LAPORAN PRAKTIKUM
PRAKTIKUM DATABASE PERTEMUAN 6

Disusun untuk Memenuhi Matakuliah Praktikum {Nama Matakuliah}
Dibimbing oleh : Sulaibatul Aslamiyah, M.Kom

Oleh:

RAFLI RAHMAN.EFENDY

1124102162

PROGRAM STUDI S1 TEKNIK INFORMATIKA
SEKOLAH TINGGI ILMU KOMPUTER PGRI BANYUWANGI
2025

LEMBAR PENGESAHAN

LAPORAN PRAKTIKUM

Matakuliah : Praktikum Database

Oleh : Rafli Rahman.Efendy

NIM : 1124102162

Telah disahkan pada

Hari : SELASA

Tanggal : 27/05/2025

Mengetahui/ Menyetujui :

Dosen Pengampu Mata Kuliah

Sulaibatul Aslamiyah, M.Kom

NIDN. 0712058304

KATA PENGANTAR

Berisikan kata pengantar yang dituliskan oleh mahasiswa untuk laporan praktikum matakuliah yang diikuti.

MODUL PRAKTIKUM 1

1.1 Capaian Praktikum Pertemuan 1

Mahasiswa dapat mencatat hasil praktikum dengan baik dan membuat laporan praktikum yang mencakup tujuan, alat, bahan, prosedur, dan hasil praktikum.

1.2 Indikator Capaian

Mahasiswa mampu menerapkan perintah Data Definition Language atau CRUD (Create, Read, Update, Delete) document pada database non relational.

1.3 Landasan Teori

Databases adalah kumpulan data yang terstruktur dan disimpan secara terorganisir untuk memudahkan pengelolaan dan pencarian informasi

1.4 Pelaksanaan Praktikum

Model data embedded cocok untuk digunakan ketika : Antar document saling ketergantungan Kita jarang melakukan perubahan isi data pada embedded document Embedded document selalu dibutuhkan ketika mengambil data document

Pendahuluan

Pada materi ini akan membahas interaksi dengan data/document pada database MongoDB, seperti menambah data dalam database, mengupdate data, menghapus data serta membaca data dalam database. MongoDB menyediakan aturan perintah dasar untuk operasi CRUD

1. **Praktikum**

Create Operations

Operasi pertama pada CRUD Operation adalah Create. Perintah untuk membuat dokumen baru pada MongoDB ada 2 :

- db.collection.insertOne( )
- db.collection.insertMany( )

db.collection.insertOne()

Perintah insertOne() berfungsi untuk menambah document tunggal ke sebuah collection. Collection harus dideklarasikan (dibuat secara implisit) di MongoDB, oleh karena itu jika collection belum ada maka operasi insert akan secara otomatis membuatnya.

```
biodata> db.createCollection("BooksDB")
{ ok: 1 }
biodata> db.BooksDB.insertOne({
...     title: "Dead Silence",
...     author: "S.A. Barnes",
...     isbn: 1250819997,
...     price: 13.99,
...     available: true
... })
...
{
  acknowledged: true,
  insertedId: ObjectId('68366e0399789bd225b7123b')
}
biodata>
```

```
db.BooksDB.insertOne({
   title: "Dead Silence",
   author: "S.A. Barnes",
   isbn: 1250819997,
   price: 13.99,
   available: true
})
```

Penjelasan Code :

- **db** Objek utama MongoDB yang mewakili database saat ini.
- **BooksDB** Nama koleksi (collection) tempat data buku disimpan. Jika belum ada, koleksi ini akan dibuat otomatis.
- **insertOne()** Fungsi untuk menambahkan satu dokumen (data) ke dalam koleksi.
- **{ ... }** Objek dokumen yang akan dimasukkan. Ini berisi data tentang buku.

db.collection.insertMany()

Perintah insertMany() digunakan untuk menambah lebih dari satu document sekaligus dalam satu perintah. Untuk menambahkan data dengan insertMany(), data haruslah berada dalam sebuah array yang ditandai dengan kurung siku ([ ]), dan sebagai pemisah antar data digunakan ymbol koma (,).

```
biodata> db.BooksDB.insertMany([
...   {
...     title: "Project Hail Mary",
...     author: "Andy Weir",
...     isbn: 0593135202,
...     price: 18.99,
...     available: true
...   },
...   {
...     title: "The Midnight Library",
...     author: "Matt Haig",
...     isbn: 0525559477,
...     price: 15.49,
...     available: true
...   },
...   {
...     title: "Klara and the Sun",
...     author: "Kazuo Ishiguro",
...     isbn: 059331817X,
...     price: 14.95,
...     available: false
...   },
...   {
...     title: "Mexican Gothic",
...     author: "Silvia Moreno-Garcia",
...     isbn: 0525620788,
...     price: 12.99,
...     available: true
...   }
... ])
...
```

```
...
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('68366faa99789bd225b7123c'),
    '1': ObjectId('68366faa99789bd225b7123d'),
    '2': ObjectId('68366faa99789bd225b7123e'),
    '3': ObjectId('68366faa99789bd225b7123f')
  }
}
biodata>
```

```
db.BooksDB.insertMany([
  {
    title: "Project Hail Mary",
    author: "Andy Weir",
    isbn: "0593135202",
    price: 18.99,
    available: true
  },
  {
    title: "The Midnight Library",
    author: "Matt Haig",
    isbn: "0525559477",
    price: 15.49,
    available: true
  },
  {
    title: "Klara and the Sun",
    author: "Kazuo Ishiguro",
    isbn: "059331817X",
    price: 14.95,
    available: false
  },
  {
    title: "Mexican Gothic",
    author: "Silvia Moreno-Garcia",
    isbn: "0525620788",
    price: 12.99,
    available: true
  }
])
```

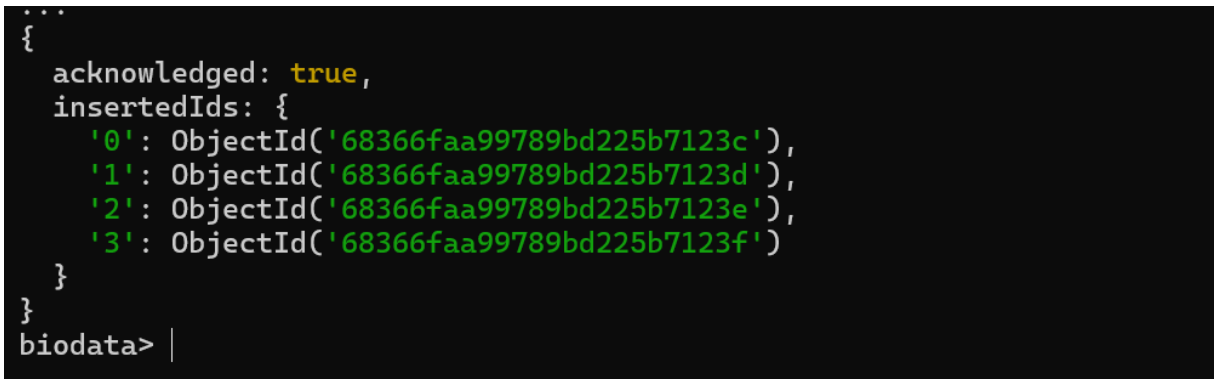
Penjelasan Code ini:

- **db** Objek utama yang merepresentasikan database aktif.
- **BooksDB** Nama koleksi tempat data buku disimpan.
- **insertMany()** Fungsi untuk menambahkan beberapa dokumen dalam satu perintah.
- [...] Array berisi beberapa objek (dokumen buku) yang akan dimasukkan ke koleksi.

Read Operations

Operasi CRUD yang berikutnya adalah perintah Read. Fungsi dari perintah Read adalah untuk membaca atau mengambil document dari sebuah Collection. Kita bisa menggunakan perintah Read dengan method.

db.collection.find()




```
{
 acknowledged: true,
 insertedIds: {
  '0': ObjectId('6836786799789bd225b71245'),
  '1': ObjectId('6836786799789bd225b71246'),
  '2': ObjectId('6836786799789bd225b71247'),
  '3': ObjectId('6836786799789bd225b71248')
 }
}
```

Penjelasan Code Output nya:

- **acknowledged: true**

  Ini berarti server MongoDB mengonfirmasi bahwa perintah insert berhasil dijalankan dan diterima dengan baik.

- **insertedIds**

  Ini adalah objek yang berisi daftar ID unik (ObjectId) yang otomatis dibuat oleh MongoDB untuk masing-masing dokumen yang baru dimasukkan.

  - 0' sampai '3' adalah indeks dari dokumen di array input insertMany.
  - ObjectId('...') adalah nilai unik yang menjadi primary key untuk setiap dokumen di koleksi.

Penjelasan Intinya:

Perintah insertMany() berhasil memasukkan 4 dokumen ke koleksi.

MongoDB mengembalikan ID dari setiap dokumen yang disimpan, supaya kamu bisa referensi dokumen tersebut jika perlu.

Update Operation

Perintah update() berfungsi untuk mengubah document yang ada dalam collection. Ada 3 method yang dapat digunakan dalam update document :

db.collection.updateOne()

Berfungsi untuk mengupdate suatu field dalam single document dengan query filter yang dituliskan. Perintah updateOne() hanya merubah sebuah field tetapi tidak menghapus field yang lain dari document.

```
biodata> db.BooksDB.updateOne(
...     { title: "Klara and the Sun" },
...     { $set: { available: true, price: 13.49 } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
biodata> db.BooksDB.updateMany(
...     { price: { $gt: 15 } },
...     { $set: { available: false } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

```
db.BooksDB.updateOne(
  { title: "Klara and the Sun" },
  { $set: { available: true, price: 13.49 } }
)
```

Penjelasan bagian Code:

1. **db.BooksDB**

Mengakses koleksi bernama BooksDB dalam database aktif saat ini.

2. **.updateOne()**

Metode ini digunakan untuk mengubah satu dokumen pertama yang cocok dengan kondisi filter.

3. **{ title: "Klara and the Sun" }**

Ini adalah filter — MongoDB akan mencari dokumen di mana field title bernilai "Klara and the Sun".

4. **{ $set: { available: true, price: 13.49 } }**

   Ini adalah aksi pembaruan:

   - $set memberitahu MongoDB untuk mengubah atau menambahkan field yang disebutkan.
   - Field available akan diubah menjadi true.
   - Field price akan diubah menjadi 13.49.

Ringkasan:

Kode ini akan mencari buku yang judulnya "Klara and the Sun" dalam koleksi BooksDB, lalu memperbarui nilai available menjadi true dan price menjadi 13.49.

db.collection.updateMany()

Berfungsi untuk mengupdate field dalam multiple document dengan query filter yang dituliskan.

```
biodata> db.BooksDB.updateMany(
...     { price: { $gt: 15 } },
...     { $set: { available: false } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
biodata> db.BooksDB.updateMany(
...     { author: "Matt Haig" },
...     { $set: { author: "Matthew Haig" } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
biodata>
```

```
db.mahasiswa.updateOne(
   { nama: "Budi" },
   { $set: { alamat: "Gellager" } }
);
```

Dan hasilnya:

Fungsi Code Perintah Ini:

Perintah ini digunakan untuk mengubah satu dokumen dalam koleksi mahasiswa di mana field nama bernilai "Budi", dengan mengatur ($set) field alamat menjadi "Gellager".

Penjelasan bagian Code:

1. **db.mahasiswa**

   Mengakses koleksi bernama mahasiswa dalam database aktif.

2. **.updateOne()**

   Fungsi ini melakukan pembaruan pada satu dokumen pertama yang cocok dengan filter.

3. **{ nama: "Budi" }**

   Ini adalah filter query: MongoDB akan mencari dokumen mahasiswa yang memiliki nama bernilai "Budi".

4. **{ $set: { alamat: "Gellager" } }**

   Ini adalah update operator:

   - $set artinya MongoDB akan menambahkan atau memperbarui field alamat.
   - Jika field alamat belum ada, maka akan dibuat.
   - Jika sudah ada, nilainya akan diubah menjadi "Gellager".

Sebelum update:
{ nama: "Budi", prodi: "MI" }

Setelah update:
{ nama: "Budi", prodi: "MI", alamat: "Gellager" }

Contoh Catatan:

- Jika tidak ada dokumen dengan nama: "Budi", maka tidak ada yang berubah.
- Jika ingin menambahkan dokumen baru jika tidak ditemukan (fitur upsert), kamu bisa tambahkan opsi ketiga:

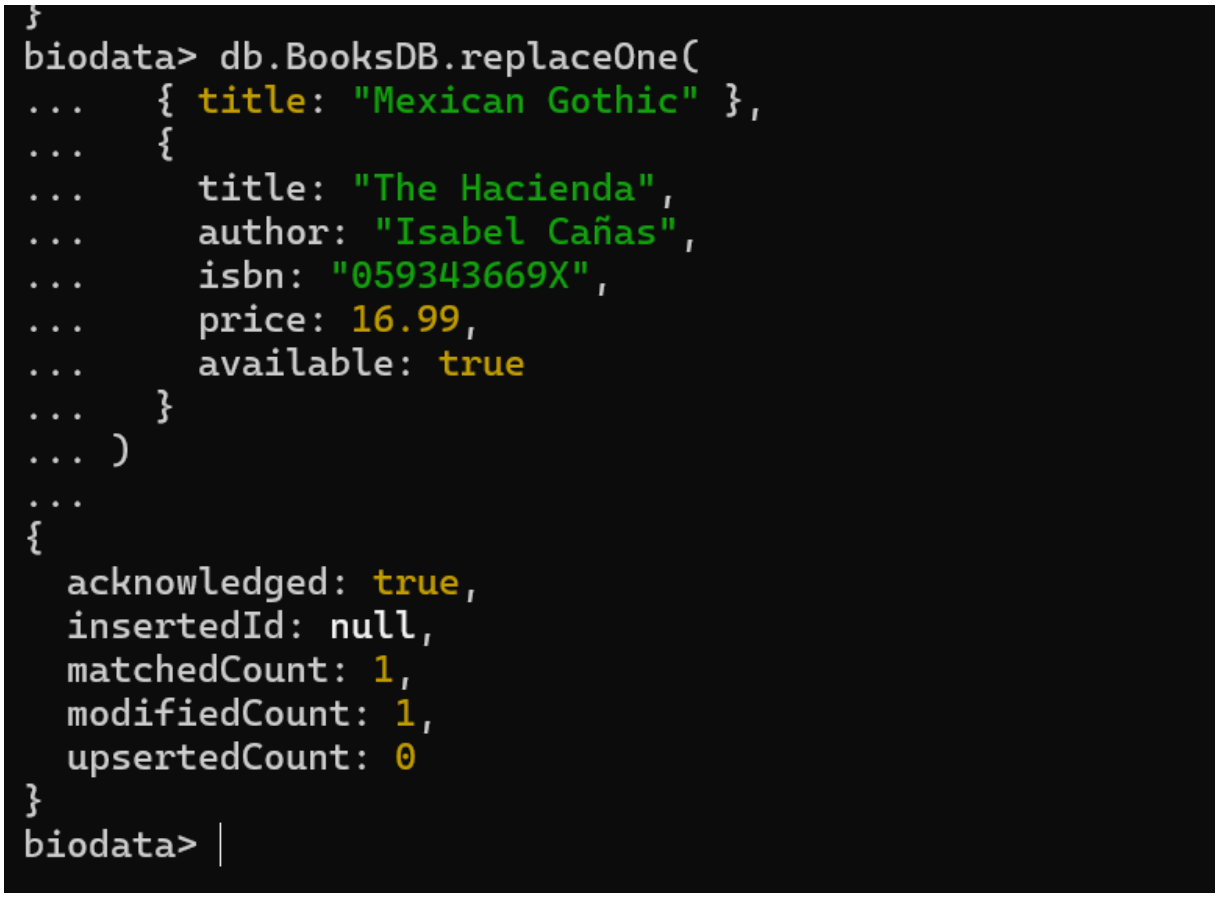
Penjelasan Code Output nya:

```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Field	Penjelasan
acknowledged	true berarti operasi berhasil dan diakui oleh server.
insertedId	null karena ini bukan operasi insert, tapi update.
matchedCount	1 berarti ada 1 dokumen yang cocok dengan filter { nama: "Budi" }.
modifiedCount	1 berarti ada 1 dokumen yang benar-benar diubah.
upsertedCount	0 karena tidak ada dokumen baru yang disisipkan (upsert = false).

db.collection.replaceOne()

Berfungsi untuk menggantikan keseluruhan field pada document dengan field dan document yang baru.




```
db.BooksDB.replaceOne(
 { title: "Mexican Gothic" },
 {
  title: "The Hacienda",
  author: "Isabel Cañas",
  isbn: "059343669X",
  price: 16.99,
  available: true
 }
)
```

Penjelasan Code ini:

1. **db.BooksDB**

   Mengakses koleksi bernama BooksDB.

2. **.replaceOne()**

   Metode ini digunakan untuk:

   - Mencari satu dokumen berdasarkan filter,
   - Lalu mengganti seluruh dokumen tersebut dengan dokumen baru.

3. **{ title: "Mexican Gothic" }**

   Ini adalah filter query — MongoDB akan mencari satu dokumen di mana title bernilai "Mexican Gothic".

Penjelasan Code Output nya:

```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Field	Penjelasan
acknowledged: true	Operasi berhasil dan diakui oleh server MongoDB.
insertedId: null	Tidak ada dokumen baru yang disisipkan karena ini bukan operasi insert atau upsert.
matchedCount: 1	Ada 1 dokumen yang berhasil cocok dengan filter.
modifiedCount: 1	Ada 1 dokumen yang berhasil diubah isinya.
upsertedCount: 0	Tidak ada dokumen baru yang dibuat lewat upsert (upsert: true).

Artinya Secara Umum:

- MongoDB menemukan satu dokumen yang cocok dengan kriteria pencarian (matchedCount: 1),
- Dan dokumen tersebut berhasil diubah (modifiedCount: 1),
- Tidak ada dokumen yang disisipkan atau ditambahkan (upsertedCount: 0, insertedId: null).

Maka output tadi berarti:

- MongoDB berhasil menemukan buku "Mexican Gothic",
- Lalu mengganti seluruh isinya dengan data baru (berhasil di-replace),
- Tidak membuat dokumen baru karena tidak perlu (upsert tidak dipakai).

Delete Operation

Operasi CRUD yang terakhir adalah perintah delete. Perintah delete berfungsi untuk menghapus document dalam collection.ada 2 method yang dapat digunakan untuk perintah delete ini :

db.collection.deleteOne()

Method ini menghapus hanya document pertama pada collection yang sesuai dengan query filter yang diberikan.

```
]
biodata> db.BooksDB.deleteOne({ title: "Klara and the Sun" })
{ acknowledged: true, deletedCount: 1 }
biodata> db.BooksDB.deleteMany({ price: { $lt: 15 } })
{ acknowledged: true, deletedCount: 1 }
biodata> db.BooksDB.deleteMany({ available: true })
{ acknowledged: true, deletedCount: 1 }
biodata> db.BooksDB.find()
[
```

1. **Menghapus Satu Dokumen Berdasarkan Judul**

```
db.BooksDB.deleteOne({ title: "Klara and the Sun" })
```

Penjelasan:

- Mencari dan menghapus satu dokumen pertama yang title-nya adalah "Klara and the Sun".

Penjelasan Output:

```
{ acknowledged: true, deletedCount: 1 }
```

- Server berhasil menerima dan memproses perintah (acknowledged: true).
- Satu dokumen berhasil dihapus (deletedCount: 1).

db.collection.deleteMany()

Method ini menghapus multiple document pada collection yang sesuai dengan query filter yang diberikan.

```
biodata> db.BooksDB.deleteMany({ price: { $lt: 15 } })
{ acknowledged: true, deletedCount: 1 }
biodata>
```

```
biodata> db.BooksDB.deleteMany({ available: true })
{ acknowledged: true, deletedCount: 1 }
biodata>
```

2. **Menghapus Semua Buku dengan Harga di Bawah 15**

```
db.BooksDB.deleteMany({ price: { $lt: 15 } })
```

Penjelasan:

- Mencari dan menghapus semua dokumen yang memiliki price kurang dari 15 ($lt = less than).

Penjelasan Output:

```
{ acknowledged: true, deletedCount: 1 }
```

- Operasi berhasil,
- Hanya ada 1 dokumen dengan harga di bawah 15 yang cocok, dan itu telah dihapus.

3. **Menghapus Semua Buku yang Tersedia (available: true)**

```
db.BooksDB.deleteMany({ available: true })
```

Penjelasan:

- Mencari dan menghapus semua dokumen buku yang field available-nya bernilai true.

Penjelasan Output:

```
{ acknowledged: true, deletedCount: 1 }
```

- Operasi sukses,
- Hanya 1 dokumen dengan available: true yang tersisa saat perintah dijalankan, dan itu berhasil dihapus.

Kesimpulan Umum:

yang telah melakukan tiga operasi penghapusan:

1. Hapus berdasarkan judul 1 terhapus.
2. Hapus berdasarkan harga < 15 1 terhapus.
3. Hapus berdasarkan available: true 1 terhapus.

```
biodata> db.BooksDB.find()
[
  {
    _id: ObjectId('68366faa99789bd225b7123c'),
    title: 'Project Hail Mary',
    author: 'Andy Weir',
    isbn: '0593135202',
    price: 18.99,
    available: false
  },
  {
    _id: ObjectId('68366faa99789bd225b7123d'),
    title: 'The Midnight Library',
    author: 'Matthew Haig',
    isbn: '0525559477',
    price: 15.49,
    available: false
  }
]
biodata>
```

Soal Latihan

Buat sebuah collection dengan nama mahasiswa. Field yang wajib ada dalam document adalah nama, alamat, angkatan, prodi, ipk.

nama	*alamat*	*angkatan*	*prodi*	*ipk*
Adi	Banyuwangi	2021	TI	3.0
Budi	Glagah	2021	MI	3.3
Cahyo	Rogojampi	2020	TI	3.5
Dina	Banyuwangi	2021	TI	3.3
Ely	Muncar	2020	MI	3.4

Tuliskan semua perintah yang anda gunakan serta hasil outputnya mengacu pada ketentuan berikut :

1. Isikan 2 data pertama pada collection mahasiswa dengan menggunakan perintah insertOne()!
2. Isikan 3 data berikutnya pada collection mahasiswa dengan perintah insertMany()!
3. Tampilkan semua document pada collection mahasiswa yang beralamatkan “Banyuwangi”!
4. Ubah alamat mahasiswa atas nama “Budi” menjadi “Gladak”!
5. Ubah semua document yang memiliki prodi “TI” menjadi “Teknik Informatika”!
6. Hapus semua document yang memiliki prodi “MI”

Tuliskan semua perintah yang anda gunakan serta hasil outputnya mengacu pada ketentuan berikut :

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6836786799789bd225b71245'),
    '1': ObjectId('6836786799789bd225b71246'),
    '2': ObjectId('6836786799789bd225b71247'),
    '3': ObjectId('6836786799789bd225b71248')
  }
}
```

```
mahasiwa> db.mahasiswa.insertMany([
...     {
...         nama: "Budi",
...         alamat: "Glagah",
...         angkatan: 2021,
...         prodi: "MI",
...         ipk: 3.3
...     },
...     {
...         nama: "Cahyo",
...         alamat: "Rogojampi",
...         angkatan: 2020,
...         prodi: "TI",
...         ipk: 3.5
...     },
...     {
...         nama: "Dina",
...         alamat: "Banyuwangi",
...         angkatan: 2021,
...         prodi: "TI",
...         ipk: 3.3
...     },
...     {
...         nama: "Ely",
...         alamat: "Muncar",
...         angkatan: 2020,
...         prodi: "MI",
...         ipk: 3.4
...     }
```

1. **Isikan 2 data pertama pada collection mahasiswa dengan menggunakan perintah insertOne()!**

```
}
mahasiwa> db.mahasiswa.insertOne({
...     nama: "Budi",
...     alamat: "Glagah",
...     angkatan: 2021,
...     prodi: "MI",
...     ipk: 3.3
... });
...
... db.mahasiswa.insertOne({
...     nama: "Cahyo",
...     alamat: "Rogojampi",
...     angkatan: 2020,
...     prodi: "TI",
...     ipk: 3.5
... });
...
{
  acknowledged: true,
  insertedId: ObjectId('6836792199789bd225b7124a')
}
```

Penjelasan:

- Menyisipkan (menambahkan) satu dokumen mahasiswa baru ke dalam koleksi mahasiswa.
- Data mahasiswa yang dimasukkan:
    - nama: "Budi"
    - alamat: "Glagah"
    - angkatan: 2021
    - prodi: "MI" (Manajemen Informatika)
    - ipk: 3.3 (Indeks Prestasi Kumulatif)

2.Penjelasan:

- Menambahkan mahasiswa baru lagi ke dalam koleksi mahasiswa.
- Data yang dimasukkan:
    - nama: "Cahyo"
    - alamat: "Rogojampi"
    - angkatan: 2020
    - prodi: "TI" (Teknik Informatika)
    - ipk: 3.5

2. **Isikan 3 data berikutnya pada collection mahasiswa dengan perintah insertMany()!**

```
mahasiwa> db.mahasiswa.insertMany([
...     {
...         nama: "Budi",
...         alamat: "Glagah",
...         angkatan: 2021,
...         prodi: "MI",
...         ipk: 3.3
...     },
...     {
...         nama: "Cahyo",
...         alamat: "Rogojampi",
...         angkatan: 2020,
...         prodi: "TI",
...         ipk: 3.5
...     },
...     {
...         nama: "Dina",
...         alamat: "Banyuwangi",
...         angkatan: 2021,
...         prodi: "TI",
...         ipk: 3.3
...     },
...     {
...         nama: "Ely",
...         alamat: "Muncar",
...         angkatan: 2020,
...         prodi: "MI",
...         ipk: 3.4
...     }
... ]);
...
{
...
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6836786799789bd225b71245'),
    '1': ObjectId('6836786799789bd225b71246'),
    '2': ObjectId('6836786799789bd225b71247'),
    '3': ObjectId('6836786799789bd225b71248')
  }
}
mahasiwa>
```

Penjelasan:

Fungsi:

Perintah ini digunakan untuk menambahkan banyak dokumen sekaligus ke dalam koleksi BooksDB di database aktif.

Data yang Dimasukkan:

Data yang di tambahkan 4 dokumen buku, masing-masing dengan properti berikut:

Project Hail Mary

- Author: Andy Weir
- ISBN: 0593135202
- Harga: $18.99
- **Tersedia: true**

The Midnight Library

- Author: Matt Haig
- ISBN: 0525559477
- Harga: $15.49
- **Tersedia: true**

Klara and the Sun

- Author: Kazuo Ishiguro
- ISBN: 059331817X
- Harga: $14.95
- Tersedia: false

Mexican Gothic

- Author: Silvia Moreno-Garcia
- ISBN: 0525620788
- Harga: $12.99
- Tersedia: true

Keterangan :

- insertMany() adalah metode untuk menyisipkan banyak dokumen dalam satu perintah.
- MongoDB akan:
    - Membuat _id secara otomatis jika tidak disediakan,
    - Menyisipkan semua dokumen ke koleksi BooksDB,
    - Memberikan output berupa daftar ID yang berhasil dimasukkan.

```
mahasiwa> db.mahasiswa.insertMany([
...     {
...         nama: "Budi",
...         alamat: "Glagah",
...         angkatan: 2021,
...         prodi: "MI",
...         ipk: 3.3
...     },
...     {
...         nama: "Cahyo",
...         alamat: "Rogojampi",
...         angkatan: 2020,
...         prodi: "TI",
...         ipk: 3.5
...     },
...     {
...         nama: "Dina",
...         alamat: "Banyuwangi",
...         angkatan: 2021,
...         prodi: "TI",
...         ipk: 3.3
...     },
...     {
...         nama: "Ely",
...         alamat: "Muncar",
...         angkatan: 2020,
...         prodi: "MI",
...         ipk: 3.4
...     }
... ]);
...
{
```

3. **Tampilkan semua document pada collection mahasiswa yang beralamatkan “Banyuwangi”!**

```
mahasiwa> db.mahasiswa.find({ alamat: "Banyuwangi" });
[
  {
    _id: ObjectId('6836786799789bd225b71247'),
    nama: 'Dina',
    alamat: 'Banyuwangi',
    angkatan: 2021,
    prodi: 'TI',
    ipk: 3.3
  }
]
mahasiwa>
```

```
mahasiwa> db.mahasiswa.find({ alamat: "Banyuwangi" }).pretty();
[
  {
    _id: ObjectId('6836786799789bd225b71247'),
    nama: 'Dina',
    alamat: 'Banyuwangi',
    angkatan: 2021,
    prodi: 'TI',
    ipk: 3.3
  }
]
mahasiwa>
```

Fungsi Perintah:

- db.mahasiswa.find(...) digunakan untuk mencari dokumen (data) di koleksi mahasiswa yang memiliki field alamat dengan nilai "Banyuwangi".
- pretty() adalah tambahan yang digunakan untuk menampilkan hasil dengan format yang lebih rapi dan mudah dibaca.

Hasil Output:

MongoDB menampilkan hasil berupa satu dokumen:

```
{
  _id: ObjectId('6836786799789bd225b71247'),
  nama: 'Dina',
  alamat: 'Banyuwangi',
  angkatan: 2021,
  prodi: 'TI',
  ipk: 3.3
}
```

Penjelasan Data:

- **id: ObjectId('6836786799789bd225b71247')**

  ID unik yang dibuat otomatis oleh MongoDB saat dokumen dimasukkan.

- **nama: "Dina"**

  Nama mahasiswa.

- **alamat: "Banyuwangi"**

Alamat tempat tinggal mahasiswa.

- **angkatan: 2021**

  Tahun masuk kuliah.

- **prodi: "TI"**

  Program Studi: Teknik Informatika.

- **ipk: 3.3**

  Indeks Prestasi Kumulatif mahasiswa tersebut.

Kesimpulan:

- Perintah berhasil menemukan dan menampilkan mahasiswa bernama Dina yang tinggal di Banyuwangi.
- find() akan menampilkan data dalam bentuk array dokumen.
- pretty() hanya untuk tampilan agar data lebih mudah dibaca (struktur baris dan indentasi).

4. **Ubah alamat mahasiswa atas nama “Budi” menjadi nama lain**

```
mahasiwa> db.mahasiswa.updateOne(
...     { nama: "Budi" },
...     { $set: { alamat: "Gellager" } }
... );
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
mahasiwa>
```

```
mahasiwa> db.mahasiswa.find()
[
  {
    _id: ObjectId('6836786799789bd225b71245'),
    nama: 'Budi',
    alamat: 'Gellager',
    angkatan: 2021,
    prodi: 'MI',
    ipk: 3.3
  },
```

Penjelasan:

- db.mahasiswa.updateOne(...) digunakan untuk mengubah (update) satu dokumen pertama dalam koleksi mahasiswa yang cocok dengan kondisi pencarian.
- { nama: "Budi" } adalah kriteria pencarian: MongoDB akan mencari mahasiswa yang nama-nya "Budi".
- $set: { alamat: "Gellager" } adalah aksi perubahan: MongoDB akan mengubah nilai field alamat menjadi "Gellager" tanpa mengubah field lainnya.

```
mahasiwa> db.mahasiswa.find()
[
  {
    _id: ObjectId('6836786799789bd225b71245'),
    nama: 'Budi',
    alamat: 'Gellager',
    angkatan: 2021,
    prodi: 'MI',
    ipk: 3.3
  },
```

5. Ubah semua document yang memiliki prodi “TI” menjadi “Teknik Informatika”!

```
mahasiwa> db.mahasiswa.updateMany(
...     { prodi: "MI" },
...     { $set: { prodi: "Multimedia" } }
... );
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 3,
  modifiedCount: 3,
  upsertedCount: 0
}
```

```
mahasiwa> db.mahasiswa.updateMany(
...     { prodi: "TI" },
...     { $set: { prodi: "Teknik Informatika" } }
... );
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 3,
  modifiedCount: 3,
  upsertedCount: 0
}
mahasiwa>
```

1.Penjelasan:

- Fungsi: Mengubah semua dokumen di koleksi mahasiswa yang memiliki prodi: "TI" menjadi prodi: "Teknik Informatika".
- $set digunakan untuk mengganti atau menetapkan nilai baru ke field prodi.

- updateMany berarti semua dokumen yang cocok dengan kondisi { prodi: "TI" } akan diperbarui.

2.Penjelasan

- Fungsi: Mengubah semua data mahasiswa yang sebelumnya prodi: "MI" menjadi prodi: "Multimedia".
- Proses ini tidak menghapus atau memengaruhi field lainnya, seperti nama, alamat, angkatan, atau ipk.
- Hanya field prodi yang diperbarui.

Kesimpulan:

Kamu telah melakukan pembaruan massal (bulk update) terhadap data mahasiswa:

- Semua TI menjadi "Teknik Informatika".
- Semua MI menjadi "Multimedia".

```
  {
    _id: ObjectId('6836786799789bd225b71246'),
    nama: 'Cahyo',
    alamat: 'Rogojampi',
    angkatan: 2020,
    prodi: 'Teknik Informatika',
    ipk: 3.5
  },
  {
    _id: ObjectId('6836786799789bd225b71247'),
    nama: 'Dina',
    alamat: 'Banyuwangi',
    angkatan: 2021,
    prodi: 'Teknik Informatika',
    ipk: 3.3
  },
  {
    _id: ObjectId('6836792199789bd225b7124a'),
    nama: 'Cahyo',
    alamat: 'Rogojampi',
    angkatan: 2020,
    prodi: 'Teknik Informatika',
    ipk: 3.5
  }
]
```

6. **Hapus semua document yang memiliki prodi MI/Multimedia**

```
mahasiwa> db.mahasiswa.deleteMany({ prodi: "Multimedia" })
{ acknowledged: true, deletedCount: 3 }
mahasiwa>
```

Penjelasan:

- deleteMany() digunakan untuk menghapus banyak dokumen sekaligus dalam satu operasi.
- { prodi: "Multimedia" } adalah filter: MongoDB akan mencari semua dokumen di koleksi mahasiswa yang memiliki field prodi dengan nilai "Multimedia" dan menghapus semuanya.

```
mahasiwa> db.mahasiswa.find()
[
  {
    _id: ObjectId('6836786799789bd225b71245'),
    nama: 'Budi',
    alamat: 'Gellager',
    angkatan: 2021,
    prodi: 'Multimedia',
    ipk: 3.3
  },
  {
    _id: ObjectId('6836792199789bd225b71249'),
    nama: 'Budi',
    alamat: 'Glagah',
    angkatan: 2021,
    prodi: 'Multimedia',
    ipk: 3.3
  },
```

A.Hasil Uji Coba/Hasil Nya

- Use = Untuk masuk kedalam databases
- db.Bukul.insertOne ({Isi data pada kolom ini}) = Funsinya adalah untuk membuat dukumen pada collection (Buku) dan hanya satu data karena syntexnya One.
- db.Anggota.insertOne ({Isi data pada kolom ini}) = Funsinya adalah untuk membuat dukumen pada collection (Anggota) dan hanya satu data karena syntexnya One.
- db.Pinjam.insertMany({Isi data pada kolom ini}) = Funsinya adalah untuk membuat dukumen pada collection (Pinjam).

Kesimpulan

Kesimpulan Percobaan 1

Dengan study kasus ini Saya bisa membuat Relasi One-to-One adalah relasi dimana suatu baris tabel A hanya berhubungan dengan suatu baris tabel B. Kita dapat menerapkan model data Embedded Document untuk mempresentasikan relasi One-to-One tersebut pada database Non Relational.

Kesimpulan Percobaan 2

Dengan study kasus ini Saya bisa membuat Mirip dengan implementasi embedded document pada relasi One-to-One, pada kasus One-to-Many, document dengan relasi “Many” akan di-embedded kedalam document dengan relasi “One”.

Kesimpulan Percobaan 3

Dengan study kasus ini Saya bisa membuatUntuk menampilkan data kita dapat menggunakan method $lookup dengan aggregation.

Kesimpulan Percobaan 4

Dengan study kasus ini Saya bisa membuat Collection dan juga menambahkan dukumen menggunakan one to one dan many to one, dan juga query untuk menampilkan isi document pada collection

BAB 1

PENUTUP

Kesimpulan

teori MySQL adalah Structured Query Language (SQL) sebagai bahasa interaktif untuk mengelola data. MySQL adalah sistem manajemen basis data relasional (RDBMS) yang menggunakan SQL untuk menjalankan fungsinya..

Saran

Saran saya tidak ada perubahan pembelajaran sama seperti si Semester 1 dan masih masuk di praktikum kali kecuali pada pengisian laporan praktikum database ke word itu yang perlu di tanyakan dan cara pengisian laporan untuk kedepannya.

DAFTAR PUSTAKA

Tuliskan rujukan yang anda gunakan baik website maupun buku seperti contoh dibawah.

1. Tim Asisten Dosen. 2014. Modul 1 Pengenalan Sistem Operasi, Ide Visual C++, Dan Algoritma Pemrograman. Malang: Unversitas Negeri Malang.
2. Program Konversi Suhu (online) http://bondanoky.blogspot.com/2012/10/program-konversi-suhu-c.html. Di akses 8 September.

LAMPIRAN

Berisikan syntax atau gambar yang dibutuhkan dalam tiap pertemuan praktikum.