MODUL 4
TIPE DATA DAN RELASI PADA MONGODB

4.1. Tujuan Praktikum

1. Mengenal tipe data pada MongoDB
2. Mengenal relasi pada MongoDB
3. Mengenal Lookup

4.2. Dasar Teori

4.2.1. Tipe Data MongoDB

Document didalam MongoDB dapat dianggap sebagai "JSON" karena secara konseptual mirip dengan object pada JavaScript. Perlu untuk diketahui bahwa JSON memiliki keterbatasan dalam hal dukungan tipe data yang diantaranya hanyalah: null, boolean, numeric, string, array, dan object. Dalam hal ini, MongoDB tersedia dengan dukungan tambahan terhadap tipe data dengan tetap membawa sifat dasar dari JSON itu sendiri.

Mongodb mempunyai beberapa tipe data dalam penyimpanannya. Tipe – tipe datanya sebagai berikut :

1. String

   Tipe data string adalah tipe data yang digunakan untuk menyimpan sebuah kata – kata pada database. Contohnya : { nama : “Basis Data Lanjut” }

2. Int

   Tipe data int atau integer adalah tipe data yang digunakan untuk menyimpan sebuah angka pada database. Disini untuk integer di bagi ke menjadi 2 yaitu int dan number long, perbedaannya adalah dari segi banyaknya penyimpanan untuk int bisa menampung data kurang lebih "2.000.000.000" sedangkan untuk number long bisa menampung data kurang lebih "9.000.000.000.000.000.000". Contohnya : { harga : 1000000 }. (untuk angka tidak menggunakan kutip 2, yang menggunakan kutip 2 hanya untuk tipe data string. Untuk tipe data ini juga jangan menulis menggunakan titik.

3. Decimal

   Tipe data decimal adalah tipe data yang digunakan untuk menyimpan sebuah angka pecahan pada database. Contohnya : { nilai : 50.5 }. *(untuk koma disini menggunakan titik, bukan menggunakan koma).*

4. Boolean

   Tipe data boolean adalah tipe data yang hanya mengenal “true” dan “false” atau benar dan salah saja. Contohnya { status_aktif : true } *(untuk penulisan true atau false disini juga tidak memakai kutip 2)*

5. Date

   Tipe data date adalah tipe data yang digunakan untuk menyimpan sebuah data tanggal pada database. Contohnya { tanggal_masuk : ISO(“2018-11-20”) } untuk tanggal yang sudah di tetapkan atau kalau ingin menyimpan tanggal sekarang bisa menggunakan new date() contohnya { hari_ini : new Date() }.

6. Array

   Tipe data array adalah tipe data yang digunakan untuk menyimpan beberapa data dalam 1 kolom pada database. Contohnya { hobi : [“sepak bola”,”renang”] }. Array bisa digunakan untuk banyak hal salah satunya adalah untuk menyimpan daftar data untuk sebuah field seperti yang telah di contohkan.

7. Object

   Tipe data object adalah tipe data yang digunakan untuk menyimpan beberapa data dalam 1 kolom pada database (hampir sama dengan array tetapi kalau array data yang ada di dalamnya kita bisa menyebutnya hobi[0] untuk baris pertama nah sedangkan untuk object ini kita bisa memberikan nama data pada setiap barisnya). Contohnya

```
{
    status : {
     status_aktif : true,
     status_lulus : false
    }
}
```

8. ObjectID

   Tipe data objectid adalah tipe data yang digunakan untuk menyimpan id pada database. Contohnya

```
{ _id : ObjectId(“4beffd2be5896hror92k123a”) }.
```

   Secara default MongoDB akan membuat ObjectID pada _id pada setiap data baru yang ditambahkan. ObjectID ini berupa 12-byte BSON (binary JSON) hexadesimal.

4 byte pertama pada ObjectID merupakan timestamp kapan waktu ObjectID itu dibuat dan 5 byte setelahnya adalah angka acak.

4.2.2. Relasi MongoDB

Relasi adalah suatu hubungan antar tabel pada database dimana suatu tabel mempunyai data yang sama dengan data yang ada di tabel lainnya. Terdapat 3 macam relasi yaitu :

1. Relasi one to one
   Hubungan antar tabel dimana tabel A adalah data master dan tabel B harus mempunyai data yang ada pada tabel A
2. Relasi one to many
   Sebuah hubungan antara tabel dimana tabel A memiliki sebuah data yang bisa di pakai pada tabel B data tersebut bisa banyak data atau beberapa data saja.
3. Relasi many to many
   Sebuah hubungan antara tabel dimana tabelnya ini ada banyak. Hubungannya itu bisa banyak tabel misalkan tabel A dengan tabel B dan tabel C dengan tabel B seperti itu.

4.2.3. Lookup

Lookup adalah sebuah metode yang digunakan untuk menampilkan data seperti find tetapi di lookup ini kita dapat menampilkan beberapa tabel untuk di tampilkan (lookup ini bisa berjalan kalau tabelnya sudah berrelasi). Secara sederhana Lookup merupakan cara kita untuk melakukan Join antar collections dan perinthan ini akan menambahkan array field baru dari setiap data dari collection lain yang ditambahkan berdasarkan kondisinya.

```
{ _id : ObjectId("4beffd2be5896hror92k123a") }.
```

```
db.collection.aggregate({
   $lookup : {
                from : "collection2",
                localField : "_id",
                foreignField : "_id",
                as : "join"
   }
})
```

Keterangan :

Aggregate : untuk mengkelompokan data

Lookup : untuk mengabungkan data

From : diisi tabel ke 2 yang akan di tampilkan

localField : diisi dengan nama data yang ada di tabel pertama (data ini harus data yang nanti sama dengan data yang ada di tabel ke 2)

foreignField : sama dengan localField tetapi untuk foreignField diisi dengan nama data yang ada di tabel kedua

as : as atau alias dapat diisi bebas karena disini jika nanti data kita mau di tampilkan kita cukup memanggil nama yang sudah di aliaskan saja.

4.2.3. Validasi

Validasi adalah sebuah metode untuk pengecekan suatu data yang di masukkan, misalkan pada saat kita login pada suatu website kita ketikan asal – asalan maka akan muncul peringatan.

Validasi bisa ditambahkan ketika kita sedang membuat collection contohnya:

```
db.createCollection (“students”, {
   $validator :

$jsonSchema : {
    bsonType: "object",
   title: "Student Object Validation",
 required: [ "alamat", "jurusan", "nama", "tahun" ],
properties: {
    nama: {
          bsonType: "string",
         description: "'nama' harus dalam rupa string dan wajib diisi"
    },
   tahun: {
         bsonType: "int",
        minimum: 2016,
       maximum: 3016,
      description: " tahun harus berupa angka dan tidak kurang dari 2016 dan tidak lebih
dari 3016"
  },
 ipk: {
    bsonType: [ "double" ],
   description: "'ipk' harus dalam bentuk tipe data double apabila ada pada sebuah
document"
}

       }
}})
```

Validasi diatas akan memastikan apabila kita akan menambahkan data ke collection students maka jika tidak memiliki field alamat, jurusan, nama serta tahun maka akan muncul error dan data tidak akan masuk dan juga ketika field nama tidak di isi atau tidak dalam bentuk string akan terjadi error juga begitu pula denga field tahun harus dalam bentuk integer dan ipk harus dalam bentuk double.

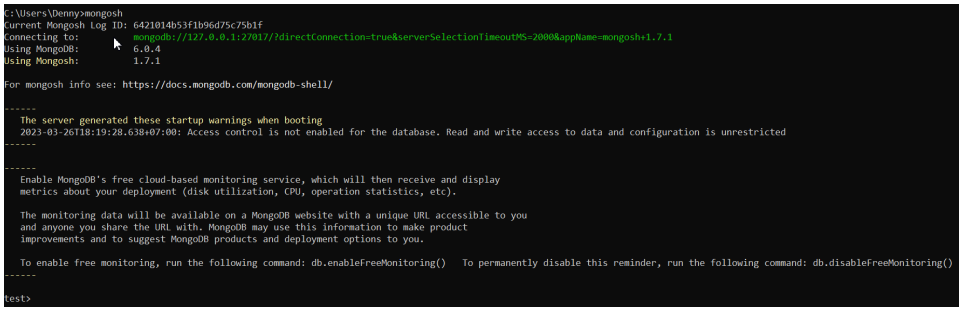
4.3. *Software*

1. MongoDB
2. Compass
3. MongoDB Shell

4.4. Tahapan Kerja

4.4.1. Tipe Data

1. Buka MongoDB shell dengan melakukan command “mongosh” pada cmd.



2. Buat Database baru dengan nama bebas menggunakan query use.

```
test> use Modul3
switched to db Modul3
Modul3>
```

3. Buat collection baru.

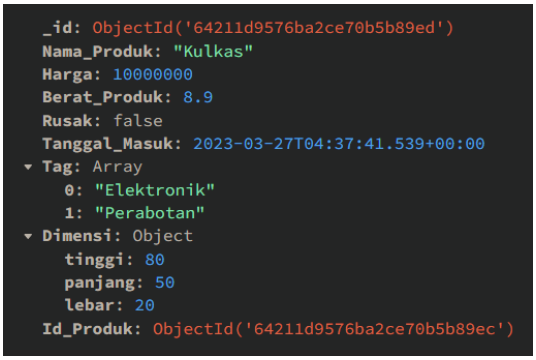
```
Modul3> db.createCollection("TipeData")
{ ok:1 }
Modul3>
```

4. Masukkan document baru dengan setiap tipe data sesuai dari bagian dasar teori.

```
Modul3> db.TipeData.insertOne(
       {
         Nama_Produk  : "Kulkas",
         Harga : 10000000,
         Berat_Produk : 8.9,
         Rusak : false,
         Tanggal_Masuk : new Date(),
         Tag : ["Elektronik", "Perabotan"],
         Dimensi : {
           tinggi : 80,
           panjang : 50,
           lebar : 20
         },
         Id_Product : ObjectId()
       }
      )
```

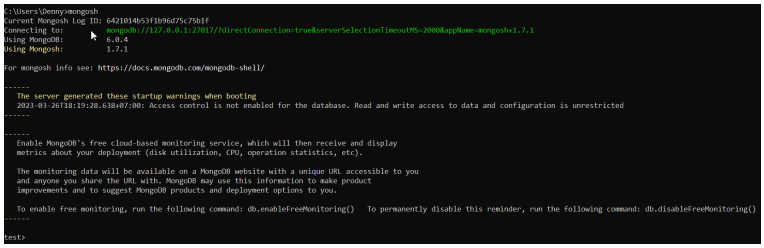
Query diatas menggunakan semua data type yang sudah diterangkan pada dasar teori :

- Nama_Produk menggunakan String
- Harga Menggunakan Integer
- Berat_Produk menggunakan Decimal
- Rusak menggunakan Boolean
- Tanggal Masuk menggunakan Date
- Tag menggunakan Array untuk menyimpan banyak nilai sekaligus
- Dimensi menggunakan Object untuk menyimpan banyak data
- Id_Produk menggunakan Object id untuk membuat id yang unik.



4.4.2. Lookup

1. Buka CMD lalu ketik mongosh untuk mengakses mongoDB shell.



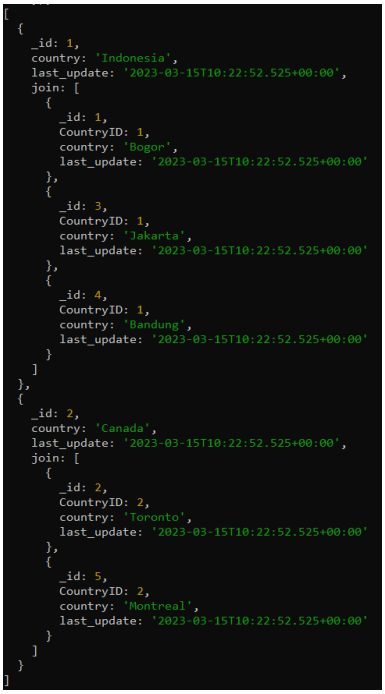
2. Lalu ganti database yang dipilih ke database rentalfilm yang telah dibuat sebelumnya dengan menggunakan query “use”.

```
test> use rentalfilm
switched to db rentalfilm
rentalfilm>
```

3. Disini kita akan melakukan query lookup sederhana dimana kita akan mengambil data “Country” beserta data “City” dengan melakukan join.

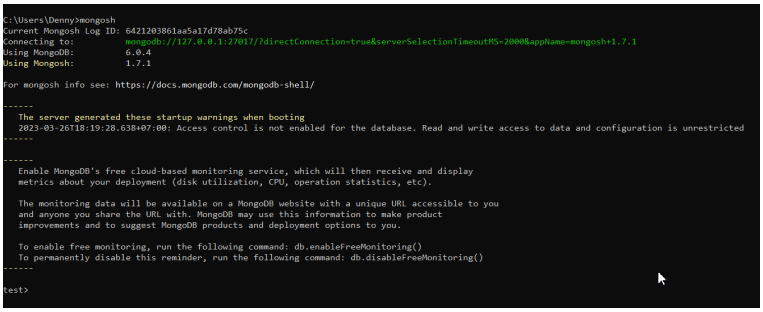
```
rentalfilm>db.country.aggregate({
   $lookup:{
        "city"
        localField: id",
        foreignField: "CountryID",
        as :"join"
   }
});
```

4. Query diatas akan menampilkan data dari collection Country beserta data city yang memiliki relasi berdasarkan field Country ID.



4.4.3. Validasi

1. Buka kembali mongo shell dengan perintah mongosh pada cmd.



2. Lalu pilih lagi database rental film.

```
rentalfilm>db.country.aggregate({
   $lookup:{
        "city"
        localField: id",
        foreignField: "CountryID",
        as :"join"
   }
});
```

```
test> use rentalfilm
switched to db rentalfilm
rentalfilm>
```

3. Buat collection baru dengan nama continent beserta validator

```
rentalfilm>db.createCollection("continent", {
   validator:{
        $jsonSchema: {
        bsonType: "object",
        title: "continent validator",
        required: ["continent", "last_update"],
        properties: {
           continent: {
              bsonType: "string",
              description: "'continent" harus dalam rupa string dan wajib
diisi"
           },
        }
    }
  }
})
{ pk: 1 }
Rentalfilm>
```

4. Lakukan testing validator dengan memasukan data yang tidak valid.

```
rentalfilm>db.continent.insertOne({cont : "Asia", last_update : ISOOate()})
```

```
rentalfilm> db.continent.insertOne({cont : "Asia", last_update : ISODate()})
Uncaught:
MongoServerError: Document failed validation
Additional information: {
  failingDocumentId: ObjectId("64210dfc76ba2ce70b5b89e6"),
  details: {
    operatorName: '$jsonSchema',
    title: 'continent validator',
    schemaRulesNotSatisfied: [
      {
        operatorName: 'required',
        specifiedAs: { required: [ 'continent', 'last_update' ] },
        missingProperties: [ 'continent' ]
      }
    ]
  }
}
```

5. Lakukan testing validator dengan memasukan data yang valid

```
rentalfilm> db.continent,insertOne({continent : "Asia", last_update :
ISODate()})
{
  Acknowledged: true,
  insertedID: ObjectId("64210e2476ba2ce705b89e7")
```

4.5. Tugas dan Latihan

1. Buatlah serta jalankan query untuk membuat lebih dari 1 document baru yang menggunakan semua jenis tipe data.
2. Buatlah collection baru serta tambahkan validator dengan minimal 2 field.
3. Buat dan jalankan query untuk menambahkan data yang valid kedalam collection yang sudah ada validatornya.
4. Buat dan jalankan query untuk menambahkan data yang tidak valid kedalam collection yang sudah ada validatornya.
5. Buat dan jalankan dua buah query lookup pada database rentalfilm yang telah dibuat selain pada collection country dan city.