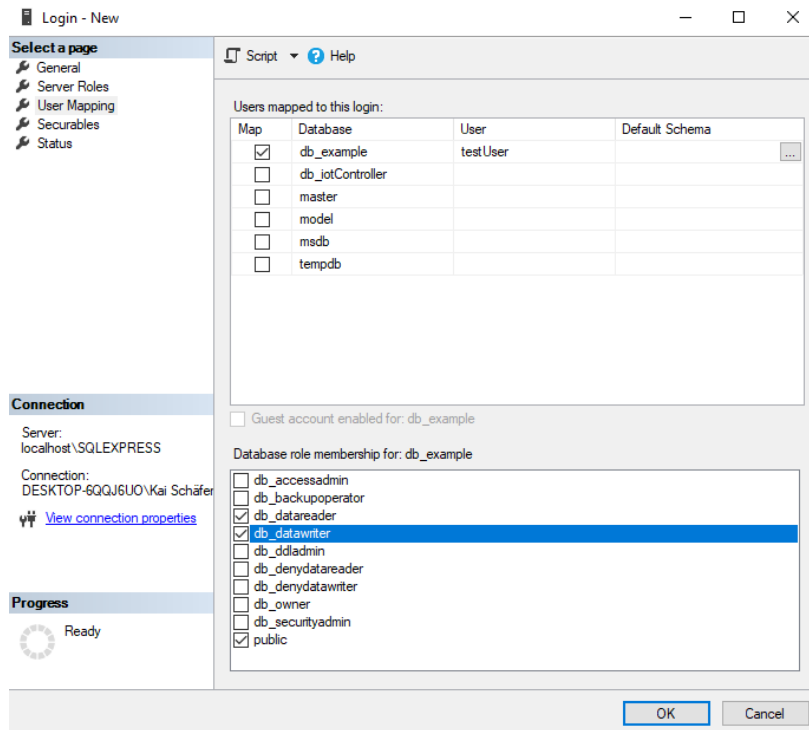


3.3.1. User Setup (Option 1)

- ▶ Right click on **Security** → **New Login...** and enter Login name and Password



- Go to **User Mapping** and map your **User** to your **Database**.
- Assign roles to the **User** → db_datareader, db_datawriter, public and press **OK**




- Right click on **Databases / db_example / Tables** → **Table...**
- Create some columns with corresponding data types and save the table

DESKTOP-6GEMDVH.d...e - dbo.testTable* ➡ ✕			
Column Name	Data Type	Allow Nulls	
id	int	<input type="checkbox"/>	
temperature	float	<input checked="" type="checkbox"/>	
timestamp	timestamp	<input checked="" type="checkbox"/>	
		<input type="checkbox"/>	



You can also define a table using so called queries, but this procedure may be easier in the first step.

If the new table does not appear immediately, it is necessary to update the **Object Explorer** manually. Select the database with left click and press the update button in the top menu . The table should appear.

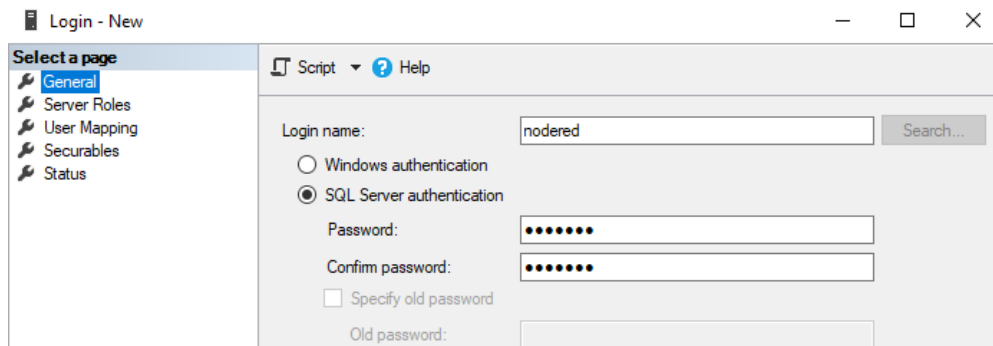
Remember to swap **localhost** with the **MSSQL-Server host IP** if you want to access the database from another device in your network.

3.3.2. User Setup (Option 2)

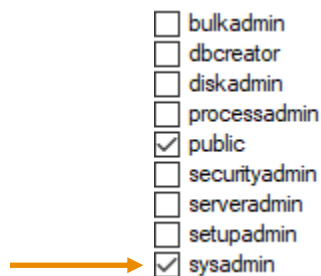
Another opportunity to manage the database server is using Node-RED. It is possible to create one user with the role of sysadmin. This user has full administrative rights and is allowed to create and delete databases with rules, tables and may assign users to it.

A configuration example is shown below.

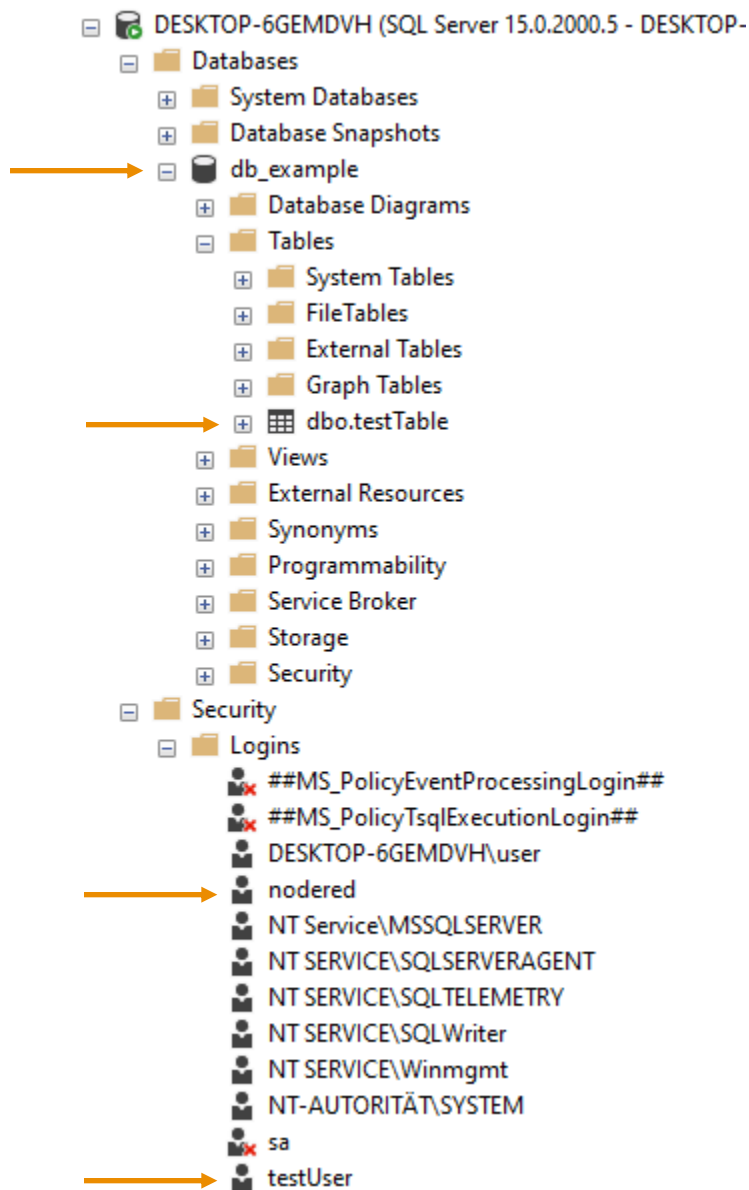
- Right click on **Security** → **"New Login..."** and enter Login name and Password (e.g. "nodered" and "nodered")



- Under **Server Roles** assign **sysadmin** and press **OK**



The project tree should look as follows.



- Continue with the Node-RED implementation.

4 Node-RED

This chapter is about the implementation of an example flow for a database communication in Node-RED. After this chapter it should be possible to create at least tables to an existing database and insert data into it.

Optionally, the user can administrate the server using Node-RED. The necessary queries and statements will be shown within the following chapters.

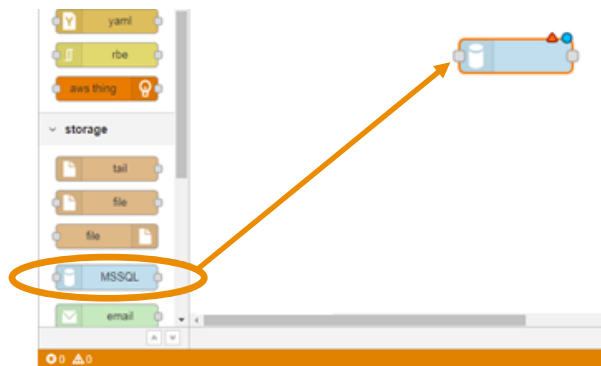
4.1 Description of the Microsoft SQL node (node-red-contrib-mssql)

This node handles a connection to a database. It can send SQL statements and process replies to SQL queries. The configured SQL statement is processed every time an input message arrives. The SQL statements can either be fixed (defined within the node), semi-fixed (defined within the node but with parameters received via incoming messages) or dynamic (incoming message payload contains the SQL statement to be send).

Experienced SQL users know that statements or queries in SQL must end with a semicolon “;”. The MSSQL node interprets the query inside a message object and sets a semicolon if not present.

4.2 Setup of the Microsoft SQL node (cloud configuration screen)

- ▶ Login to u-create IoT web interface and click on **Cloud configuration**.
- ▶ Create a new flow or use an existing flow.
- ▶ Find “MSSQL” in the node palette on the left, use the **Filter nodes** field if needed. Insert the node into the flow as shown on the following figure (drag and drop into worksheet), connect other nodes if needed.



- ▶ Configure **MSSQL** node (described in the next chapters).

4.3 Simple database communication (Example 1)

This application reads data from a global variable of UC20-W2000-IOT using node **iodata** and writes its value into a given SQL table. If needed, it can also read the complete database table. To receive data, we use the **iodata** node, which we configure to cyclically read a previously defined global variable humidity.

A double click on the node opens the configuration screen, which is shown on the figure below. On the left side main config and SQL query are illustrated, while on the right side – a database connection. Click on the button (1) to generate / edit the database connection. Within the new window (5), enter necessary information for database connection. These are specific to your database, thus need to be provided by the MS SQL database administrator.

Field Name (2) defines the name of the node. With option (3), you can define in which part of the outgoing message the results of SQL queries are to be stored (usually msg.payload). Any SQL statements can be entered in the query field (4). This field may contain:

- Complete valid SQL statement
- SQL statements containing variables obtained from incoming **msg-object**, using so called mustache notation. E.g.: the value of property msg.payload.value can be referenced via {{payload.value}}
- Or nothing. In this case, an incoming msg.payload has to contain a valid SQL statement.

The figure consists of two side-by-side screenshots of the u-create IoT configuration interface.

The left screenshot is titled "Edit MSSQL node". It features a "node properties" section with the following fields:

- Connection:** A dropdown menu with "Add new MSSQL-CN..." and a pencil icon (labeled 1).
- Name:** A text input field with "Name" (labeled 2).
- Query:** A large text area for entering SQL queries (labeled 4).
- Result to:** A dropdown menu with "msg: payload" (labeled 3).

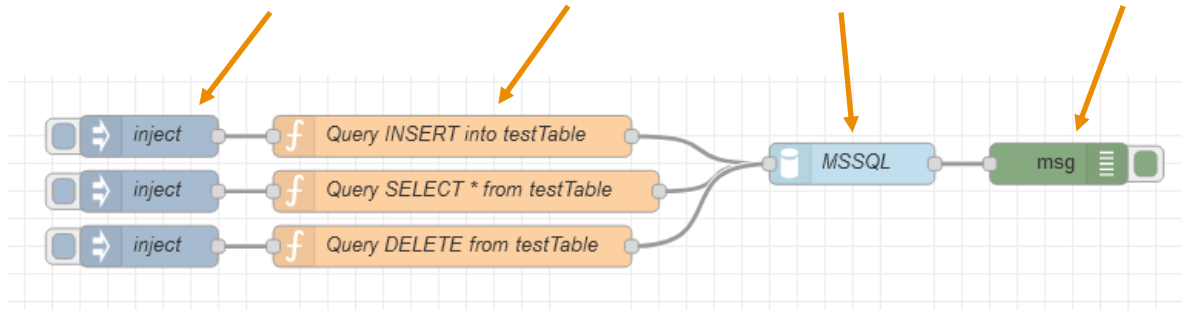
 A tip at the bottom states: "Tip: You can use the [mustache](#) format."

The right screenshot is titled "MSSQL > Add new MSSQL-CN config node". It features a form with the following fields:

- Name:** "ENTER CONNECTION NAME HERE"
- Server:** "ENTER SERVER NAME OR IP HERE"
- Username:** "USER NAME FOR AUTHENTICATION"
- Password:** A password input field.
- Domain:** "ENTER DOMAIN HERE IF NECESSARY"
- Database:** "ENTER DATABASE NAME HERE"
- Use Encryption?** A checkbox that is checked.

 A yellow note at the bottom states: "SQL Databases hosted on Azure will need this checked".

Example 1 consists of **Inject** nodes, **Function** nodes, one **MSSQL** node and one **Debug** node

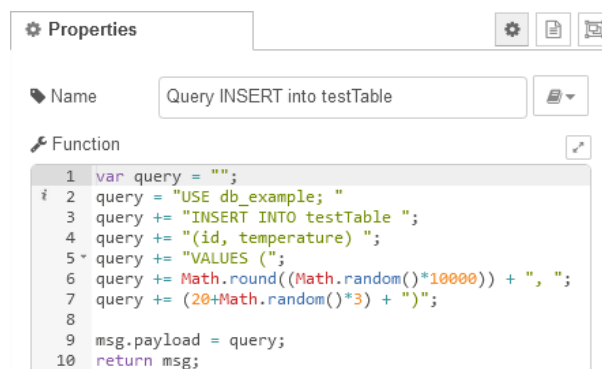


The **Inject** nodes are not configured. They work as “manually triggered”. The three **Function** nodes are holding the SQL queries. They are configured as follows:

- The query within the code below inserts random values into the table

```
"USE db_example;"
```

```
"INSERT INTO testTable (id, temperature) VALUES (value1, value2);"
```

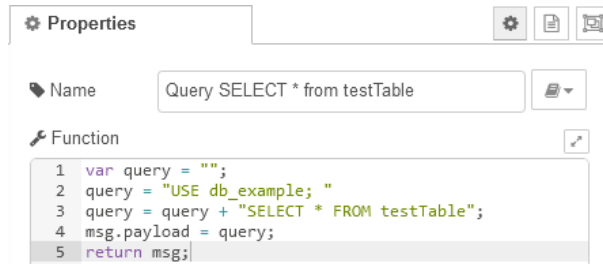


Copy:

```
var query = "";  
query = "USE db_example; "  
query += "INSERT INTO testTable ";  
query += "(id, temperature) ";  
query += "VALUES (";  
query += Math.round((Math.random()*10000)) + ", ";  
query += (20+Math.random()*3) + ")";  
  
msg.payload = query;  
return msg;
```

- The query within the code below reads out all rows of the table

```
"USE db_example;"  
"SELECT * FROM testable;"
```



Copy:

```
var query = "";  
query = "USE db_example; "  
query = query + "SELECT * FROM testTable";  
msg.payload = query;  
return msg;
```

- The query within the code below deletes all rows consisting of values unequal to 0

```
"USE db_example;"  
"DELETE FROM testable WHERE temperature <> 0;"
```



Copy:

```
var query = "";  
query = "USE db_example; "  
query += "DELETE FROM testTable WHERE temperature <> 0;"  
msg.payload = query;  
return msg;
```

The **MSSQL** node is configured as follows:

- Name: here it consists of information about the user and the SQL server instance
- Server: here the IP address of a dedicated virtual machine inside the network
- Username: here the setup which was done in the SSMS
- Password: here the setup which was done in the SSMS
- Use Encryption must be unchecked as long as the DB is not hosted on Azure

Edit MSSQL node > Edit MSSQL-CN node

Delete Cancel Update

Properties

Name testUser@MSSQLSERVER

Server 192.168.0.175

Username testUser

Password

Domain

Database

Use Encryption? ☐

SQL Databases hosted on Azure will need this checked

An output of the **Debug** node after executing the second **inject** node (SELECT) could look like following. It depends on if you have already inserted values into the Table (first **inject** node)

```
25/11/2020, 12:33:56 node: c28bb49f.e3f248
msg : Object
object
  _msgid: "365d586f.31aaf8"
  topic: ""
  payload: array[5]
    0: object
      id: 1620
      temperature: 20.896333105215387
      timestamp: buffer[8]
        0: 0x0
        1: 0x0
        2: 0x0
        3: 0x0
        4: 0x0
        5: 0x0
        6: 0x7
        7: 0xd8
    1: object
    2: object
    3: object
    4: object
```

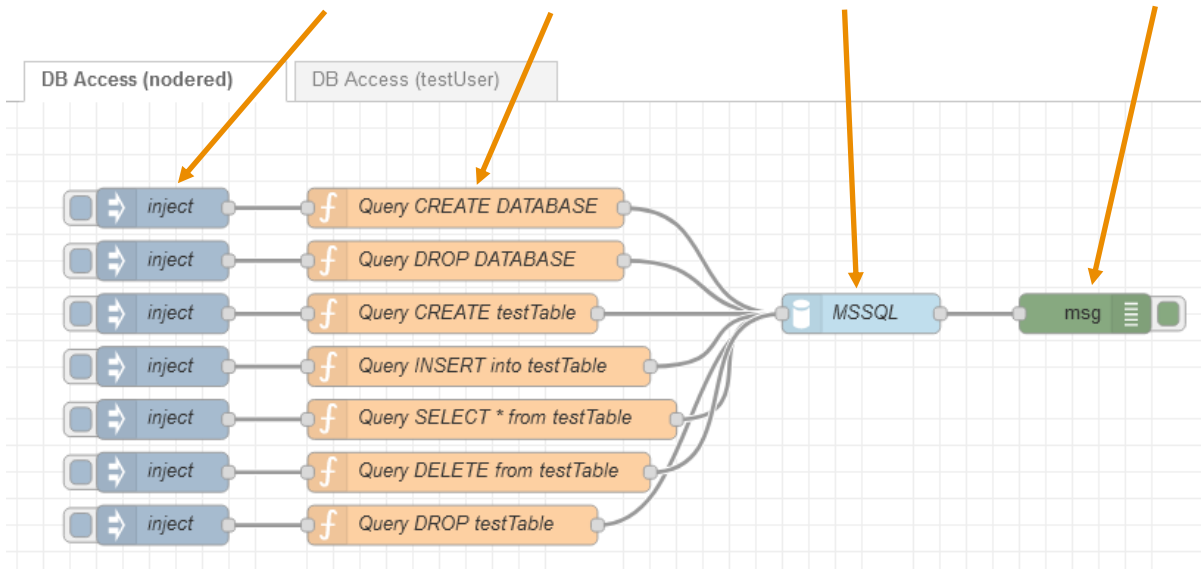
4.4 Advanced database communication (Example 2)

Based on the explanations in Example 1 about the Node-RED environment, the user should now be able to add the required nodes inside a flow and configure the **MSSQL** node according to the specified parameters mentioned below.

The illustrated flow shows the most common operations done on or with a database.

- Create DB
- Drop DB (=Delete)
- Create Table
- Drop Table (=Delete)
- Insert Values (=Write)
- Select Values (=Read)

Example 2 consists of **Inject** nodes, **Function** nodes, one **MSSQL** node and one **Debug** node



The **Inject** nodes are not configured. They work as “manually triggered”. The **Function** nodes are holding the SQL queries. They are configured as follows.

- The query below creates a new database called nodered (if not existing yet).



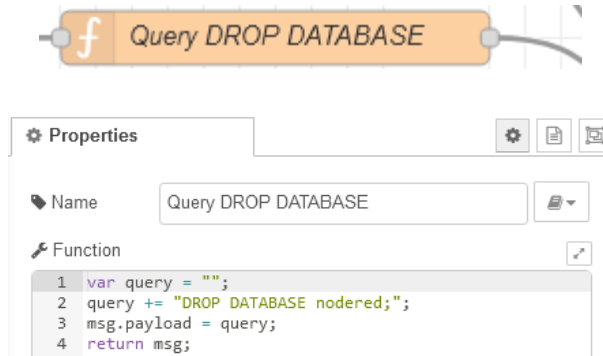
```

Properties
Name: Query CREATE DATABASE
Function:
1 var query = "";
2 query = "CREATE DATABASE nodered;";
3 msg.payload = query;
4 return msg;
    
```

Copy:

```
var query = "";
query = "CREATE DATABASE nodered;";
msg.payload = query;
return msg;
```

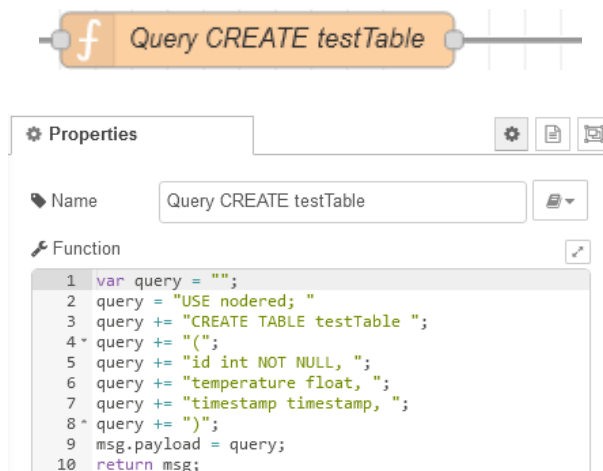
- The query below deletes an existing database named nodered.



Copy:

```
var query = "";
query += "DROP DATABASE nodered;";
msg.payload = query;
return msg;
```

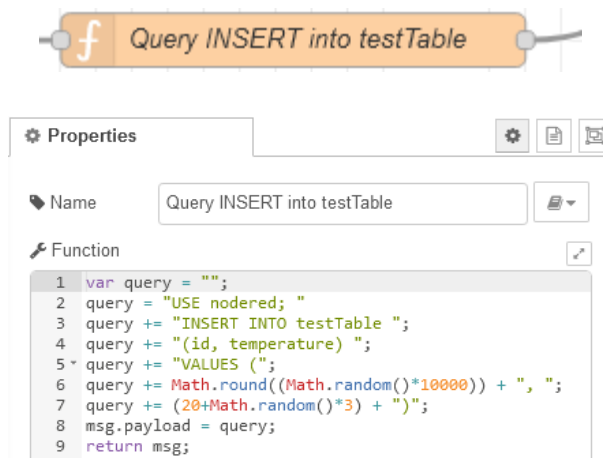
- The query below creates a new table called "testTable" including column names and types



Copy:

```
var query = "";
query = "USE nodered; "
query += "CREATE TABLE testTable ";
query += "(";
query += "id int NOT NULL, ";
query += "temperature float, ";
query += "timestamp timestamp";
query += ");";
msg.payload = query;
return msg;
```

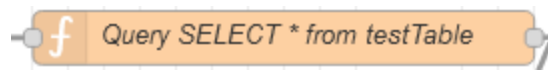
- The query below inserts values into a table called testTable



Copy:

```
var query = "";
query = "USE nodered; "
query += "INSERT INTO testTable ";
query += "(id, temperature) ";
query += "VALUES (";
query += Math.round((Math.random()*10000)) + ", ";
query += (20+Math.random()*3) + ");";
msg.payload = query;
return msg;
```

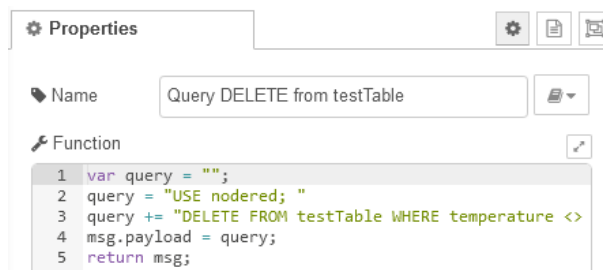
- The query below reads out all entries inside table testTable



Copy:

```
var query = "";
query = "USE nodered; "
query += "SELECT * FROM testTable;";
msg.payload = query;
return msg;
```

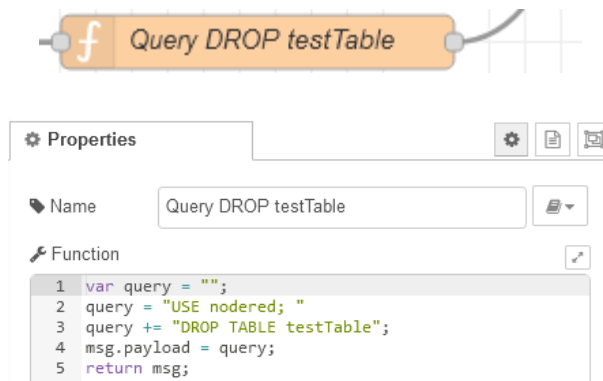
- The query below deletes all entries in testtable, which hold a temperature value unequal to 0



Copy:

```
var query = "";
query = "USE nodered; "
query += "DELETE FROM testTable WHERE temperature <> 0;";
msg.payload = query;
return msg;
```

- The query below deletes testTable completely.



Copy:

```
var query = "";  
query = "USE nodered; "  
query += "DROP TABLE testTable";  
msg.payload = query;  
return msg;
```

The **MSSQL** node is configured the way shown below:



The Query field stays empty. The requesting query comes from outside. The inject node generates a message with timestamp as payload information. msg.payload becomes overwritten while passing the function nodes like Query CREATE DATABASE.

Let's configure the database connection.

Properties

Name: nodered@MSSQLSERVER

Server: 192.168.0.175

Username: nodered

Password:

Domain:

Database:

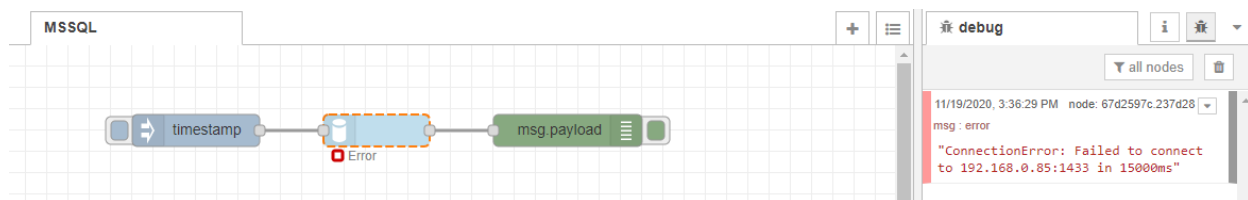
Use Encryption? ☐

SQL Databases hosted on Azure will need this checked

- Name: A describing name for your database connection
- Server: IP address of your database server (here a virtual machine)
- Username: nodered.
- Password: your password for user nodered.

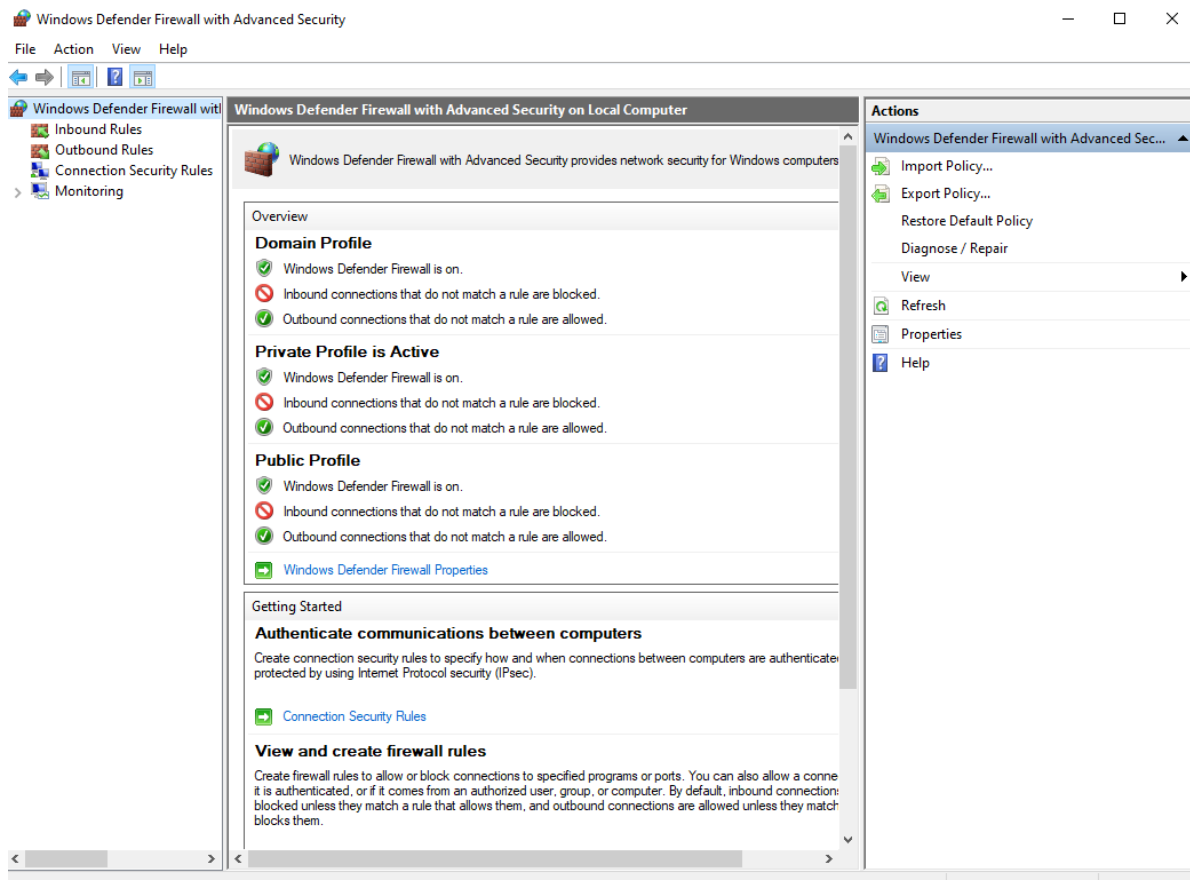
5 Troubleshooting

Problem: You receive an error message once you try to send a query: “ConnectionError: Failed to connect to 192.168.0.85:1433 in 15000ms”



Solution: If the **MSSQL** node was configured correctly and the previous configuration steps for the SSMS were respected, it might be a firewall conflict. Create an inbound rule for TCP port 1433

- Run **Windows Defender Firewall** and click **Advanced settings** on the left pane.



Application Note for MSSQL Database Operations in u-create IoT

- ▶ Click on **Inbound Rules**
- ▶ On the right side click on **New Rule...**
- ▶ Select **Port**
- ▶ Click **Next**

The screenshot shows the 'New Inbound Rule Wizard' dialog box. The title bar says 'New Inbound Rule Wizard'. The main heading is 'Rule Type'. Below it, it says 'Select the type of firewall rule to create.' On the left, there is a 'Steps:' list with 'Rule Type' selected. The main area asks 'What type of rule would you like to create?' with three radio button options: 'Program' (Rule that controls connections for a program.), 'Port' (selected, Rule that controls connections for a TCP or UDP port.), and 'Predefined:' (Rule that controls connections for a Windows experience.). Under 'Predefined:', there is a dropdown menu showing '@FirewallAPI.dll, 80200'. At the bottom, there are buttons for '< Back', 'Next >', and 'Cancel'.

- ▶ In the next window select **TCP**
- ▶ Enter **1433** into the **Specific local ports** field and click **Next**

The screenshot shows the 'New Inbound Rule Wizard' dialog box, Step 2: Protocol and Ports. The title bar says 'New Inbound Rule Wizard'. The main heading is 'Protocol and Ports'. Below it, it says 'Specify the protocols and ports to which this rule applies.' On the left, the 'Steps:' list has 'Protocol and Ports' selected. The main area asks 'Does this rule apply to TCP or UDP?' with two radio button options: 'TCP' (selected) and 'UDP'. Below that, it asks 'Does this rule apply to all local ports or specific local ports?' with two radio button options: 'All local ports' and 'Specific local ports:' (selected). Under 'Specific local ports:', there is a text box containing '1433' and an example text 'Example: 80, 443, 5000-5010'. At the bottom, there are buttons for '< Back', 'Next >', and 'Cancel'.

- ▶ Select **Allow the connection** and click **Next**
- ▶ Check **Domain**, **Private** and **Public** and proceed with **Next**
- ▶ Enter a name like **MSSQL TCP Port 1433 IN** and then click **Finish**