Please note that the code snippet below constructs a database connection string (DSN) using a formatted string approach in Python.

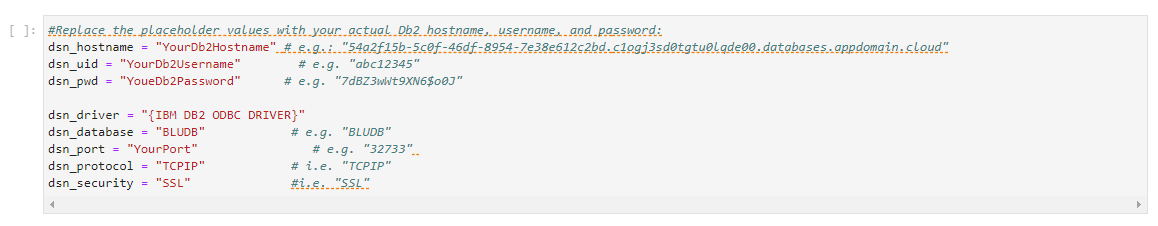


Let's break it down step by step:

1. The code defines a format template for the connection string dsn. The template includes placeholders for various parameters needed for the database connection.
2. The placeholders in the template are:
   * {0}: Driver
   * {1}: Database
   * {2}: Hostname
   * {3}: Port
   * {4}: Protocol
   * {5}: User ID (UID)
   * {6}: Password (PWD)
   * {7}: Security settings (SECURITY)
3. The .format() method is used to substitute the placeholders in the template with actual values. It takes the values of variables dsn\_driver, dsn\_database, dsn\_hostname, dsn\_port, dsn\_protocol, dsn\_uid, dsn\_pwd, and dsn\_security, and inserts them into the placeholders in the dsn string.
4. Finally, the constructed connection string dsn is printed to the console to verify that the correct values are specified for each parameter.

So, for example, if the variables dsn\_driver, dsn\_database, etc., are properly defined with appropriate values before this code snippet is executed, the printed DSN string will represent a valid connection string that can be used to establish a connection to the specified database with the provided credentials and settings.

In the lab, we have already included a code block that you just need to fill with the appropriate values so that the DSN string will establish a connection to the database.



* Please note that the below code interacts with an IBM DB2 database connection to retrieve server information and then prints out specific attributes of that information.

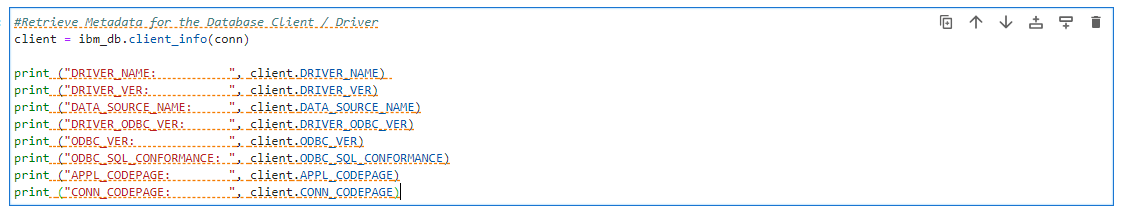


Let's break it down:

1. server = ibm\_db.server\_info(conn): This line calls the ibm\_db.server\_info() function, which retrieves information about the connected DB2 server. The conn variable represents the database connection object previously established.
2. print("DBMS\_NAME: ", server.DBMS\_NAME): This line prints the name of the DBMS (Database Management System) to which the connection is made. The server.DBMS\_NAME attribute contains this information.
3. print("DBMS\_VER: ", server.DBMS\_VER): This line prints the version of the DBMS. The server.DBMS\_VER attribute contains this information.
4. print("DB\_NAME: ", server.DB\_NAME): This line prints the name of the specific database to which the connection is made. The server.DB\_NAME attribute contains this information.

In summary, this code fetches server information from the connected DB2 database and then prints out specific attributes of that information, such as the DBMS name, version, and the name of the connected database. This can be useful for verifying the connection details and for diagnostic purposes.

* Please note that the below code interacts with an IBM DB2 database connection to retrieve client information and then prints out specific attributes of that information.



Let's break it down:

1. client = ibm\_db.client\_info(conn): This line calls the ibm\_db.client\_info() function, which retrieves information about the client (application) that established the database connection. The conn variable represents the database connection object previously established.
2. print("DRIVER\_NAME: ", client.DRIVER\_NAME): This line prints the name of the driver used by the client application. The client.DRIVER\_NAME attribute contains this information.
3. print("DRIVER\_VER: ", client.DRIVER\_VER): This line prints the version of the driver used by the client application. The client.DRIVER\_VER attribute contains this information.
4. print("DATA\_SOURCE\_NAME: ", client.DATA\_SOURCE\_NAME): This line prints the data source name associated with the client application. The client.DATA\_SOURCE\_NAME attribute contains this information.
5. print("DRIVER\_ODBC\_VER: ", client.DRIVER\_ODBC\_VER): This line prints the version of the ODBC (Open Database Connectivity) driver used by the client application. The client.DRIVER\_ODBC\_VER attribute contains this information.
6. print("ODBC\_VER: ", client.ODBC\_VER): This line prints the version of the ODBC specification supported by the client application. The client.ODBC\_VER attribute contains this information.
7. print("ODBC\_SQL\_CONFORMANCE: ", client.ODBC\_SQL\_CONFORMANCE): This line prints the level of SQL conformance supported by the ODBC driver used by the client application. The client.ODBC\_SQL\_CONFORMANCE attribute contains this information.
8. print("APPL\_CODEPAGE: ", client.APPL\_CODEPAGE): This line prints the code page used by the client application for character data. The client.APPL\_CODEPAGE attribute contains this information.
9. print("CONN\_CODEPAGE: ", client.CONN\_CODEPAGE): This line prints the code page used by the database connection for character data. The client.CONN\_CODEPAGE attribute contains this information.

In summary, this code fetches client information from the connected DB2 database and then prints out specific attributes of that information, such as the driver name, version, data source name, ODBC version, SQL conformance level, and code pages used for character data. This information can be useful for understanding the characteristics of the client application and its interaction with the database.

Please note that the below code attempts to establish a connection to a database using the ibm\_db.connect() function provided by the IBM DB2 Python driver.



Let's break it down:

1. try: -This keyword indicates the start of a block of code where exceptions will be caught. Python will attempt to execute the code inside the try block, and if an exception occurs, it will be caught by the except block.
2. conn = ibm\_db.connect(dsn, "", ""): This line attempts to establish a connection to the database using the ibm\_db.connect() function. The dsn variable contains the connection string needed to connect to the database. The empty strings "" represent the username and password, which are not provided here. The ibm\_db.connect() function returns a connection object (conn) if the connection is successful.
3. print ("Connected to database: ", dsn\_database, "as user: ", dsn\_uid, "on host: ", dsn\_hostname): If the connection is successfully established, this line prints a message confirming the successful connection. It includes information about the database name (dsn\_database), the user ID (dsn\_uid), and the hostname (dsn\_hostname).
4. except:- This keyword indicates the start of a block of code that will be executed if an exception occurs within the try block.
5. print ("Unable to connect: ", ibm\_db.conn\_errormsg() ): If an exception occurs during the attempt to connect to the database, this line prints an error message. The ibm\_db.conn\_errormsg() function is used to retrieve the error message associated with the connection attempt.

In summary, this code tries to establish a connection to a database using the provided connection string (dsn). If the connection is successful, it prints a confirmation message. If an error occurs during the connection attempt, it prints an error message.