**MongoDB Practical Lab**

1. Issue the following command to use the MongoLab database
   * use MongoLab



1. Insert the following document into the People collection
   * name: Joe Bloggs
   * age: 21

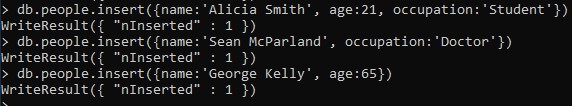


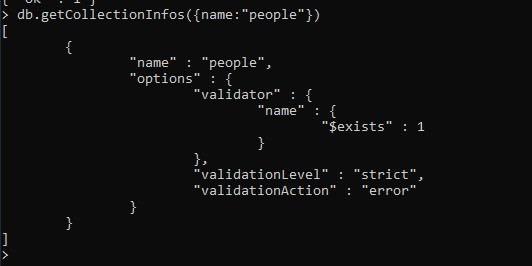
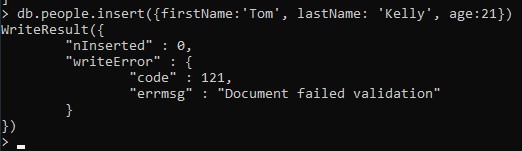
1. Insert the following document into the People collection
   * name: Tom Kelly
   * age: 21  
       
     
2. Display all the documents in the People collection  
     
   
3. Insert the following document into the People collection
   * name: Joe Whyte
   * age: 23



1. Display the documents with age = 21   
     
   
2. Insert the following document into the People collection
   * name: Jill Finnerty
   * age: 30
   * Occupation:Teacher  
     
3. Explain why MongoDB is described as a schema free database. Insert 3 more documents into the People collection to illustrate your answer

MongoDB is considered to be schema-less because it doesn’t require a rigid, predefined data structure. This database model employs a dynamic schema using collections and indexes.

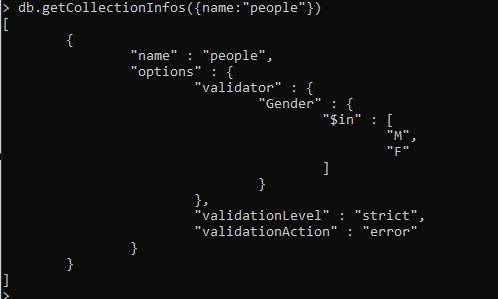
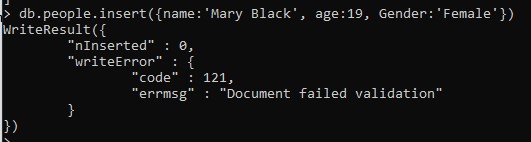
As can be seen from the below screenshot, data inserted into a collection does not need to format the same structure, the first insertion has three attributes; name, age and occupation. The second and third records only contain data for two of the attributes but are still inserted into the database with no issue.  
  


1. Count the number of documents in the People collection  
     
   
2. Update Tom Kelly’s age to 22  
     
   
3. Run the following command to verify that the People collection has **no** validation rules set
   * db.getCollectionInfos({name:"People"})  
       
     
4. Run a command (collMod) to add a rule to ensure that an insert will fail if you don’t include a name attribute  
     
   
5. Verify that your validation rule has been added using the following command
   * db.getCollectionInfos({name:"People"})  
       
     
6. Try to insert a person with the following details
   * FirstName: Tom
   * LastName: Kelly
   * age: 21  
       
     
7. Use the following command to add a gender field with the value ‘M’ to all documents in the People collection.

db.People.update({}, {$set: {"Gender": 'M'}}, false, true)

(See slide on “Adding a new field to all documents in a collection” for details of this command)

Then display all the documents in the People collection.  
  


1. Insert a person with the following details
   * name: Mary Kelly
   * age: 19
   * gender: ‘F’  
       
     
2. Run a command to ensure that each document that is inserted or updated should have the gender field set to ‘M’ or ‘F’.  
     
   
3. Verify that your validation rule has been added using the following command
   * db.getCollectionInfos({name:"People"})  
       
     
4. Try to insert a person with the following details
   * name: Mary Black
   * age: 19
   * gender: “Female”  
       
     
5. Briefly discuss the advantages of validating documents in MongoDB  
     
   MongoDB provides the capability to validate documents during updates and insertions, as well as specifying validation rules on a per-collection basis. This provides a lot of flexibility to the user in terms of which parts of documents are and are not validated for any given collection.