CREATE DATABASE

TEAM ID	**PNT2022TMID00795**
PROJECT NAME	**VirtualEye - LifeGuard for Swimming Pools to Detect Active Drowning**

Step 1: Before you begin:

Create a service instance and credentials

Python version requirement

You must have a current version of the Python programming language that is installed on your system.

1. Check that Python is installed by running the following command at a prompt:

```
python3 --version
```

2. Verify that you get a result similar to the following example:

```
Python 3.8.1
```

Python Client Library requirement

1. Check that the client library is installed successfully by running the following command at a prompt:

```
pip freeze
```

2. Inspect the list, looking for an IBM Cloudant entry similar to the following example:

```
cloudant==2.14.0
```

Step 2: Connecting to a service instance

You must connect to your service instance before you create a database.

1. Run these import statements to connect to the service instance.

```
from cloudant.client import Cloudant
from cloudant.error import CloudantException
from cloudant.result import Result, ResultByKey
```

2. Find `username`, `password`, and `URL` in your Classic service credentials and replace `serviceUsername`, `servicePassword`, and `serviceURL` in the following example.
3. Establish a connection to the service instance.

```
client = Cloudant(serviceUsername, servicePassword, url=serviceURL)
client.connect()
```

4. Now replace ACCOUNT_NAME and API_KEY with the values from your IAM API service credentials.

```
client = Cloudant.iam(ACCOUNT_NAME, API_KEY, connect=True)
```

Now, your Python application can access the service instance on IBM Cloud.

Step 3: Creating a database within the service instance

1. Create this instance by defining a variable in the Python application.

```
databaseName = "demoDB"
```

2. Create the database.

```
myDatabaseDemo = client.create_database(databaseName)
```

3. Verify that the database was created successfully.

```
if myDatabaseDemo.exists():
    print("'{0}' successfully created.\n".format(databaseName))
```

Step 4: Storing a small collection of data as documents within the database

You want to store a small, simple collection of data in the database. This data is used in other tutorials, like Using IBM Cloudant Query to find data.

1. Create sample data

```
sampleData = [[1, "one", "boiling", 100], [2, "two", "hot", 40], [3, "three", "hot", 75],
[4, "four", "hot", 97], [5, "five", "warm", 20], [6, "six", "cold", 10], [7, "seven",
"freezing", 0], [8, "eight", "freezing", -5]]
```

2. Use a `for` statement to retrieve the fields in each row by going through each row in the array.

```
for document in sampleData:
    number = document[0]
    name = document[1]
    description = document[2]
```

```
    temperature = document[3]
```

3. Create a JSON document that represents all the data in the row.

```
jsonDocument = { "numberField": number, "nameField": name, "descriptionField":
description, "temperatureField": temperature }
```

4. Create a document by using the Database API.

```
newDocument = myDatabaseDemo.create_document(jsonDocument)
```

5. Check that the document exists in the database.

```
if newDocument.exists():
    print("Document '{0}' successfully created.".format(number))
```