FIT9132 Introduction to Databases

Week 12 Tutorial Activities

Big Data and NO SQL

FIT Database Teaching Team

Complete the week 12 activities listed below

FIT9132 2021 S2

FIT9132 Introduction to Databases

Author: FIT Database Teaching Team

Important

Remember before starting any lab activity which involves working with files, first use SQLDeveloper to pull from the FIT GitLab server so as to ensure your local files and the FIT GitLab server files are in sync.

Learning Objectives:

- be able to appreciate other data model other than relational database
- be able to generate JSON formatted data using SQL Select command
- be able to use basic MongoDB commands for basic CRUD operations
- be able to compare the MongoDB queries with their equivalent SQL

12.1 MongoDB

12.1.1 Class Discussion

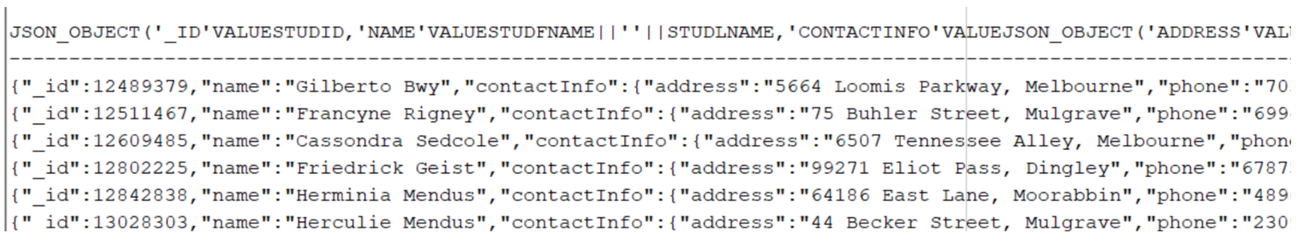
In this tutorial we will focus on MongoDB, one of the popular Big Data platforms. The data in MongoDB must be stored in a specific format, for example:

```
{
    "_id": 12489379,
    "name": "Gilberto Bwy",
    "contactInfo": {
      "address": "5664 Loomis Parkway, Melbourne",
      "phone": "7037621034",
      "email": "Gilberto.Bwy@student.monash.edu"
    },
    "enrolmentInfo": [
      {
        "unitcode": "FIT1045",
        "year": "2019",
        "semester": 1,
        "mark": 40,
        "grade": "N"
      },
      {
        "unitcode": "FIT2094",
        "year": "2020",
        "semester": 1,
        "mark": 63,
        "grade": "C"
      },
      {
        "unitcode": "FIT1050",
        "year": "2019",
        "semester": 2,
        "mark": 92,
        "grade": "HD"
      },
      {
        "unitcode": "FIT1045",
        "year": "2019",
        "semester": 2,
        "mark": 89,
        "grade": "HD"
      },
      {
        "unitcode": "FIT1050",
        "year": "2019",
        "semester": 1,
        "mark": 44,
        "grade": "N"
      }
    ]
  }
```

We can generate JSON structures from relational database data and use the JSON output to create suitable documents in MongoDB. To generate JSON in Oracle SQL use:

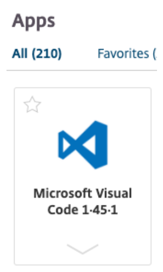
```
SET PAGESIZE 75

SELECT
    JSON_OBJECT ( '_id' VALUE studid, 'name' VALUE studfname
                || ' '
                || studlname,
                'contactInfo' VALUE JSON_OBJECT (
                    'address' VALUE studaddress,
                    'phone'   VALUE rtrim(studphone),
                    'email' VALUE studemail
                    ),
                'enrolmentInfo' VALUE JSON_ARRAYAGG(
                    JSON_OBJECT(
                        'unitcode' VALUE unitcode,
                        'year' VALUE to_char(ofyear, 'yyyy'),
                        'semester' VALUE semester,
                        'mark' VALUE mark,
                        'grade' VALUE grade
                        )
                    ) FORMAT JSON )
    || ','
FROM
    uni.student
    NATURAL JOIN uni.enrolment
GROUP BY
    studid,
    studfname,
    studlname,
    studaddress,
    studphone,
    studemail
ORDER BY
    studid;
```

This will produce output of the form:



```
JSON_OBJECT('_ID'VALUESTUDID,'NAME'VALUESTUDFNAME||''||STUDLNAME,'CONTACTINFO'VALUEJSON_OBJECT('ADDRESS'VAL
------------------------------------------------------------------------------------------------------------
{"_id":12489379,"name":"Gilberto Bwy","contactInfo":{"address":"5664 Loomis Parkway, Melbourne","phone":"70
{"_id":12511467,"name":"Francyne Rigney","contactInfo":{"address":"75 Buhler Street, Mulgrave","phone":"699
{"_id":12609485,"name":"Cassondra Sedcole","contactInfo":{"address":"6507 Tennessee Alley, Melbourne","phon
{"_id":12802225,"name":"Friedrick Geist","contactInfo":{"address":"99271 Eliot Pass, Dingley","phone":"6787
{"_id":12842838,"name":"Herminia Mendus","contactInfo":{"address":"64186 East Lane, Moorabbin","phone":"489
{"_id":13028303,"name":"Herculie Mendus","contactInfo":{"address":"44 Becker Street, Mulgrave","phone":"230
```

Open Microsoft Studio Visual Code on MoVE or locally if you have your own local version:



Copy the generated SQL output into Microsoft Studio Visual Code.

Make sure that you paste the text within square brackets **[paste your text here], remove the last comma symbol**, on the right bottom corner **change** the document type from **plain text to json,** then right click on your text and select **Format Document**. Save the file as student_enrolment.json in your local repo.

Open the MongoDB shell https://docs.mongodb.com/manual/tutorial/getting-started/, then click the working window to connect to the database.

Alternatively, you can install your own local copy of MongoDB by following the steps listed at: https://docs.mongodb.com/manual/administration/install-community/

Note for MacOS the install of the community edition occurs via brew and requires:

- A 64-bit Intel CPU or Apple Silicon CPU
- macOS Mojave (10.14) (or higher)
- Command Line Tools (CLT) for Xcode:
    - xcode-select --install
    - or install the full Xcode via the App Store (note this is a large download (11Gb) and will take some time)
- A Bourne-compatible shell for installation (e.g. bash or zsh)

For MS Windows you need to download, run the msi installer and set up MongoDB as part of the installation.

After you have connected to MongoDB locally or via the web shell, type

```
use results
```

and hit enter. You are now using the results database on MongoDB.

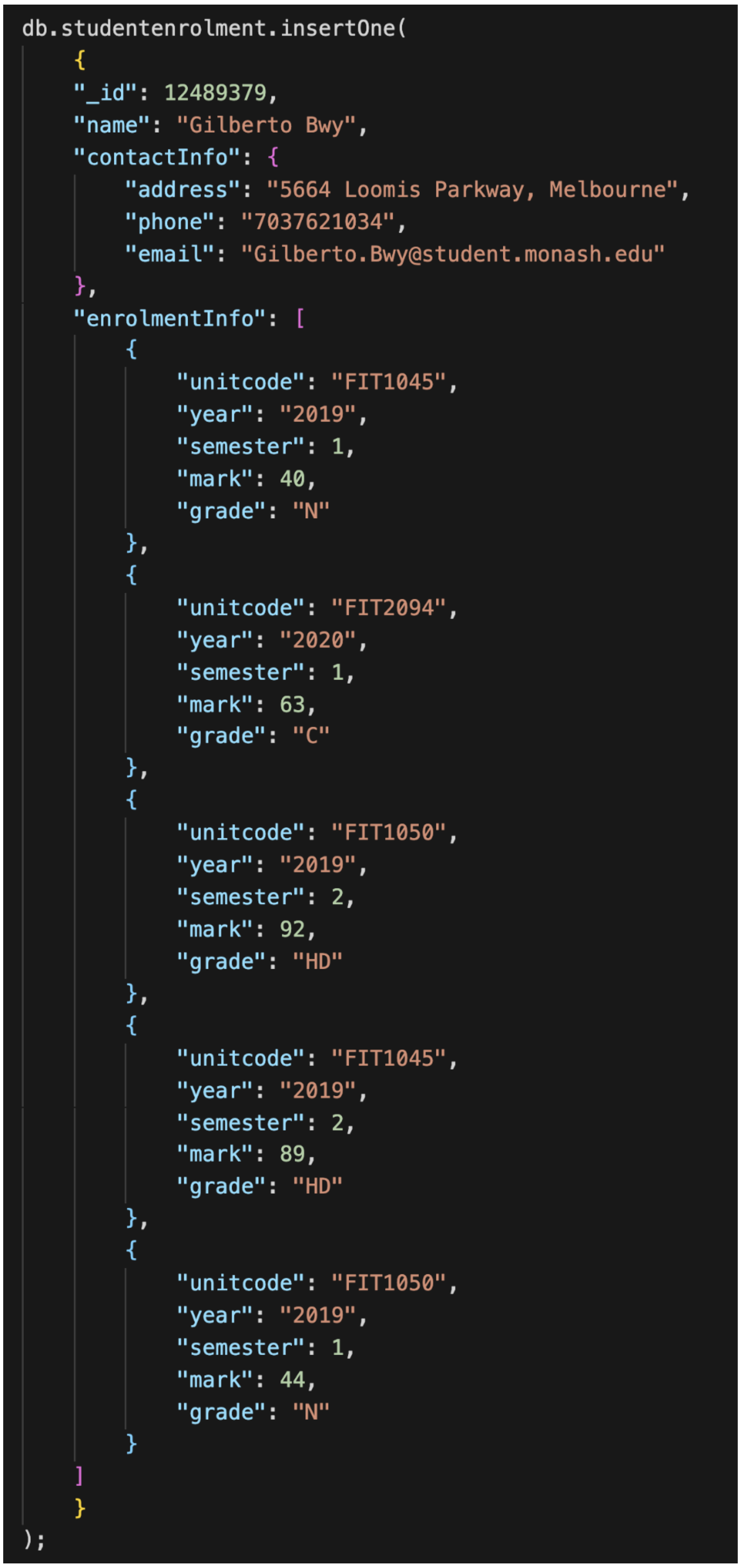
A MongoDB database is a set of collections, and a collection is a set of documents. To show collections in a database, type

```
show collections
```

and hit enter. At the moment, there is no collection in the results database.

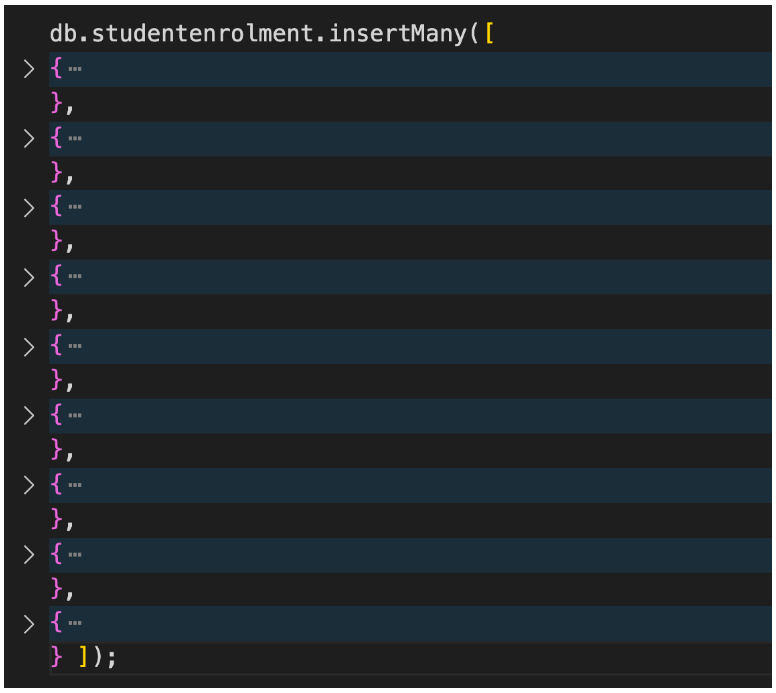
Now, let’s create, read, update and delete (CRUD) documents on MongoDB

1. Create a collection by inserting one document into the collection. For this activity, we will call the collection studentenrolment



```
db.studentenrolment.insertOne(
    {
    "_id": 12489379,
    "name": "Gilberto Bwy",
    "contactInfo": {
        "address": "5664 Loomis Parkway, Melbourne",
        "phone": "7037621034",
        "email": "Gilberto.Bwy@student.monash.edu"
    },
    "enrolmentInfo": [
        {
            "unitcode": "FIT1045",
            "year": "2019",
            "semester": 1,
            "mark": 40,
            "grade": "N"
        },
        {
            "unitcode": "FIT2094",
            "year": "2020",
            "semester": 1,
            "mark": 63,
            "grade": "C"
        },
        {
            "unitcode": "FIT1050",
            "year": "2019",
            "semester": 2,
            "mark": 92,
            "grade": "HD"
        },
        {
            "unitcode": "FIT1045",
            "year": "2019",
            "semester": 2,
            "mark": 89,
            "grade": "HD"
        },
        {
            "unitcode": "FIT1050",
            "year": "2019",
            "semester": 1,
            "mark": 44,
            "grade": "N"
        }
    ]
    }
);
```

2. Insert 9 extra documents into the collection using (documents shown collapsed)



```
db.studentenrolment.insertMany([
{…
},
{…
},
{…
},
{…
},
{…
},
{…
},
{…
},
{…
},
{…
} ]);
```

3. Read data

   a. Check how many documents have been inserted into the collection

   ```
   > db.studentenrolment.find().count();
   ```

   b. Read all documents

   ```
   > db.studentenrolment.find();
   ```

   ```
   > db.studentenrolment.find().pretty();
   ```

   c. Show the data for student id = 13119134

   ```
   > db.studentenrolment.find({"_id":13119134}).pretty();
   ```

   d. Show id and name of students who were enrolled in FIT3157

   ```
   > db.studentenrolment.find(
   ...     {"enrolmentInfo.unitcode":"FIT3157"},
   ...     {"_id":1,"name":1}
   [... );
   ```

   e. Show the name and address of students who live in Mulgrave or Moorabbin

```
> db.studentenrolment.find(
...     {$or:[{"contactInfo.address":/.*Mulgrave.*/},{"contactInfo.address":/.*Moorabbin.*/}]},
...     {"_id":0,"name":1,"contactInfo.address":1}
... );
```

4. Add/remove data from an array

   a. Show the data for student id = 13119134

```
> db.studentenrolment.find({"_id":13119134}).pretty();
```

   b. Add a new enrolment for student id = 13119134, the student was enrolled in FIT3074 in semester 1 2020. Set the mark and grade as null.

```
> db.studentenrolment.update(
...     {"_id":13119134},
...     {$push:{"enrolmentInfo":{"unitcode":"FIT3074","year":"2020","semester":1,"mark":null,"grade":null}}}
... );
```

   c. Check if the data is correctly inserted

```
> db.studentenrolment.find({"_id":13119134}).pretty();
```

   d. Remove the enrolment data for student id = 13119134 in FIT3074 for semester 1 2020.

```
> db.studentenrolment.update(
...     {"_id":13119134},
...     {$pull:{"enrolmentInfo":{"unitcode":"FIT3074","year":"2020","semester":1}}}
... );
```

   e. Check if the data is correctly removed

```
> db.studentenrolment.find({"_id":13119134}).pretty();
```

5. Update Value

   Update student id 12609485's name from Cassondra Sedcole to Cassondra Williams. She also changed her phone number to 0412999999

```
> db.studentenrolment.find({"_id":12609485}).pretty();
```

```
> db.studentenrolment.updateOne(
...     {"_id":12609485},
...     {$set:{"name" : "Cassondra Williams","contactInfo.phone":"0412999999"}}
... );
```

```
> db.studentenrolment.find({"_id":12609485}).pretty();
```

6. Delete all details (document) of student id 12489379

```
> db.studentenrolment.deleteOne({"_id":12489379});
```

```
> db.studentenrolment.find().count();
```

12.1.2 MongoDB Create Update and Delete

First, download the week12_bigdata.txt file from Moodle. Write the necessary SQL statements and MongoDB scripts for **12.1.2 and 12.1.3** questions in that file.

1. Write an SQL select statement to generate a collection of documents using the following structure/format from the UNI database.

```
{
    "_id": 12489379,
    "name": "Gilberto Bwy",
    "contactInfo": {
      "address": "5664 Loomis Parkway, Melbourne",
      "phone": "7037621034",
      "email": "Gilberto.Bwy@student.monash.edu"
    },
    "dob": "30-08-1992",
    "enrolmentInfo": [
      {
        "unitcode": "FIT1045",
        "unitname": "Algorithms and programming fundamentals in python",
        "year": "2019",
        "semester": 1,
        "mark": 40,
        "grade": "N"
      },
      {
        "unitcode": "FIT2094",
        "unitname": "Databases",
        "year": "2020",
        "semester": 1,
        "mark": 63,
        "grade": "C"
      },
      {
        "unitcode": "FIT1050",
        "unitname": "Web fundamentals",
        "year": "2019",
        "semester": 2,
        "mark": 92,
        "grade": "HD"
      },
      {
        "unitcode": "FIT1045",
        "unitname": "Algorithms and programming fundamentals in python",
        "year": "2019",
        "semester": 2,
        "mark": 89,
        "grade": "HD"
      },
      {
        "unitcode": "FIT1050",
        "unitname": "Web fundamentals",
        "year": "2019",
        "semester": 1,
        "mark": 44,
        "grade": "N"
      }
    ]
  }
```

2. Name the collection as **enrolment** (you may use any suitable name for your database) and insert the first 10 documents generated by the select statement into MongoDB (*if it returns an error, try to add a maximum 5 documents at one time*).
3. Create a new enrolment for studid 12489379, the student is enrolled in FIT2002 (IT Project Management) in semester 1 2020. Since this is a new enrolment, set the mark and the grade as null.
4. Update this enrolment for studid 12489379 in FIT2002, set the mark to 65 and grade to C
5. Delete this enrolment for student id 12489379 in FIT2002

12.1.3 MongoDB Read

Write db.find() commands for following questions:

1. Retrieve the document for student id = 12802225
2. Show the id and name of students who have any mark greater than 95 in any enrolment (hint: use $gt:95)
3. Retrieve the name and contact info of students who enrolled in any unit which has "web design" as part of its name
4. Retrieve the id and name of any students who have grades WH or N

12.2 SETU

During the remainder of this tutorial:

- please complete your SETU for this unit (it provides important feedback to the teaching staff)

Important

You need to get into the habit of establishing this as a standard FIT9132 workflow - Pull at the start of your working session, work on the activities you wish to/are able to complete during this session, add (stage), commit changes and then Push the changes back to the FIT GitLab server