**Introduction to MongoDB and Python**

MongoDB is a open-source N0SQL database that is written in C++. A NoSQL database has a dynamic schema for unstructured data. In NoSQL, data is stored in several ways: it can be column-oriented, document-oriented, graph-based or organized as a key-value store. A NoSQL database has the following advantages:

* Documents can be created without having to first define their structure
* Each document can have its own unique structure
* The syntax can vary from database to database
* Large volumes of structured, semi-structured, and unstructured data
* Object-oriented programming that is easy to use and flexible
* It is horizontally scalable

In PyMongo we use dictionaries to represent documents. Let's show an example of a PyMongo document below:

article = {"author": "Derrick Mwiti",

"about": "Introduction to MongoDB and Python",

"tags": ["mongodb", "python", "pymongo"]}

articles = db.articles

result = articles.insert\_one(article)

print("First article key is: {}".format(result.inserted\_id))

Add many articles

new\_articles = articles.insert\_many([article1, article2])

returns the first match that it comes across. If it is empty it gives up the first article.

print(articles.find\_one())

Find all articles in a collection we can use

for article in articles.find():

print(article)

Sometimes we might not want to return all the fields from our documents. Let's show we'd fetch specific fields. In our case we use 0 to specify that the \_id should not be fetched and 1 to specify that author and about should be fetched. MongoDB doesn't allow us to specify zero twice. For example, specify tags to 0 below will generate an error. We are not allowed to specify both 0 and 1 values in the same object (unless one of the fields is the \_id field). When we specify a field with the value 0, all other fields get the value 1.

for article in articles.find({},{ "\_id": 0, "author": 1, "about": 1}):

print(article)