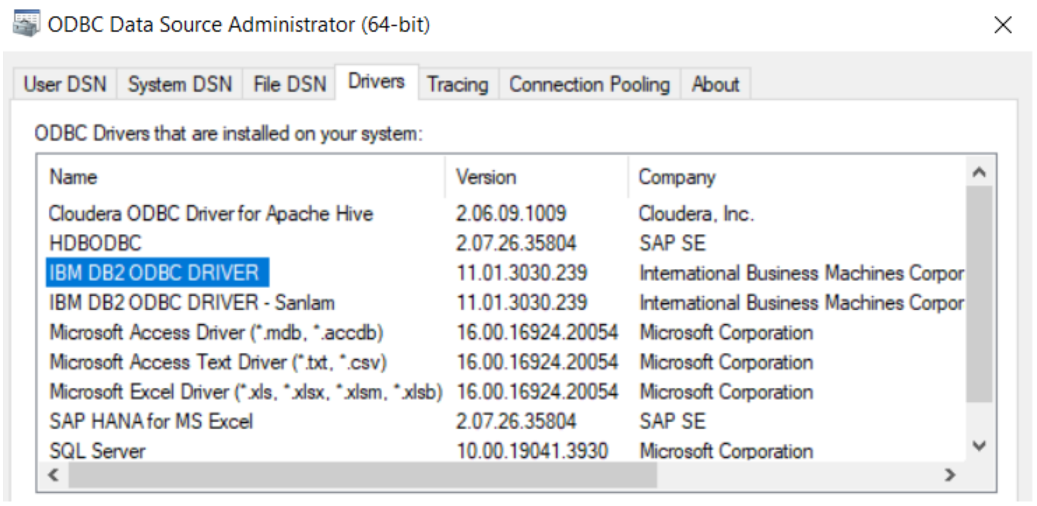
AA - Python to DB2

This page serves as a platform for connecting to DB2 directly from a Python environment through an ODBC Connection. This pattern is tailored for the purpose of exporting data from DB2 and being able to manipulate it.

Pre-requisites for DB2

A version of the IBM DB2 ODBC Driver needs to be installed locally. Please see details below:

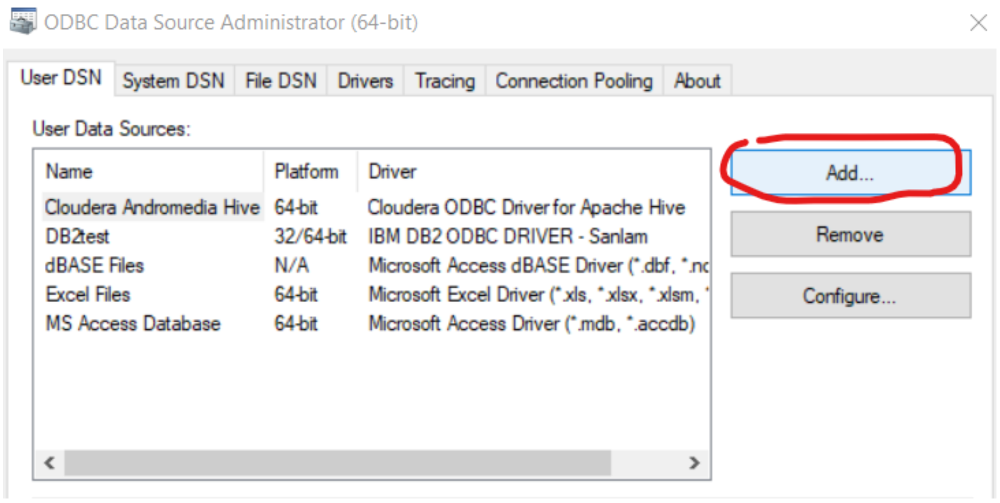


ODBC Drivers that are installed on your system:

Name	Version	Company
Cloudera ODBC Driver for Apache Hive	2.06.09.1009	Cloudera, Inc.
HDBODBC	2.07.26.35804	SAP SE
IBM DB2 ODBC DRIVER	11.01.3030.239	International Business Machines Corpor
IBM DB2 ODBC DRIVER - Sanlam	11.01.3030.239	International Business Machines Corpor
Microsoft Access Driver (*.mdb, *.accdb)	16.00.16924.20054	Microsoft Corporation
Microsoft Access Text Driver (*.txt, *.csv)	16.00.16924.20054	Microsoft Corporation
Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)	16.00.16924.20054	Microsoft Corporation
SAP HANA for MS Excel	2.07.26.35804	SAP SE
SQL Server	10.00.19041.3930	Microsoft Corporation

Now on the '*User DSN*' tab, follow the screenshots and instructions below:

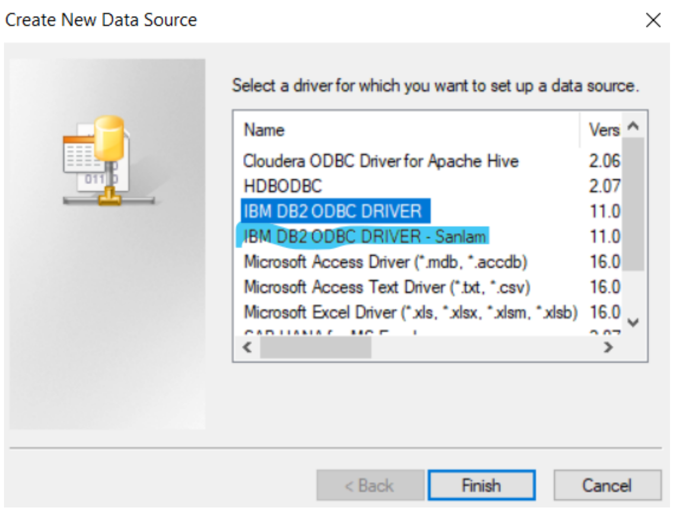
Click on *'Add...'.*



User Data Sources:

Name	Platform	Driver
Cloudera Andromedia Hive	64-bit	Cloudera ODBC Driver for Apache Hive
DB2test	32/64-bit	IBM DB2 ODBC DRIVER - Sanlam
dBASE Files	N/A	Microsoft Access dBASE Driver (*.dbf, *.nc
Excel Files	64-bit	Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *
MS Access Database	64-bit	Microsoft Access Driver (*.mdb, *.accdb)

Select either *'IBM DB2 ODBC DRIVER'* and click *'Finish'*

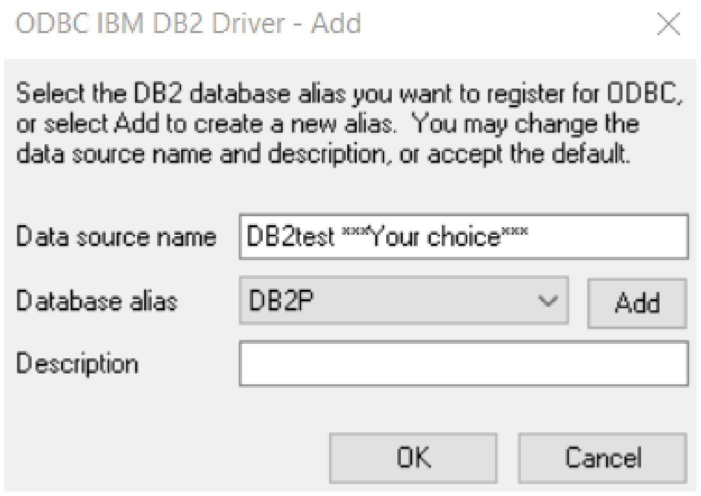


Create New Data Source

Select a driver for which you want to set up a data source.

Name	Vers
Cloudera ODBC Driver for Apache Hive	2.06
HDBODBC	2.07
IBM DB2 ODBC DRIVER	11.0
IBM DB2 ODBC DRIVER - Sanlam	11.0
Microsoft Access Driver (*.mdb, *.accdb)	16.0
Microsoft Access Text Driver (*.txt, *.csv)	16.0
Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)	16.0

Enter a *'Data source name'*, this can be whatever you want but please **remember what you named it for the code portion**. Select ***'DB2P'*** as the *'Databas e Alias'* and click *'OK'*.



ODBC IBM DB2 Driver - Add

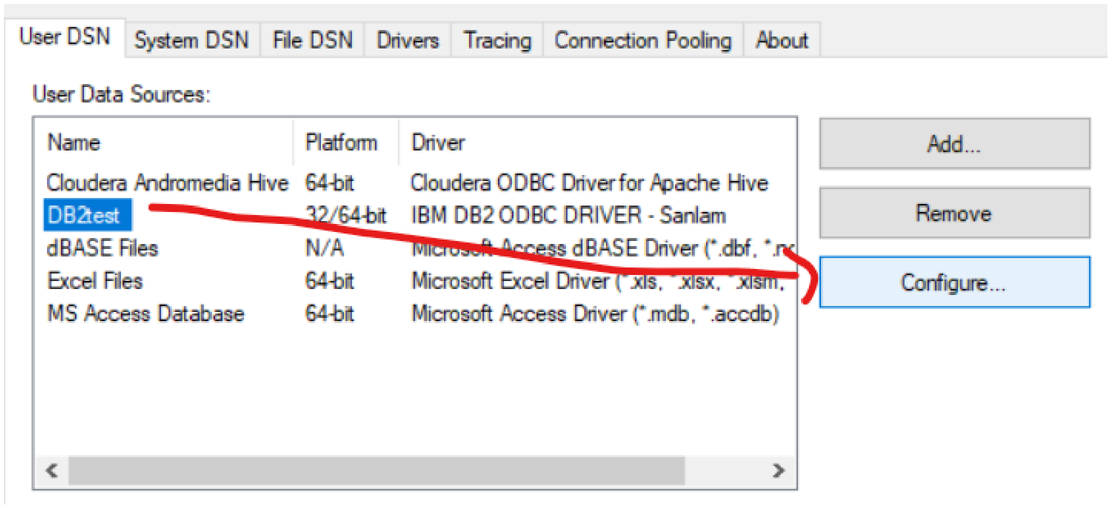
Select the DB2 database alias you want to register for ODBC, or select Add to create a new alias. You may change the data source name and description, or accept the default.

Data source name: DB2test ***Your choice***

Database alias: DB2P

Description:

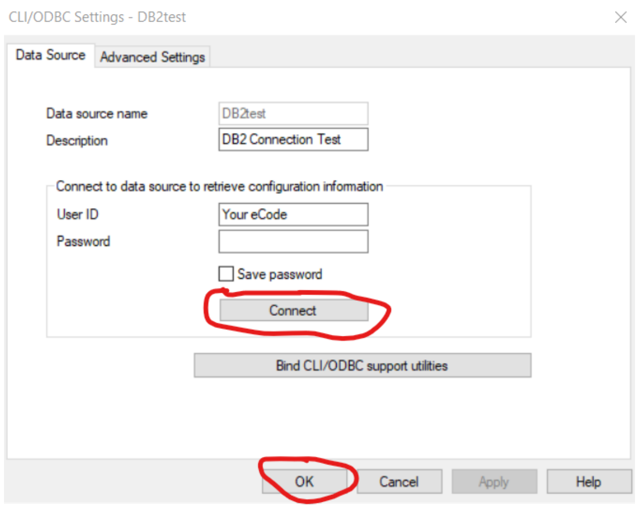
With the new added data source selected, click on *'Configure...'*



User Data Sources:

Name	Platform	Driver
Cloudera Andromedia Hive	64-bit	Cloudera ODBC Driver for Apache Hive
DB2test	32/64-bit	IBM DB2 ODBC DRIVER - Sanlam
dBASE Files	N/A	Microsoft Access dBASE Driver (*.dbf, *.nd
Excel Files	64-bit	Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm,
MS Access Database	64-bit	Microsoft Access Driver (*.mdb, *.accdb)

Add in your e-Code and **RACF password**, click *'Connect'* to test the connection. When done, click *'OK'*


CLI/ODBC Settings - DB2test

Data source name: DB2test
Description: DB2 Connection Test

Connect to data source to retrieve configuration information
User ID: Your eCode
Password:
Save password

The following Pre-requisites are required on the Python front once the ODBC driver has been set up:

Software requirements

Requirement	Description	version
Base Python	The Code engine	latest version
(Optional) Jupyter Notebook or any prefrered IDE	IDE for Python	latest version

Python Package requirements:

Ensure that these packages are installed before the execution of the R script below. This will ensure that the connection to Hive is successful

Package Name
pyodbc
getPass

The following code will prompt you to enter your **data source name (created above), e-code and RACF password as inputs**. Take note of where the DSN name must be replaced and the SQL query to test your connection.

Python Script: Python to DB2

```
################################ Installing required packages ##########################################

### Uncomment if needed
# !pip install pyodbc
# !pip install getPass

################################ Script to Create a Connection to DB2 from Python
##########################################   import pyodbc
import getpass

class DatabaseConnector:
    """
    A class for connecting to a database using pyodbc.

    Attributes:
    - dsn_name (str): The Data Source Name (DSN) for the database.
    - username (str): The username for accessing the database.
    - password (str): The password for accessing the database.
    - credentials_set (bool): Flag indicating whether credentials have been set.
    """

    _instance = None

    def __new__(cls, *args, **kwargs):
        """
        Creates a new instance of the DatabaseConnector class if it doesn't already exist.

        Returns:
        - DatabaseConnector: The DatabaseConnector instance.
        """
        if not cls._instance:
            cls._instance = super().__new__(cls)
            cls._instance.dsn_name = None
            cls._instance.username = None
            cls._instance.password = None
            cls._instance.credentials_set = False
        return cls._instance

    def connect_to_database(self):
        """
        Connects to the database using the stored credentials.

        Prompts the user to input credentials if they haven't been set or if the previous connection attempt
failed.

        Returns:
        - pyodbc.Connection or None: A connection object if successful, None if connection failed.
        """
        max_tries = 3
        tries = 0

        while tries < max_tries:
            if not self.credentials_set:
                self.dsn_name = input('DSN Name:\n')
                self.username = input('Username:\n')
                self.password = getpass.getpass(prompt='Password:\n')
                self.credentials_set = True

            try:
                conn = pyodbc.connect("DSN="+self.dsn_name+";UID="+self.username+";PWD="+self.password)
                print("Connected to the database successfully!")
                return conn
            except pyodbc.Error as e:
                print("Error connecting to the database:", e)
                self.credentials_set = False  # Reset to False if credentials are incorrect
                tries += 1
                print(f"You have {max_tries - tries} tries left.")

        print("Maximum number of tries exceeded. Please check your credentials.")
        return None

# Example usage:
connector = DatabaseConnector()
con = connector.connect_to_database()
# Use con to interact with the database


  ################ Testing a query #############

cur = con.cursor()

cur.execute("SELECT ... FROM BIDTB.....") ##### a small SQL query just to test #####

data = cur.fetchall()

data
```