https://www.halvorsen.blog

Python and Microsoft Azure Databases

Hans-Petter Halvorsen

Contents

- SQL Server
- Python and SQL server
- Datalogging Example Saving Data to Local SQL Server Database
- Microsoft Azure
- Databases in Microsoft Azure
- Datalogging Example Saving Data to Azure Database

SQL Server

Table of Contents

Database Systems

- Oracle
- MySQL
- MariaDB
- Sybase
- Microsoft Access
- Microsoft SQL Server
- … (we have hundreds different Database Systems)

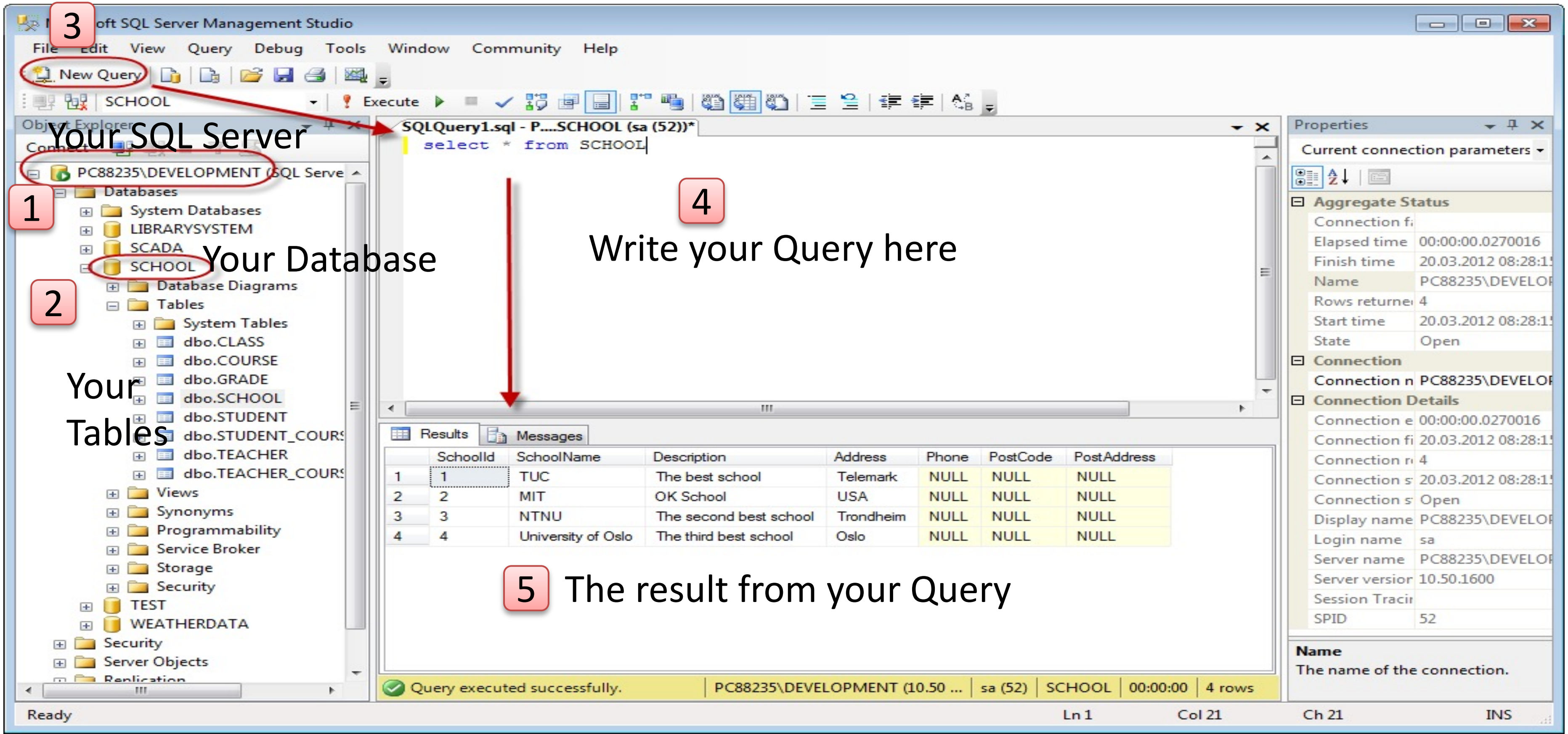
SQL Server

- SQL Server consists of a Database Engine and a Management Studio.
- The Database Engine has no graphical interface - it is just a service running in the background of your computer (preferable on the server).
- The Management Studio is graphical tool for configuring and viewing the information in the database. It can be installed on the server or on the client (or both).

SQL Server

- SQL Server Express
  - Free version of SQL Server that has all we need for the exercises in this Tutorial
- SQL Server Express consist of 2 parts (separate installation packages):
  - SQL Server Express
  - SQL Server Management Studio (SSMS) – This software can be used to create Databases, create Tables, Insert/Retrieve or Modify Data, etc.
- SQL Server Express Installation: https://youtu.be/hhhggAlUYo8

SQL Server Management Studio



1 Your SQL Server

2 Your Database

Your Tables

3

4 Write your Query here

```
select * from SCHOOL
```

5 The result from your Query

	SchoolId	SchoolName	Description	Address	Phone	PostCode	PostAddress
1	1	TUC	The best school	Telemark	NULL	NULL	NULL
2	2	MIT	OK School	USA	NULL	NULL	NULL
3	3	NTNU	The second best school	Trondheim	NULL	NULL	NULL
4	4	University of Oslo	The third best school	Oslo	NULL	NULL	NULL

Query executed successfully. | PC88235\DEVELOPMENT (10.50 ... | sa (52) | SCHOOL | 00:00:00 | 4 rows

Current connection parameters

Aggregate Status	
Connection fi	
Elapsed time	00:00:00.0270016
Finish time	20.03.2012 08:28:1
Name	PC88235\DEVELOI
Rows returne	4
Start time	20.03.2012 08:28:1
State	Open
Connection	
Connection n	PC88235\DEVELOI
Connection Details	
Connection e	00:00:00.0270016
Connection fi	20.03.2012 08:28:1
Connection r	4
Connection s	20.03.2012 08:28:1
Connection s	Open
Display name	PC88235\DEVELOI
Login name	sa
Server name	PC88235\DEVELOI
Server versior	10.50.1600
Session Tracir	
SPID	52

Name
The name of the connection.

https://www.halvorsen.blog

Python and SQL Server

Hans-Petter Halvorsen

Table of Contents

Python

- Python is a fairly old Programming Language (1991) compared to many other Programming Languages like C# (2000), Swift (2014), Java (1995), PHP (1995).
- Python has during the last 10 years become more and more popular.
- Today, Python has become one of the most popular Programming Languages.

Software used in this Tutorial:

- Anaconda Distribution (Python + most used Libraries/Packages are included)
- Spyder Python editor (included with Anaconda Distribution)

Python Drivers for SQL Server

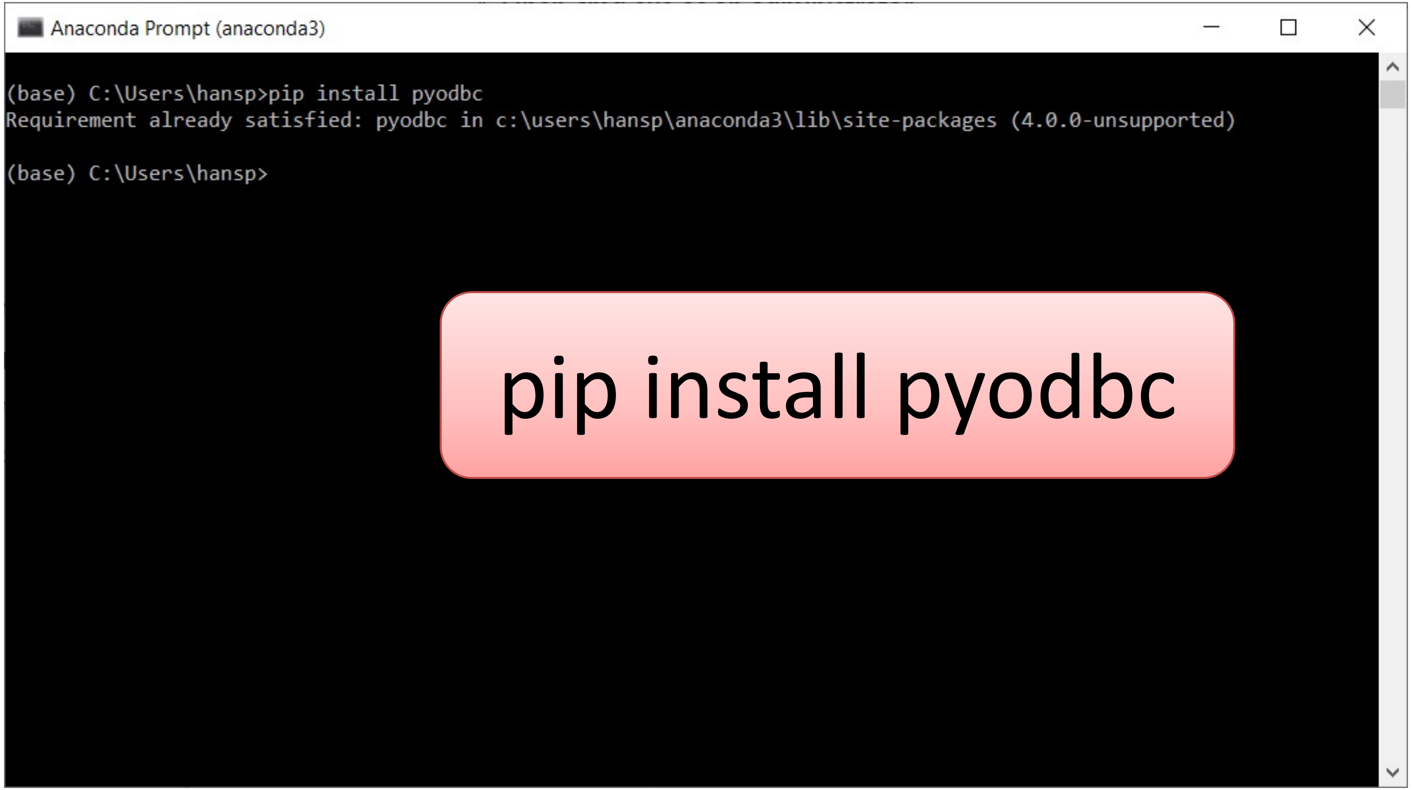
- There are several python SQL drivers available:
  - pyodbc
  - pymssql
- These Drivers are not made made Microsoft but the Python Community.
- However, Microsoft places its testing efforts and its confidence in pyodbc driver.
- Microsoft contributes to the pyODBC open-source community and is an active participant in the repository at GitHub

https://docs.microsoft.com/sql/connect/python/python-driver-for-sql-server

pyodbc

- pyodbc is an open-source Python module that can access ODBC databases, e.g., SQL Server
- https://pypi.org/project/pyodbc/
- Installation: pip install pyodbc

pyodbc



```
(base) C:\Users\hansp>pip install pyodbc
Requirement already satisfied: pyodbc in c:\users\hansp\anaconda3\lib\site-packages (4.0.0-unsupported)

(base) C:\Users\hansp>
```

pip install pyodbc

Connect to Database from Python

The newest and
recommend driver

```
import pyodbc

driver = "{ODBC Driver 17 for SQL Server}"
server = "xxxxxx"
database = "xxxxx"
username = "xxxxx"
password = "xxxxxx"
conn = pyodbc.connect("DRIVER=" + driver
                      + ";SERVER=" + server
                      + ";DATABASE=" + database
                      + ";UID=" + username
                      + ";PWD=" + password )
```

Connect to Database from Python

Example:

Server Name

If Server is on your local PC, you can use LOCALHOST

```
import pyodbc

driver = "{ODBC Driver 17 for SQL Server}"
server = "TESTPC\\SQLEXPRESS"
database = "BOOKSTORE"
username = "sa"
password = "Test123"
conn = pyodbc.connect("DRIVER=" + driver
                      + ";SERVER=" + server
                      + ";DATABASE=" + database
                      + ";UID=" + username
                      + ";PWD=" + password )
```

Instance Name (you can have multiple instances of SQL Server on the same computer)

Here is the built-in “sa” user (System Administrator) used to connect to the Database. In general, you should use another user than the sa user. The sa user is used here for simplicity. You can easily create a new user in SQL Server Management Studio

Datalogging Example Saving Data to Local SQL Server Database

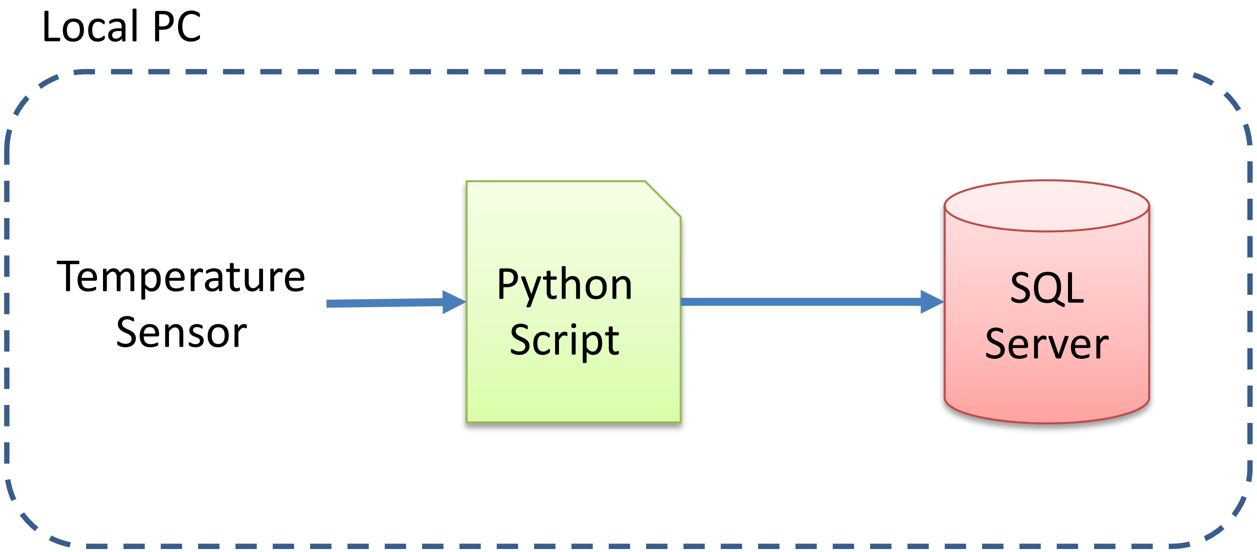
Hans-Petter Halvorsen

Table of Contents

Datalogging Example

- We can log data from a DAQ device or similar
- We start by creating a simple Random Generator that simulates a Temperature Sensor and log these data to the SQL Server database
- Then we will in another script read the data from the database and plot them.

System Overview



SQL Server Database

Let's create a New Database called, e.g., “LOGGINGSYSTEM”

We insert the following Table:

```
CREATE TABLE [MEASUREMENTDATA]
(
        [MeasurmentId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
        [SensorName] [varchar](50) NOT NULL,
        [MeasurementValue] float NOT NULL,
        [MeasurementDateTime] datetime NOT NULL
)
GO
```

Note! This is a very simplified Database to show the basic principles. It does not reflect best practice. Typically, you have multiple tables that are related to each other and more columns like Unit, etc.

Logging Data

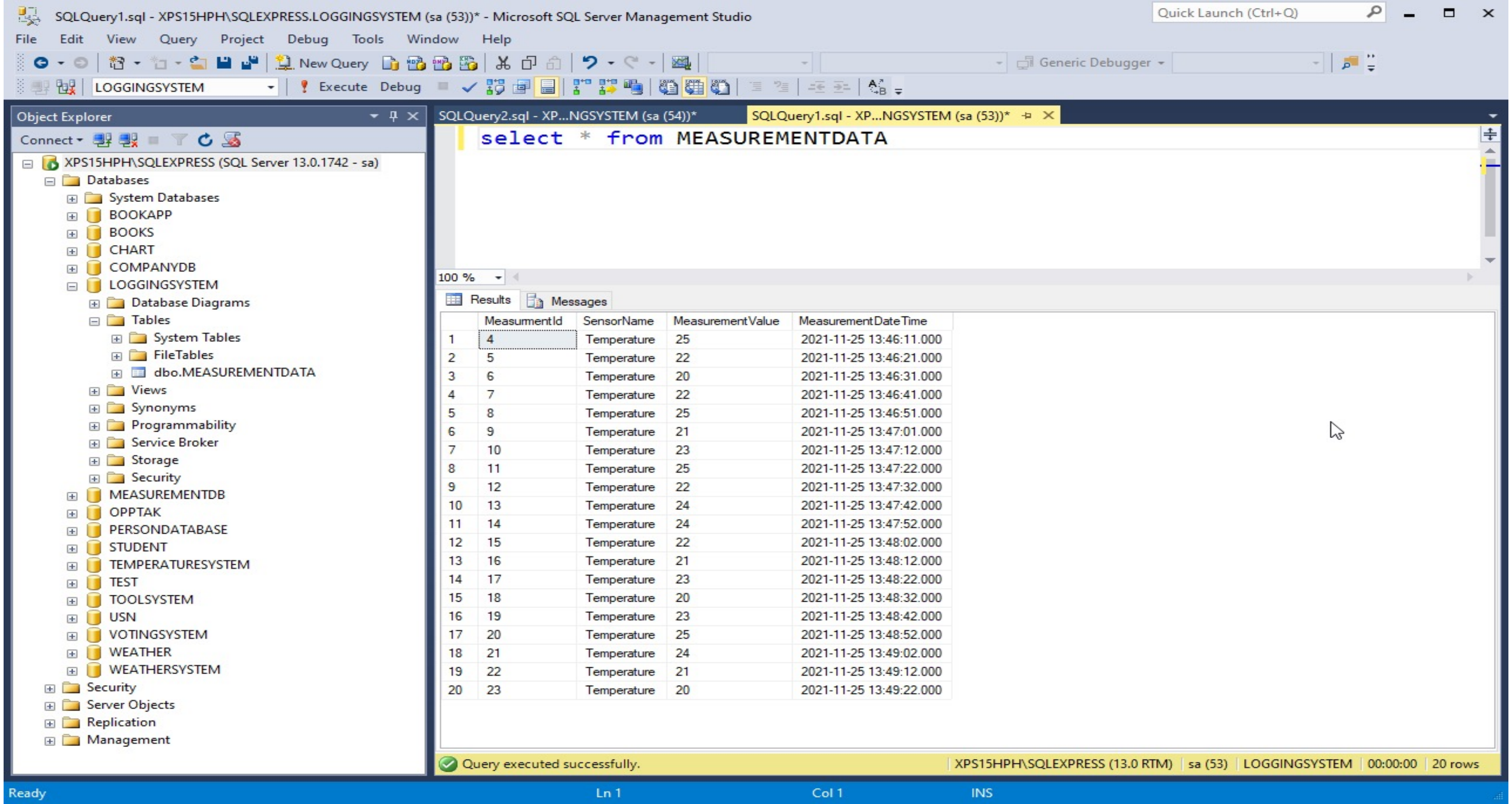
```
import pyodbc
import random
import time
from datetime import datetime
import database

# Connect to Database
connectionString = database.GetConnectionString()
conn = pyodbc.connect(connectionString)
cursor = conn.cursor()
query = "INSERT INTO MEASUREMENTDATA (SensorName, MeasurementValue, MeasurementDateTime) VALUES (?,?,?)"

sensorName = "Temperature"
Ts = 10 # Sampling Time
N = 20
for k in range(N):
    # Generate Random Data
    LowLimit = 20
    UpperLimit = 25
    measurementValue = random.randint(LowLimit, UpperLimit)

    #Find Date and Time
    now = datetime.now()
    datetimeformat = "%Y-%m-%d %H:%M:%S"
    measurementDateTime = now.strftime(datetimeformat)

    # Insert Data into Database
    parameters = sensorName, measurementValue, measurementDateTime
    cursor.execute(query, parameters)
    cursor.commit()

    # Wait
    time.sleep(Ts)
```

Connection String

The Connection string has been put in a separate Python File called “database.py”:

```
def GetConnectionString():
    driver = "{ODBC Driver 17 for SQL Server}"
    server = ”xxxxxx"
    database = "LOGGINGSYSTEM"
    username = "sa"
    password = "xxxxxx"

    connectionString = "DRIVER=" + driver + ";SERVER=" + server + ";DATABASE=" + database + ";UID=" + username + ";PWD=" + password

    return connectionString
```

Logged Data



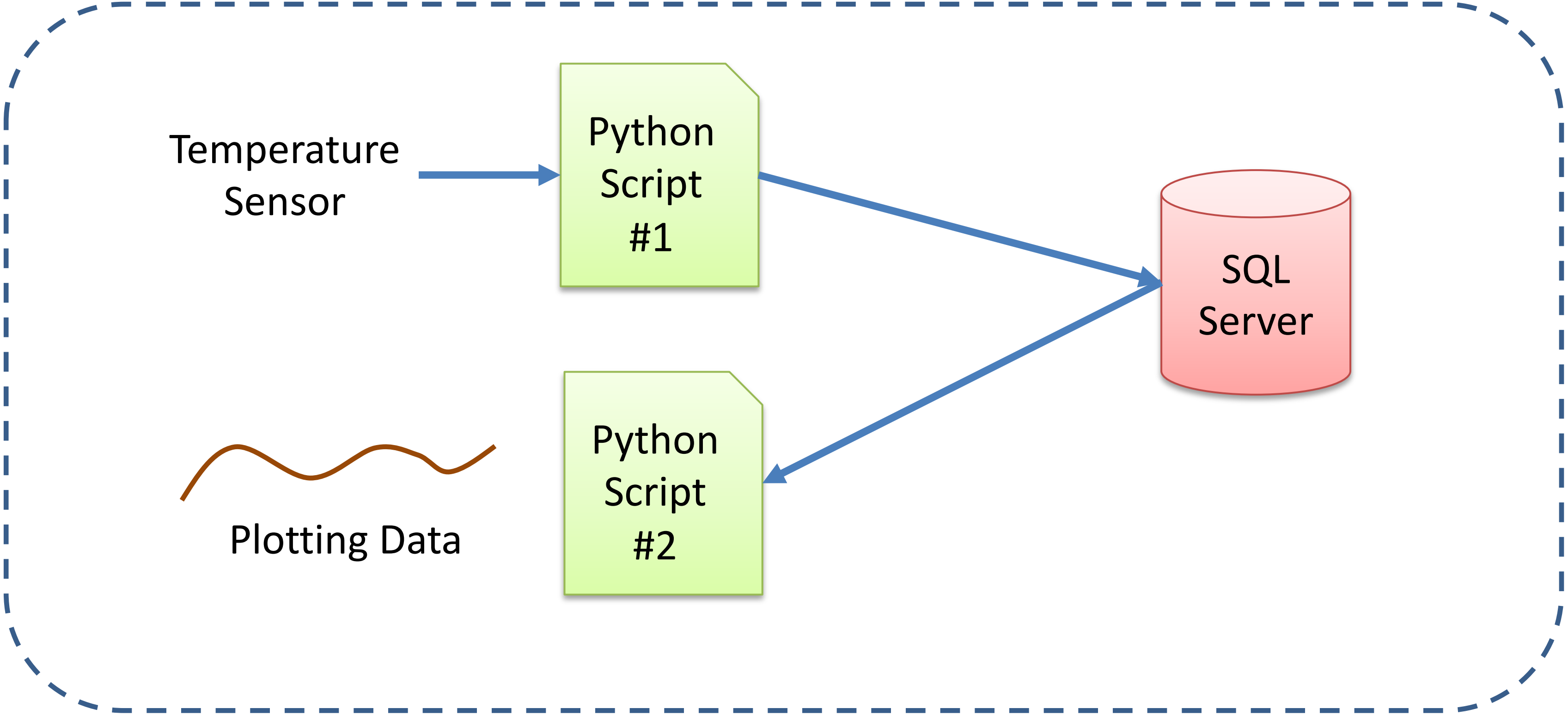
```sql
select * from MEASUREMENTDATA
```

	MeasurmentId	SensorName	MeasurementValue	MeasurementDateTime
1	4	Temperature	25	2021-11-25 13:46:11.000
2	5	Temperature	22	2021-11-25 13:46:21.000
3	6	Temperature	20	2021-11-25 13:46:31.000
4	7	Temperature	22	2021-11-25 13:46:41.000
5	8	Temperature	25	2021-11-25 13:46:51.000
6	9	Temperature	21	2021-11-25 13:47:01.000
7	10	Temperature	23	2021-11-25 13:47:12.000
8	11	Temperature	25	2021-11-25 13:47:22.000
9	12	Temperature	22	2021-11-25 13:47:32.000
10	13	Temperature	24	2021-11-25 13:47:42.000
11	14	Temperature	24	2021-11-25 13:47:52.000
12	15	Temperature	22	2021-11-25 13:48:02.000
13	16	Temperature	21	2021-11-25 13:48:12.000
14	17	Temperature	23	2021-11-25 13:48:22.000
15	18	Temperature	20	2021-11-25 13:48:32.000
16	19	Temperature	23	2021-11-25 13:48:42.000
17	20	Temperature	25	2021-11-25 13:48:52.000
18	21	Temperature	24	2021-11-25 13:49:02.000
19	22	Temperature	21	2021-11-25 13:49:12.000
20	23	Temperature	20	2021-11-25 13:49:22.000

Query executed successfully. XPS15HPH\SQLEXPRESS (13.0 RTM) | sa (53) | LOGGINGSYSTEM | 00:00:00 | 20 rows

System Overview

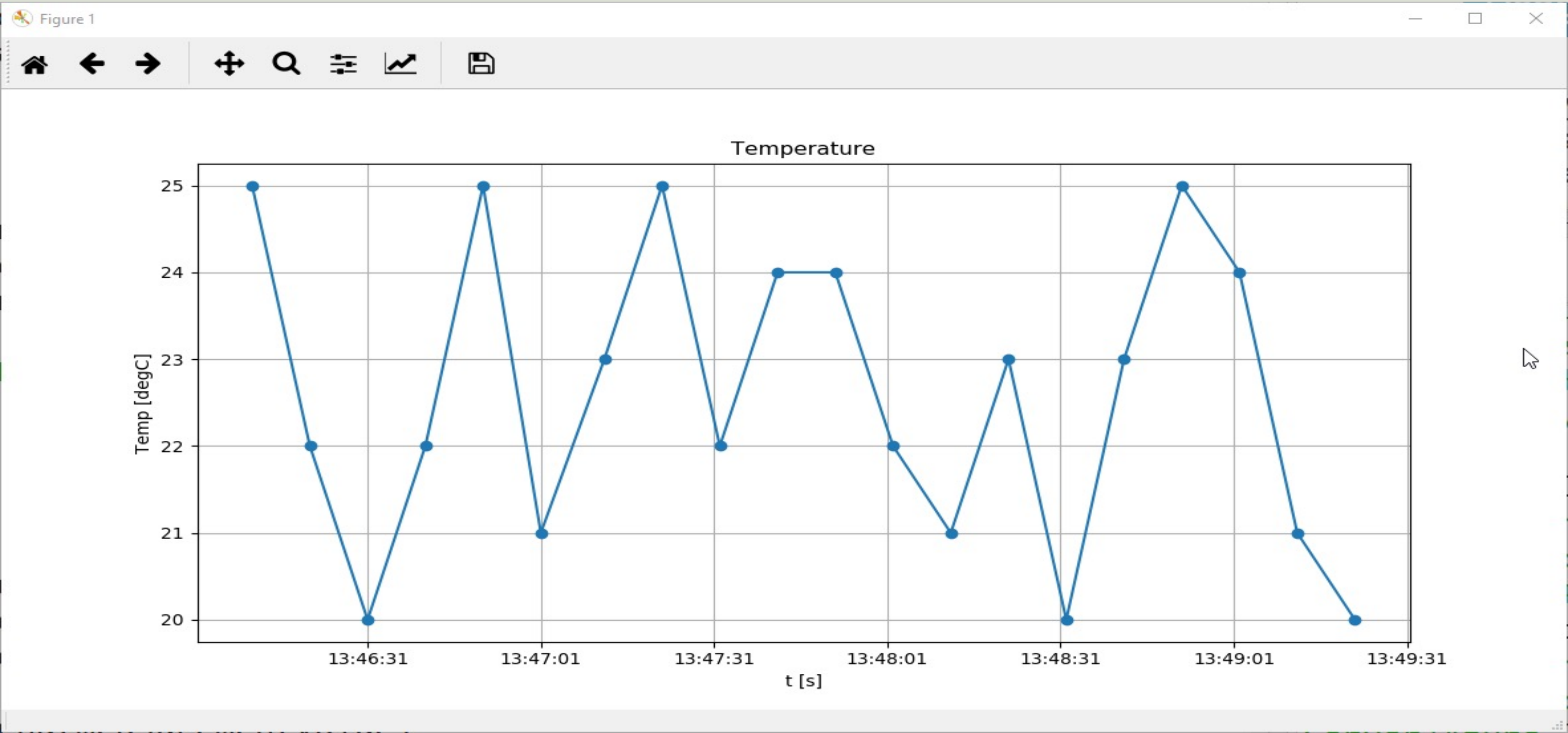
Local PC



Plotting Data

```
import pyodbc
import matplotlib.pyplot as plt
import database

sensorName = "Temperature"

# Connect to Database
connectionString = database.GetConnectionString()
conn = pyodbc.connect(connectionString)
cursor = conn.cursor()
query = "SELECT MeasurementValue, MeasurementDateTime FROM MEASUREMENTDATA WHERE SensorName=?"
parameters = [sensorName]

t = []; data = []

# Retrieving and Formatting Data
for row in cursor.execute(query, parameters):
    measurementValue = row.MeasurementValue
    measurementDateTime = row.MeasurementDateTime

    data.append(measurementValue)
    t.append(measurementDateTime)

# Plotting
plt.plot(t, data, 'o-')
plt.title('Temperature')
plt.xlabel('t [s]')
plt.ylabel('Temp [degC]')
plt.grid()
plt.show()
```

Plotted Data



Microsoft Azure

Hans-Petter Halvorsen

Table of Contents

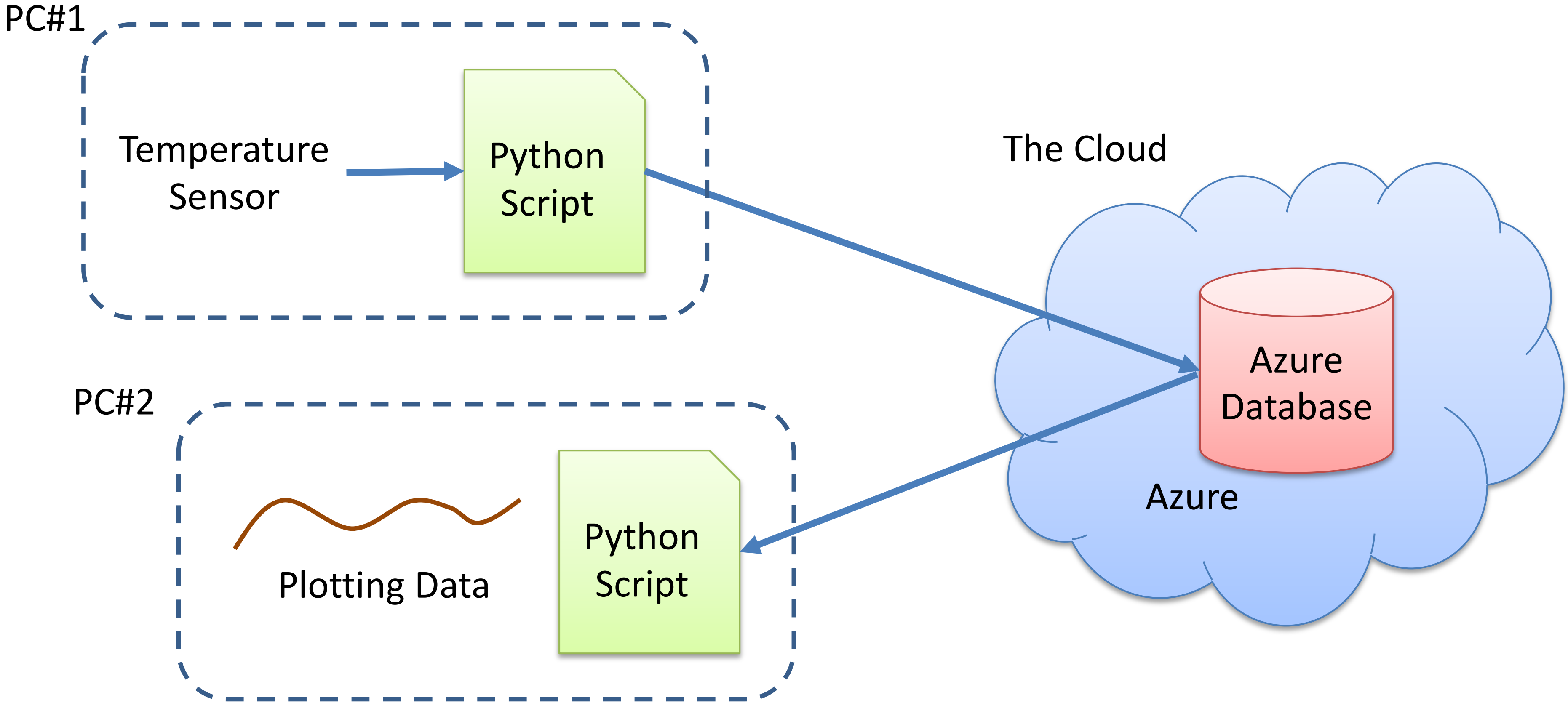
Microsoft Azure

- Microsoft Azure is a Cloud Platform from Microsoft
- You could say it is “Windows running in the Cloud”
- Here you can host Databases, Web Applications, Virtual Machines, etc.
- Azure Portal: https://portal.azure.com

Next Step

- We have created a local Datalogging System
- Next, we want to replace the local SQL Server Database with a Database in the Cloud
- We will use Microsoft Azure
- In that way others can get access to the logged data as well

System Overview



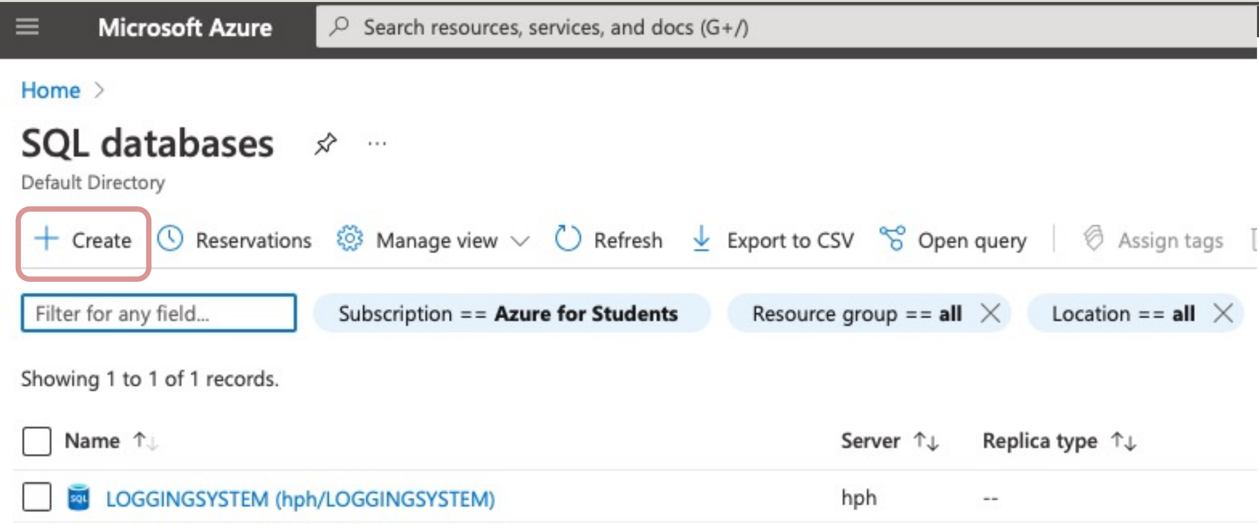
https://www.halvorsen.blog

Databases in Microsoft Azure

Hans-Petter Halvorsen

Table of Contents

Configure Database in Azure



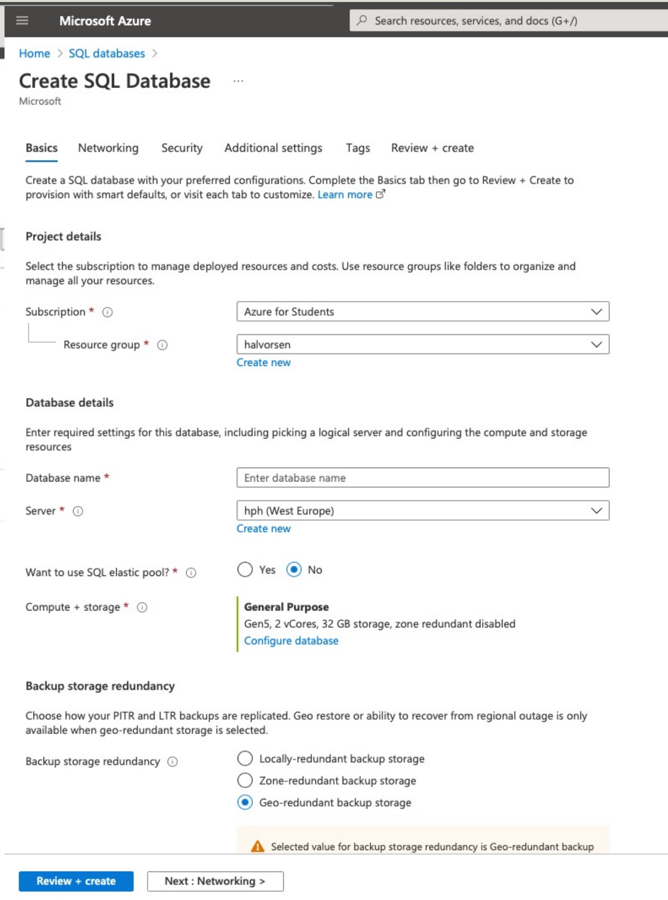
SQL databases

Default Directory

Subscription == **Azure for Students** | Resource group == **all** | Location == **all**

Showing 1 to 1 of 1 records.

Name	Server	Replica type
LOGGINGSYSTEM (hph/LOGGINGSYSTEM)	hph	--



Create SQL Database

Microsoft

Create a SQL database with your preferred configurations. Complete the Basics tab then go to Review + Create to provision with smart defaults, or visit each tab to customize. Learn more

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * : Azure for Students

Resource group * : halvorsen
Create new

Database details

Enter required settings for this database, including picking a logical server and configuring the compute and storage resources

Database name * : Enter database name

Server * : hph (West Europe)
Create new

Want to use SQL elastic pool? * ○ Yes ● No

Compute + storage * : **General Purpose**
Gen5, 2 vCores, 32 GB storage, zone redundant disabled
Configure database

Backup storage redundancy

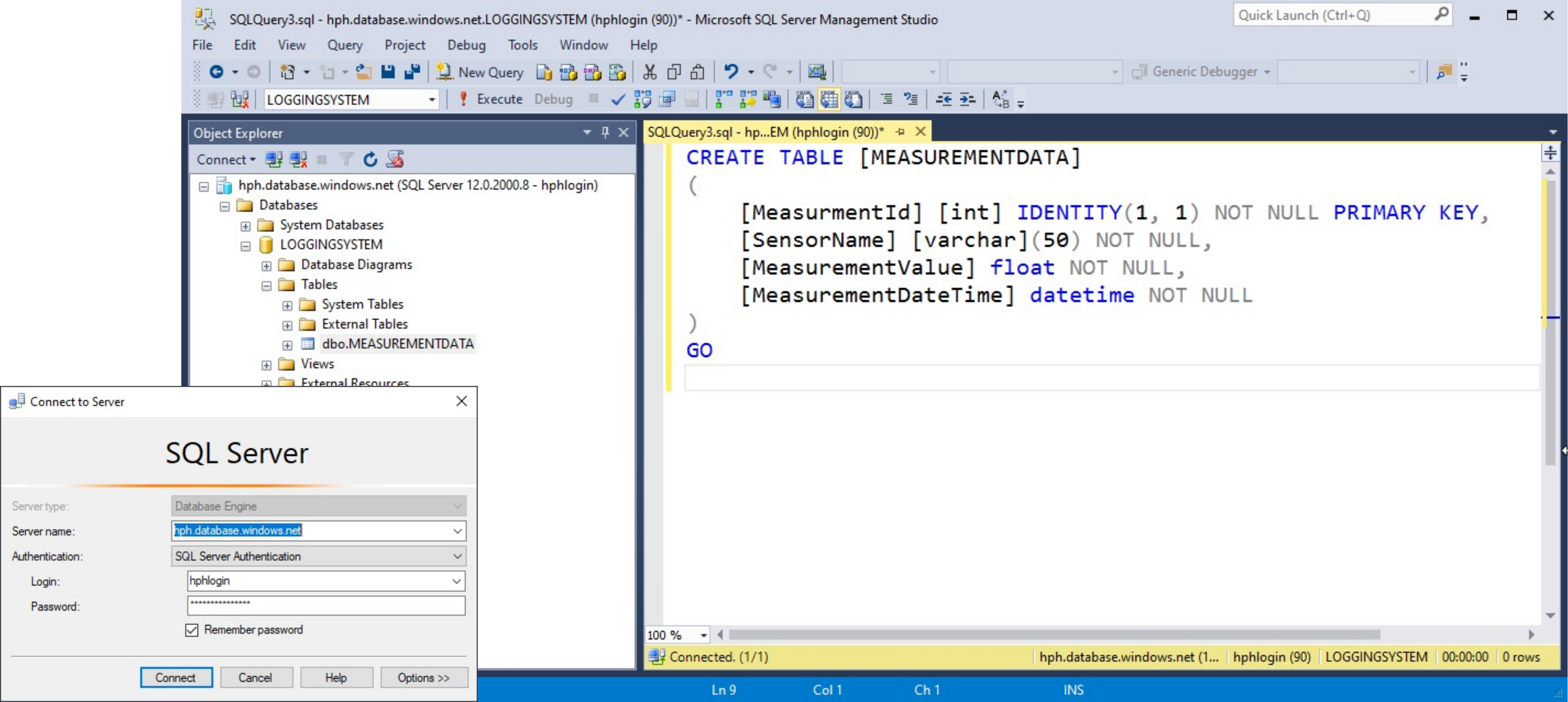
Choose how your PITR and LTR backups are replicated. Geo restore or ability to recover from regional outage is only available when geo-redundant storage is selected.

Backup storage redundancy
○ Locally-redundant backup storage
○ Zone-redundant backup storage
● Geo-redundant backup storage

⚠ Selected value for backup storage redundancy is Geo-redundant backup

Create Table

We will use SQL Server Management Studio and connect to the Azure Database:



SQLQuery3.sql - hph.database.windows.net.LOGGINGSYSTEM (hphlogin (90))* - Microsoft SQL Server Management Studio

```
CREATE TABLE [MEASUREMENTDATA]
(
    [MeasurmentId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
    [SensorName] [varchar](50) NOT NULL,
    [MeasurementValue] float NOT NULL,
    [MeasurementDateTime] datetime NOT NULL
)
GO
```

Connected. (1/1) | hph.database.windows.net (1... | hphlogin (90) | LOGGINGSYSTEM | 00:00:00 | 0 rows

Object Explorer:

- hph.database.windows.net (SQL Server 12.0.2000.8 - hphlogin)
  - Databases
    - System Databases
    - LOGGINGSYSTEM
      - Database Diagrams
      - Tables
        - System Tables
        - External Tables
        - dbo.MEASUREMENTDATA
      - Views
      - External Resources

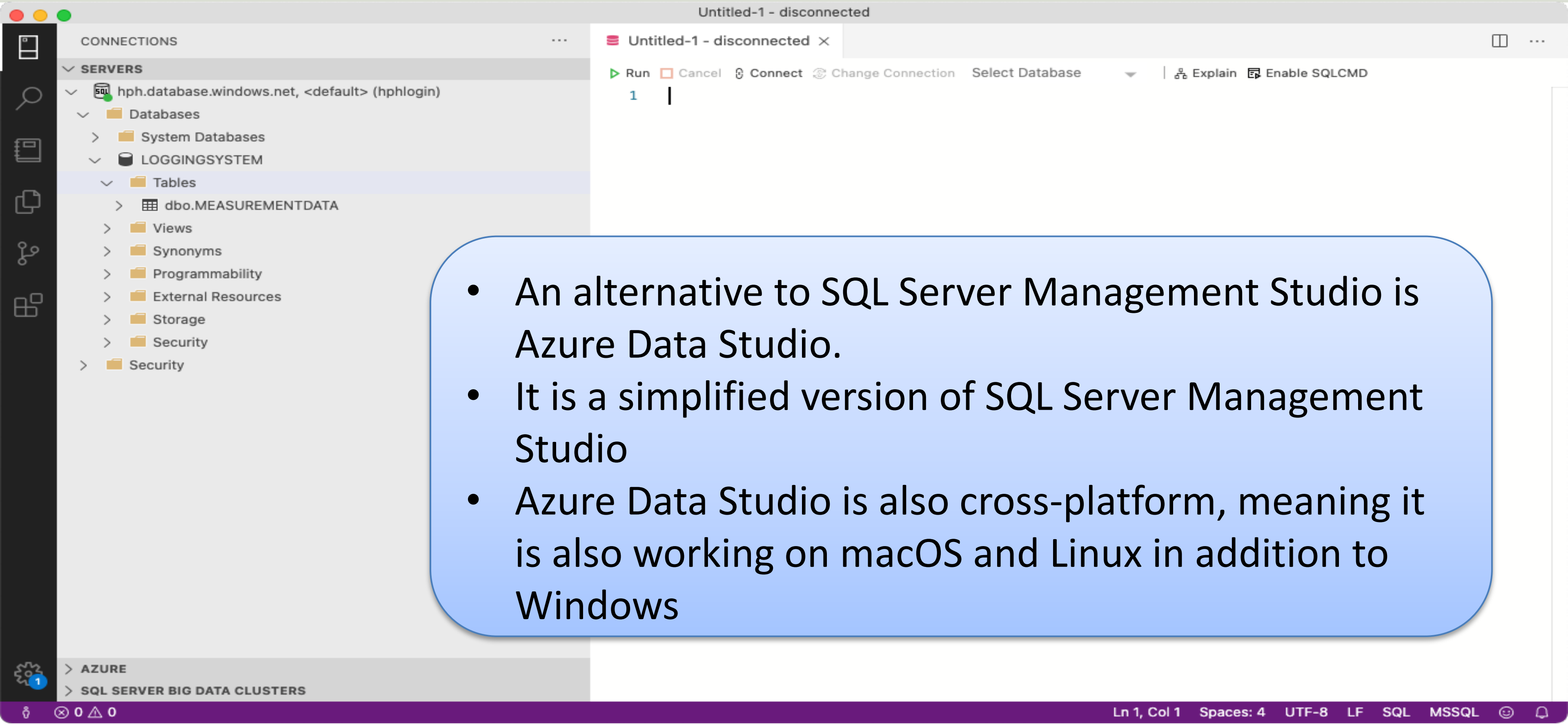
Connect to Server

SQL Server

Server type:	Database Engine
Server name:	hph.database.windows.net
Authentication:	SQL Server Authentication
Login:	hphlogin
Password:	***************

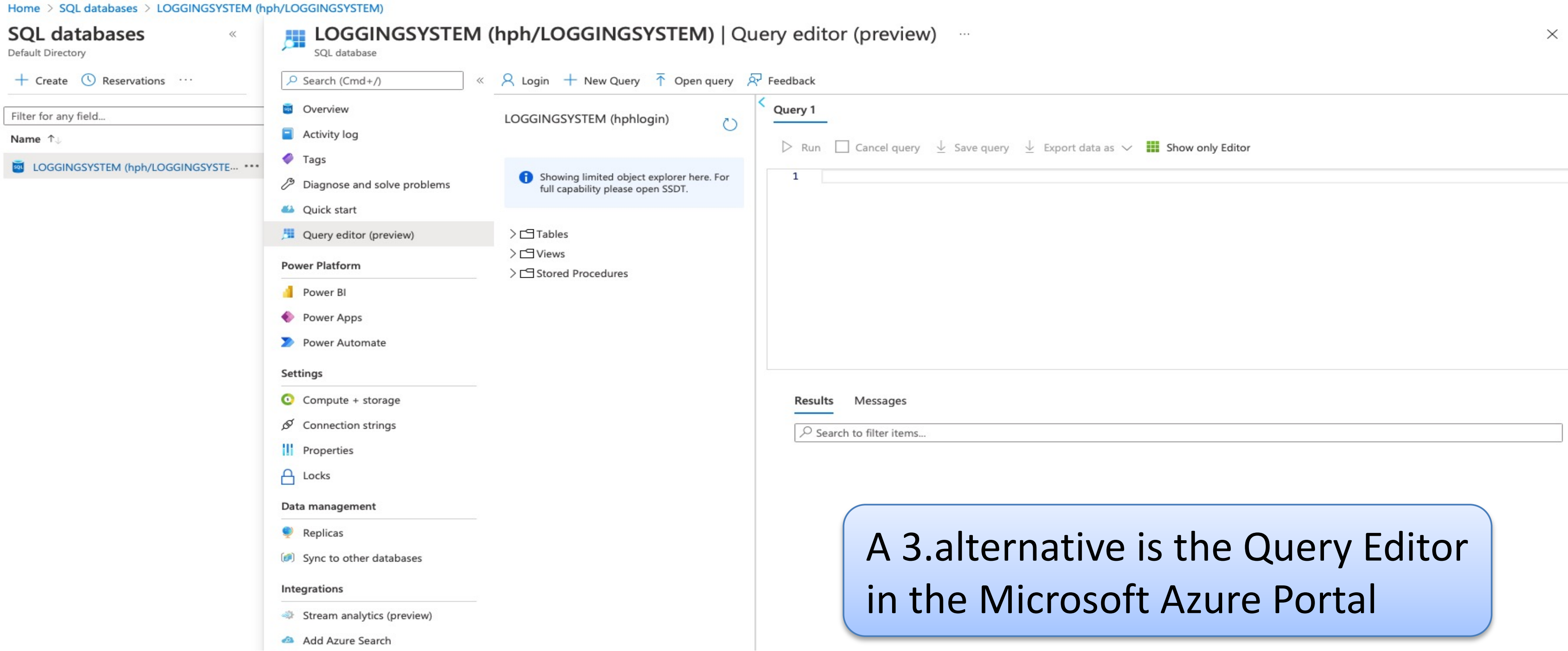
☑ Remember password

Azure Data Studio



- An alternative to SQL Server Management Studio is Azure Data Studio.
- It is a simplified version of SQL Server Management Studio
- Azure Data Studio is also cross-platform, meaning it is also working on macOS and Linux in addition to Windows

Azure Query Editor

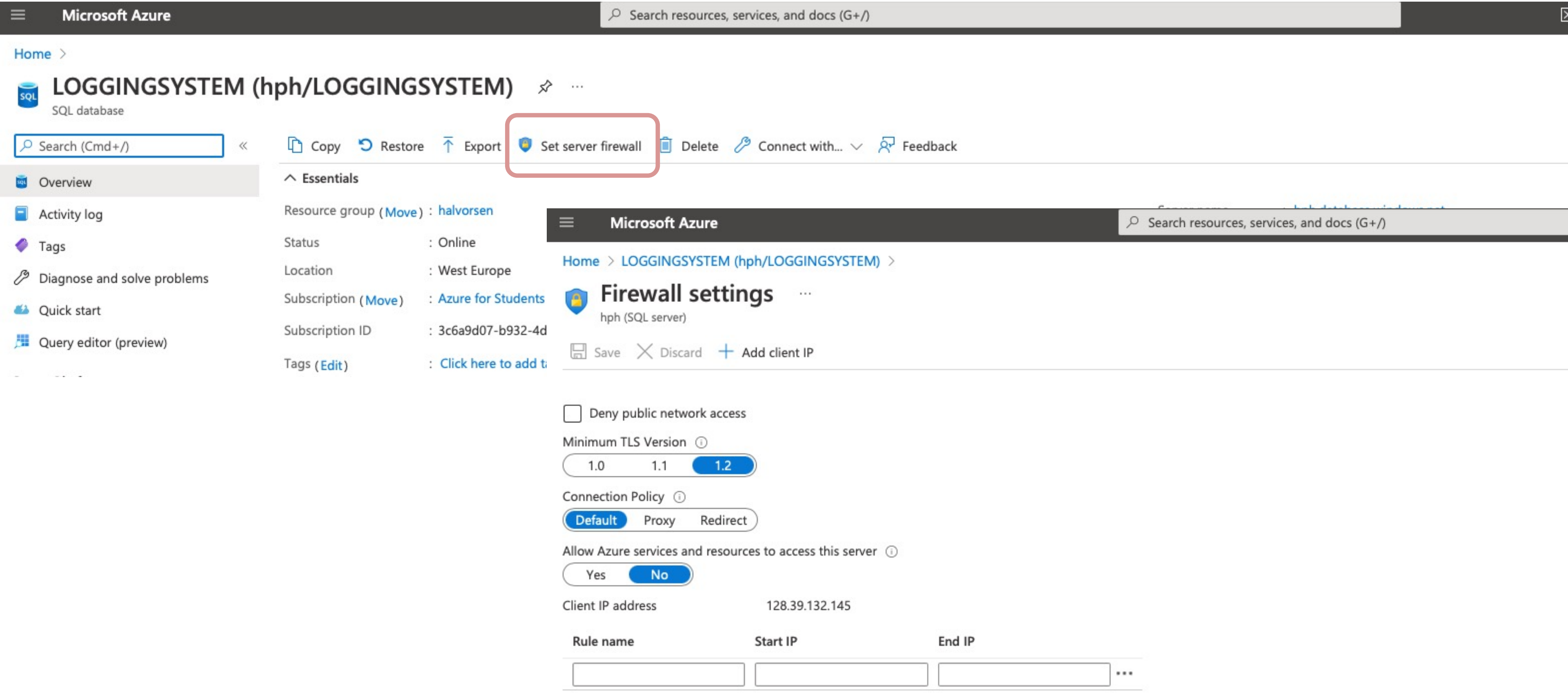


Showing limited object explorer here. For full capability please open SSDT.

A 3.alternative is the Query Editor in the Microsoft Azure Portal

Firewall

We need to give access to the computers running the Python Scripts



LOGGINGSYSTEM (hph/LOGGINGSYSTEM)

Resource group (Move) : halvorsen

Status : Online

Location : West Europe

Subscription (Move) : Azure for Students

Subscription ID : 3c6a9d07-b932-4d

Tags (Edit) : Click here to add t

Firewall settings

hph (SQL server)

□ Deny public network access

Minimum TLS Version: 1.0 1.1 [✓ 1.2]

Connection Policy: [✓ Default] Proxy Redirect

Allow Azure services and resources to access this server: Yes [✓ No]

Client IP address 128.39.132.145

Rule name	Start IP	End IP

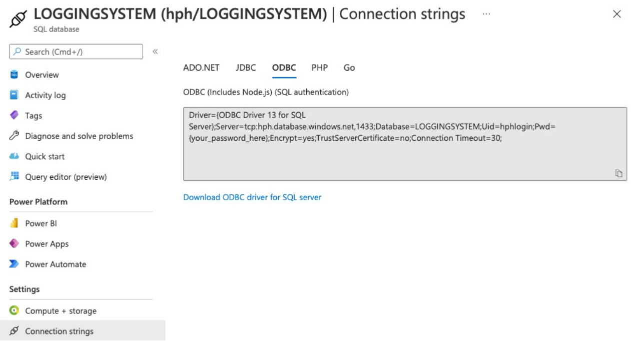
Datalogging Example Saving Data to Azure Database

Hans-Petter Halvorsen

Table of Contents

Python Code

- The Python Code is 100% the same
- The only thing we need to change is the Connection String
- You find the Connection String in the Azure Portal



LOGGINGSYSTEM (hph/LOGGINGSYSTEM) | Connection strings

SQL database

ODBC (Includes Node.js) (SQL authentication)

```
Driver={ODBC Driver 13 for SQL
Server};Server=tcp:hph.database.windows.net,1433;Database=LOGGINGSYSTEM;Uid=hphlogin;Pwd=
{your_password_here};Encrypt=yes;TrustServerCertificate=no;Connection Timeout=30;
```

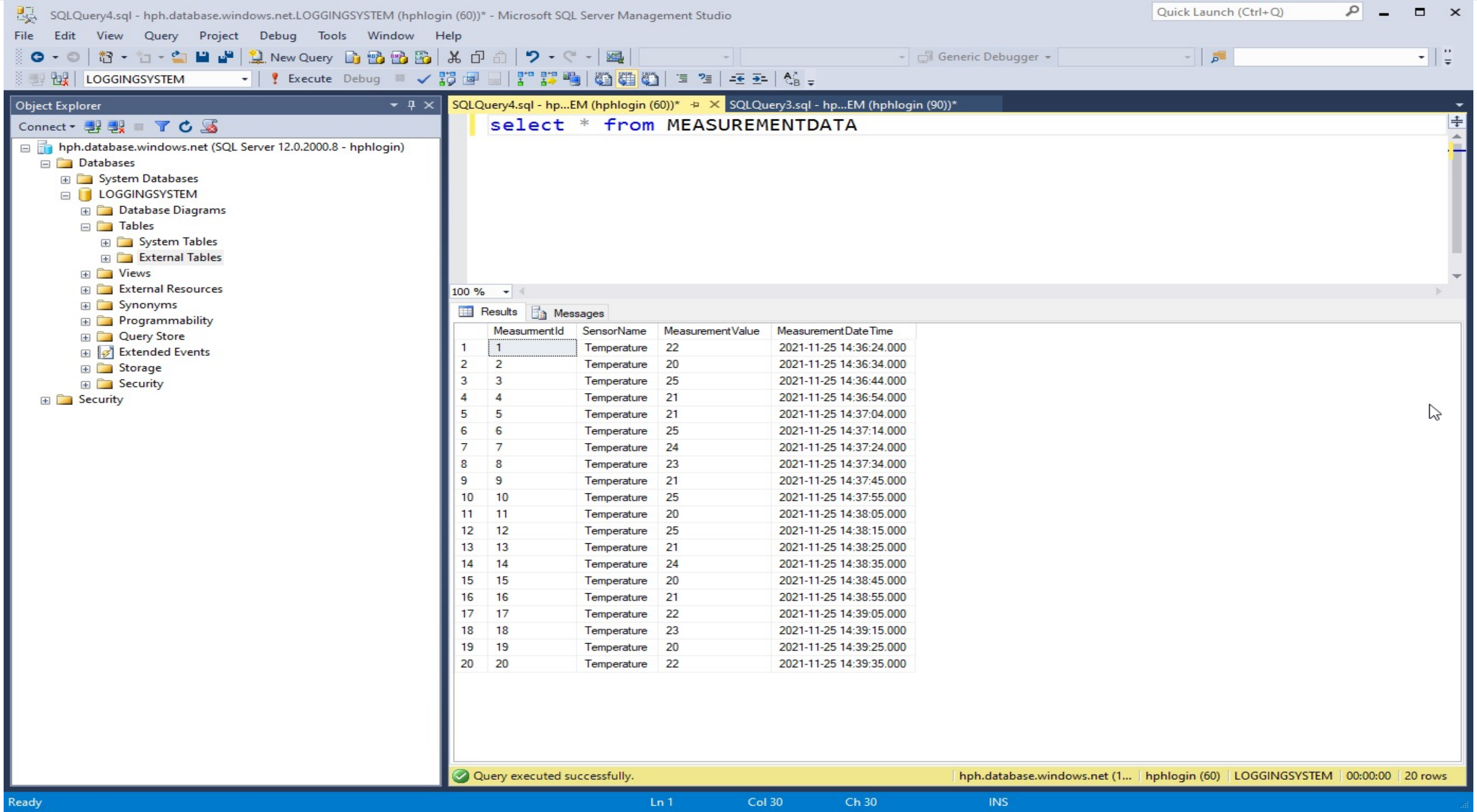
Connection String

The Connection string has been put in a separate Python File called “database.py”:

```
def GetConnectionString():
    driver = "{ODBC Driver 17 for SQL Server}"
    server = "xxxxxx"
    database = "LOGGINGSYSTEM"
    username = "sa"
    password = "xxxxxx"

    connectionString = "DRIVER=" + driver + ";SERVER=" + server + ";DATABASE=" + database + ";UID=" + username + ";PWD=" + password

    return connectionString

def GetConnectionStringAzure():
    driver = "{ODBC Driver 17 for SQL Server}"
    server = "xxx.database.windows.net"
    database = "LOGGINGSYSTEM"
    username = "xxxxxx"
    password = "xxxxxxx"

    connectionString = "DRIVER=" + driver + ";SERVER=" + server + ";DATABASE=" + database + ";UID=" + username + ";PWD=" + password

    return connectionString
```

```python
import pyodbc
import random
import time
from datetime import datetime
import database

# Connect to Database
connectionString = database.GetConnectionStringAzure()
conn = pyodbc.connect(connectionString)
cursor = conn.cursor()
query = "INSERT INTO MEASUREMENTDATA (SensorName, MeasurementValue, MeasurementDateTime) VALUES (?,?,?)"

sensorName = "Temperature"
Ts = 10 # Sampling Time
N = 20
for k in range(N):
    # Generate Random Data
    LowLimit = 20
    UpperLimit = 25
    measurementValue = random.randint(LowLimit, UpperLimit)

    #Find Date and Time
    now = datetime.now()
    datetimeformat = "%Y-%m-%d %H:%M:%S"
    measurementDateTime = now.strftime(datetimeformat)

    # Insert Data into Database
    parameters = sensorName, measurementValue, measurementDateTime
    cursor.execute(query, parameters)
    cursor.commit()

    # Wait
    time.sleep(Ts)
```

Final Results

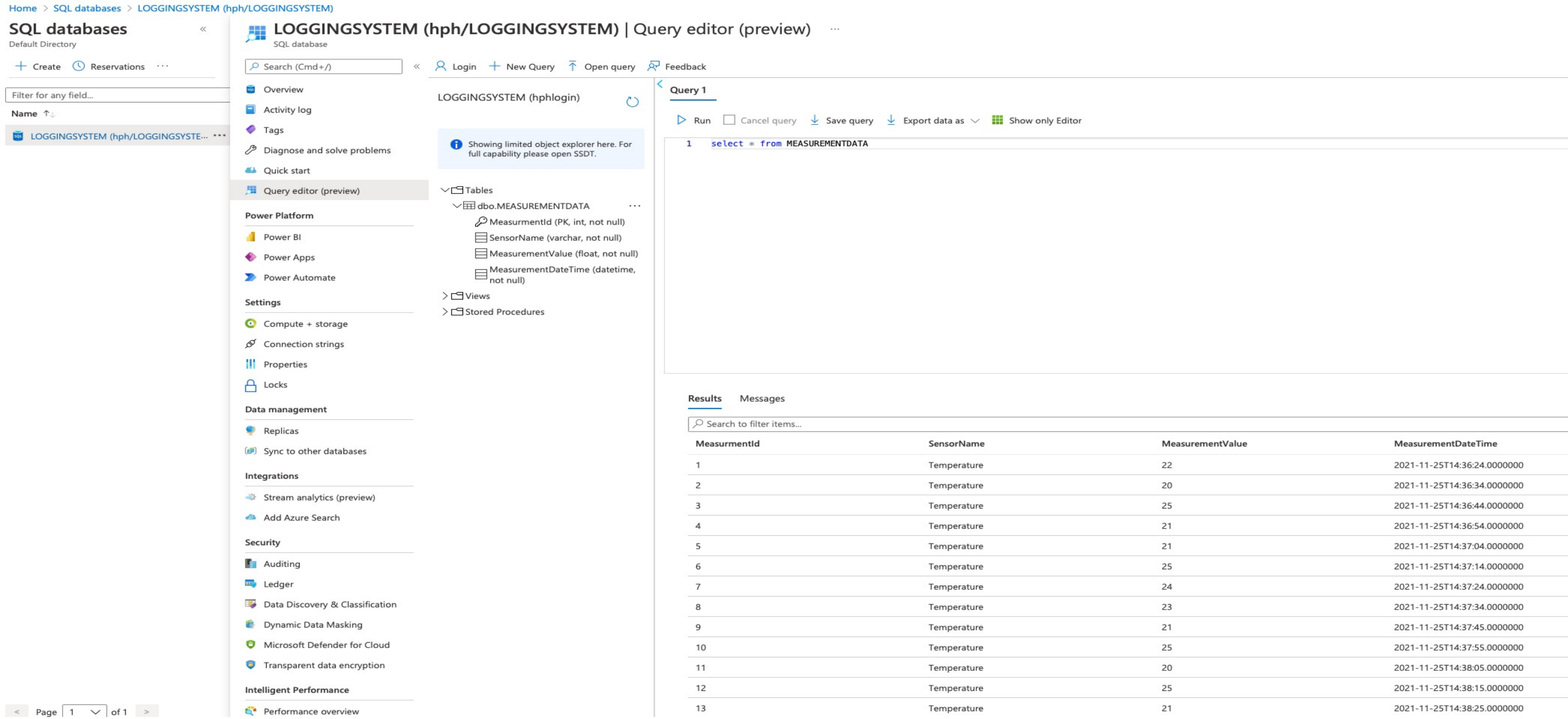


```
select * from MEASUREMENTDATA
```

	MeasurmentId	SensorName	MeasurementValue	MeasurementDateTime
1	1	Temperature	22	2021-11-25 14:36:24.000
2	2	Temperature	20	2021-11-25 14:36:34.000
3	3	Temperature	25	2021-11-25 14:36:44.000
4	4	Temperature	21	2021-11-25 14:36:54.000
5	5	Temperature	21	2021-11-25 14:37:04.000
6	6	Temperature	25	2021-11-25 14:37:14.000
7	7	Temperature	24	2021-11-25 14:37:24.000
8	8	Temperature	23	2021-11-25 14:37:34.000
9	9	Temperature	21	2021-11-25 14:37:45.000
10	10	Temperature	25	2021-11-25 14:37:55.000
11	11	Temperature	20	2021-11-25 14:38:05.000
12	12	Temperature	25	2021-11-25 14:38:15.000
13	13	Temperature	21	2021-11-25 14:38:25.000
14	14	Temperature	24	2021-11-25 14:38:35.000
15	15	Temperature	20	2021-11-25 14:38:45.000
16	16	Temperature	21	2021-11-25 14:38:55.000
17	17	Temperature	22	2021-11-25 14:39:05.000
18	18	Temperature	23	2021-11-25 14:39:15.000
19	19	Temperature	20	2021-11-25 14:39:25.000
20	20	Temperature	22	2021-11-25 14:39:35.000

Query executed successfully. hph.database.windows.net (1... hphlogin (60) LOGGINGSYSTEM 00:00:00 20 rows

Final Results

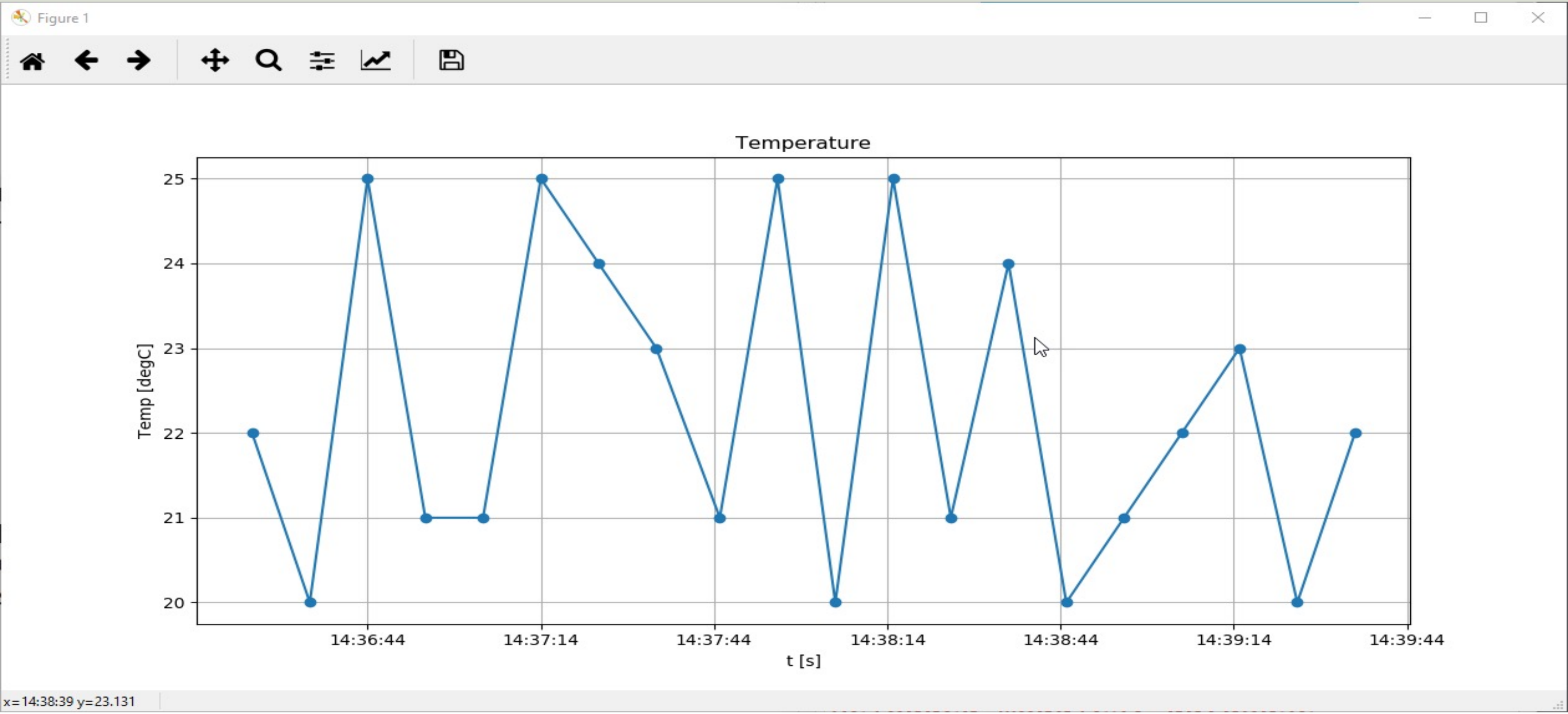


```
select * from MEASUREMENTDATA
```

MeasurmentId	SensorName	MeasurementValue	MeasurementDateTime
1	Temperature	22	2021-11-25T14:36:24.0000000
2	Temperature	20	2021-11-25T14:36:34.0000000
3	Temperature	25	2021-11-25T14:36:44.0000000
4	Temperature	21	2021-11-25T14:36:54.0000000
5	Temperature	21	2021-11-25T14:37:04.0000000
6	Temperature	25	2021-11-25T14:37:14.0000000
7	Temperature	24	2021-11-25T14:37:24.0000000
8	Temperature	23	2021-11-25T14:37:34.0000000
9	Temperature	21	2021-11-25T14:37:45.0000000
10	Temperature	25	2021-11-25T14:37:55.0000000
11	Temperature	20	2021-11-25T14:38:05.0000000
12	Temperature	25	2021-11-25T14:38:15.0000000
13	Temperature	21	2021-11-25T14:38:25.0000000

```python
import pyodbc
import matplotlib.pyplot as plt
import database

sensorName = "Temperature"

# Connect to Database
connectionString = database.GetConnectionStringAzure()
conn = pyodbc.connect(connectionString)
cursor = conn.cursor()
query = "SELECT MeasurementValue, MeasurementDateTime FROM MEASUREMENTDATA WHERE SensorName=?"
parameters = [sensorName]

t = []; data = []

# Retrieving and Formatting Data
for row in cursor.execute(query, parameters):
    measurementValue = row.MeasurementValue
    measurementDateTime = row.MeasurementDateTime

    data.append(measurementValue)
    t.append(measurementDateTime)

# Plotting
plt.plot(t, data, 'o-')
plt.title('Temperature')
plt.xlabel('t [s]')
plt.ylabel('Temp [degC]')
plt.grid()
plt.show()
```

Final Results



Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog