In [ ]:
```
#IASK 2: IDENTIFY THE DATABASE CONNECTION CREDENTIALS
#CONNECTING TO DASHDB OR DB2 DATABASE REQUIRES THE FOLLOWING INFORMATION
```

In [21]:
```
import ibm_db
```

In [11]:
```
dsn_hostname = "ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.data
dsn_uid = "dmq73837"        # e.g. "abc12345"
dsn_pwd = "A6ZZPRoaOUUNvI1J"      # e.g. "7dBZ3wWt9XN6$o0J"

dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "bludb"            # e.g. "BLUDB"
dsn_port = "31321"                # e.g. "32733"
dsn_protocol = "TCPIP"            # i.e. "TCPIP"
dsn_security = "SSL"              #i.e. "SSL"
```

In [ ]:
```
#TASK3: CREATE THE DATABASE CONNECTION
#IBM_DB API USES THE IBM DATA SERVER DRIVE FOR ODBC AND CLI APIS TO CONNECT TO
#CREATE THE DATABASE CONNECTION
```

In [12]:
```
dsn = (
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, dsn_port,

#print the connection string to check correct values are specified
print(dsn)
```

```
DRIVER={IBM DB2 ODBC DRIVER};DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-8fc0-
d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=31321;PROTOC
OL=TCPIP;UID=dmq73837;PWD=A6ZZPRoaOUUNvI1J;SECURITY=SSL;
```

In [18]:
```
try:
    conn = ibm_db.connect(dsn, "", "")
    print ("Connected to database: ", dsn_database, "as user: ", dsn_uid, "on

except:
    print ("Unable to connect: ", ibm_db.conn_errormsg() )
```

```
Connected to database:  bludb as user:  dmq73837 on host:  ba99a9e6-d59e-4883
-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud
```

In [19]:
```
server = ibm_db.server_info(conn)
```

In [20]:
```
print ("DBMS_NAME: ", server.DBMS_NAME)
print ("DBMS_VER:  ", server.DBMS_VER)
print ("DB_NAME:   ", server.DB_NAME)
```

```
DBMS_NAME:  DB2/LINUXX8664
DBMS_VER:   11.05.0700
DB_NAME:    BLUDB
```

In [16]:
```
print ("DRIVER_NAME:          ", client.DRIVER_NAME)
print ("DRIVER_VER:           ", client.DRIVER_VER)
print ("DATA_SOURCE_NAME:     ", client.DATA_SOURCE_NAME)
print ("DRIVER_ODBC_VER:      ", client.DRIVER_ODBC_VER)
print ("ODBC_VER:             ", client.ODBC_VER)
print ("ODBC_SQL_CONFORMANCE: ", client.ODBC_SQL_CONFORMANCE)
print ("APPL_CODEPAGE:        ", client.APPL_CODEPAGE)
print ("CONN_CODEPAGE:        ", client.CONN_CODEPAGE)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [16], in <cell line: 1>()
----> 1 print ("DRIVER_NAME:          ", client.DRIVER_NAME)
      2 print ("DRIVER_VER:           ", client.DRIVER_VER)
      3 print ("DATA_SOURCE_NAME:     ", client.DATA_SOURCE_NAME)

NameError: name 'client' is not defined
```

In [17]:
```
ibm_db.close(conn)
```

Out[17]:
```
True
```

In [24]:
```
#TASK 4: CREATE A TABLE IN THE DATABASE
#IN THIS STEP WE WILL CREATE A TABLE IN THE DATABAE WITH FOLLOWING DETAILS
```

In [25]:
```
dropQuery = "drop table INSTRUCTOR"
```

In [27]:
```
dropStmt = ibm_db.exec_immediate(conn, dropQuery)
```

```
---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
Input In [27], in <cell line: 1>()
----> 1 dropStmt = ibm_db.exec_immediate(conn, dropQuery)

 SQLCODE=-204ion: [IBM][CLI Driver][DB2/LINUXX8664] SQL0204N  "DMQ73837.INSTR
UCTOR" is an undefined name.  SQLSTATE=42704
```

In [28]:
```
#Construct the Create Table DDL statement - replace the ... with rest of the s
createQuery = "create table INSTRUCTOR(id INTEGER PRIMARY KEY NOT NULL, f
```

In [29]:
```
createStmt = ibm_db.replace_with_name_of_execution_method(conn, createQuery)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Input In [29], in <cell line: 1>()
----> 1 createStmt = ibm_db.replace_with_name_of_execution_method(conn, creat
eQuery)

AttributeError: module 'ibm_db' has no attribute 'replace_with_name_of_execut
ion_method'
```

In [30]:
```
insertQuery = "..."
```

In [ ]:
```
#TASK 5: INSERT DATA INTO THE TALE
#INTHIS STEP WE WILL INSERT SOME ROWS OF DATA INTO THE TABLE
```

In [32]:
```
createQuery = "create table INSTRUCTOR(ID INTEGER PRIMARY KEY NOT NULL, FNAME 

createStmt = ibm_db.exec_immediate(conn,createQuery)
```

In [33]:
```
insertQuery = "insert into INSTRUCTOR values (1, 'Rav', 'Ahuja', 'TORONTO', 'C

insertStmt = ibm_db.exec_immediate(conn, insertQuery)
```

In [35]:
```
#replace ... with the insert statement that inerts the remaining two rows of d
insertQuery2 = "..."
```

In [37]:
```
insertQuery2 = "insert into INSTRUCTOR values (2, 'Raul', 'Chong', 'Markham', 

insertStmt2 = ibm_db.exec_immediate(conn, insertQuery2)
```

In [ ]:
```
# TASK 6
#QUERY DATA IN THE TABLE
#IN THIS STEP WE FILL RETRIEVE DATA WE INSERT INTO THE INSTRUCTOR TABLE
```

In [38]:
```
selectQuery = "select * from INSTRUCTOR"
```

In [39]:
```
selectStmt = ibm_db.exec_immediate(conn, selectQuery)
```

In [40]:
```
while ibm_db.fetch_row(selectStmt) != False:
    print (" ID:",  ibm_db.result(selectStmt, 0), " FNAME:",  ibm_db.result(sel
```

```
 ID: 1  FNAME: Rav
 ID: 2  FNAME: Raul
 ID: 3  FNAME: Hima
```

In [41]:
```
#Construct the query that retrieves all rows from the INSTRUCTOR table
selectQuery = "select * from INSTRUCTOR"

#Execute the statement
selectStmt = ibm_db.exec_immediate(conn, selectQuery)

#Fetch the Dictionary (for the first row only)
ibm_db.fetch_both(selectStmt)
```

Out[41]:
```
{'ID': 1,
 0: 1,
 'FNAME': 'Rav',
 1: 'Rav',
 'LNAME': 'Ahuja',
 2: 'Ahuja',
 'CITY': 'TORONTO',
 3: 'TORONTO',
 'CCODE': 'CA',
 4: 'CA'}
```

In [42]:
```
#Fetch the rest of the rows and print the ID and FNAME for those rows
while ibm_db.fetch_row(selectStmt) != False:
    print (" ID:",  ibm_db.result(selectStmt, 0), " FNAME:",  ibm_db.result(se
ID: 2  FNAME: Raul
```

In [43]:
```
updateQuery = "update INSTRUCTOR set CITY='MOOSETOWN' where FNAME='Rav'"
```

In [44]:

In [51]:
```
import pandas
```

In [52]:
```
#connection for pandas
```

In [55]:
```
#query statement to retrieve all rows in INSTRUCTOR table
```

In [57]:
```
#retrieve the query results into a pandas dataframe
```

```
C:\Users\Edson\anaconda3\lib\site-packages\pandas\io\sql.py:761: UserWarning:
pandas only support SQLAlchemy connectable(engine/connection) ordatabase stri
ng URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, pleas
e consider using SQLAlchemy
  warnings.warn(
```

In [58]:
```
#print just the LNAME for first row in the pandas data frame
```

Out[58]:
```
'Ahuja'
```

In [59]:

Out[59]:

	ID	FNAME	LNAME	CITY	CCODE
0	1	Rav	Ahuja	MOOSETOWN	CA
1	2	Raul	Chong	Markham	CA
2	3	Hima	Vasudevan	Chicago	US

In [ ]: