**LAPORAN PRAKTIKUM**

**DATABASE NO RELATIONAL**

Disusun untuk Memenuhi Matakuliah Database No Relational

Dibimbing oleh

Sulaibatul Aslamiyah S.Kom M.Kom



Oleh:

**Rachmad Lukman Maulana**

**1124102171**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**SEKOLAH TINGGI ILMU KOMPUTER PGRI BANYUWANGI**

**2025**

**LEMBAR PENGESAHAN**

**LAPORAN PRAKTIKUM**

Matakuliah : Database No\_Relational

Oleh : Rachmad Lukman Maulana

NIM : 1124102171

# Telah disahkan pada

Hari : Kamis

Tanggal : 2025

**Mengetahui/ Menyetujui :**

Dosen Pengampu Mata Kuliah

**Sulaibatul Aslamiyah S.Kom M.Kom**

NIDN. 0712018304

**KATA PENGANTAR**

Puji syukur kita panjatkan kepada Tuhan Yang Maha Esa, karena berkat rahmat-Nya, openyusunan materi mata kuliah *Database No\_SQL* ini dapat diselesaikan dengan baik. Mata kuliah *Database No\_SQL* merupakan bagian penting dalam pengembangan kemampuan mahasiswa dalam bidang teknologi informasi dan komputer. Dalam mata kuliah ini, mahasiswa akan mempelajari dasar-dasar tentang pengelolaan dan manajemen data menggunakan sistem manajemen basis data No\_SQL yang banyak digunakan dalam pengembangan aplikasi berbasis web maupun desktop.

Database adalah komponen penting dalam sistem informasi modern, dan No\_SQL sebagai salah satu sistem manajemen basis data relasional yang paling populer, memberikan landasan yang kuat bagi mahasiswa untuk mengelola dan mengorganisir data secara efisien. Materi yang akan dipelajari mencakup struktur dasar database, perancangan tabel, pengolahan data, hingga query yang digunakan untuk memanipulasi dan mengambil data yang relevan sesuai kebutuhan.

Dengan mempelajari mata kuliah ini, diharapkan mahasiswa dapat memahami konsepkonsep dasar yang akan berguna dalam dunia profesional, serta mampu mengembangkan dan menerapkan solusi berbasis database yang efektif dan efisien. Selain itu, penguasaan terhadap No\_SQL juga akan memperluas keterampilan mahasiswa dalam menghadapi tantangan dunia teknologi yang semakin berkembang pesat.

Semoga materi yang disampaikan dalam mata kuliah ini dapat memberikan manfaat yang sebesar-besarnya bagi mahasiswa dan dapat diterapkan dengan baik dalam dunia kerja maupun dalam pengembangan karir di masa depan. Terima kasih kepada semua pihak yang telah mendukung terselenggaranya mata kuliah ini, baik dari pengajar, staf pengelola, maupun mahasiswa yang telah berpartisipasi dengan antusias.

## 1.1 Capaian Pertemuan 10

Pada praktikum MongoDB ini, kami akan mengimplementasikan JSON Schema adalah standar resmi untuk memvalidasi data JSON. Dengan menggunakan JSON Schema, kita bisa memberi batasan, data JSON apa yang valid, sehingga bisa dimasukkan ke dalam collectionoperasi

**Pembuatan Database dan Collection:**

1. Berhasil membuat database STIKOM dan collection data\_mhs untuk menyimpan data mahasiswa yang sudah tertera di modul pertemuan 13.

## 1.2 Indikator Capaian

1. Memahami konsep dasar NoSQL, khususnya MongoDB.
2. Melatih penggunaan perintah operasi Schema Validasi yang bertujuan untuk memberi Batasan pada suatu dokumen. MongoDB untuk melakukan banyak operasi sekaligus.
3. Mempelajari cara membuat database, collection, dan melakukan Schema Validasi.

## 1.3 Landasan Teori

Database NoSQL adalah jenis sistem manajemen basis data yang tidak menggunakan model relasional seperti pada database SQL tradisional. NoSQL dirancang untuk menangani data yang tidak terstruktur atau semi-terstruktur dan memberikan fleksibilitas lebih dalam penyimpanan serta pengolahan data yang besar dan kompleks. Salah satu ciri utama database NoSQL adalah kemampuannya untuk menyimpan data dalam bentuk yang lebih fleksibel.

## 1.4 Pelaksanaan Praktikum

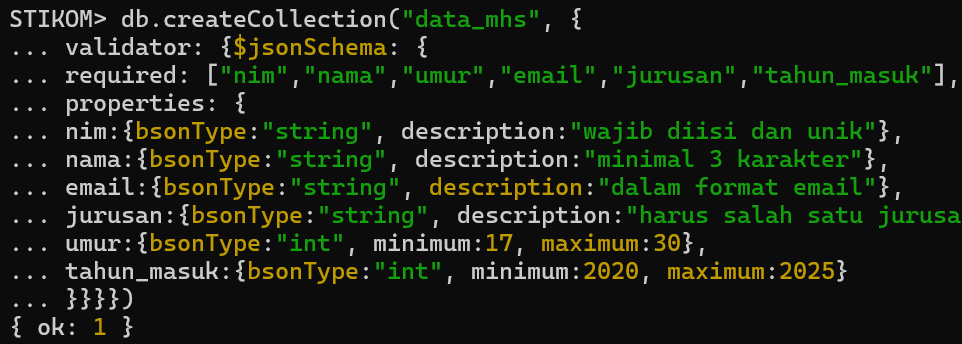
### 1.4.1 Implementasi Hail Uji Coba

Pada percobaan pertama kita akan membuat database dengan nama STIKOM dan membuat sebuah collection dengan nama data\_mhs. Yang di dalamnya terdapat nim, nama, umur, email, jurusan, dan tahun masuk.

1. **Membuat Database Bulk Write**



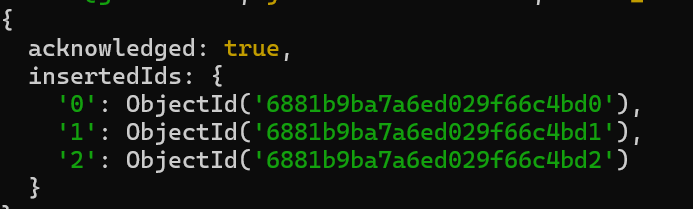
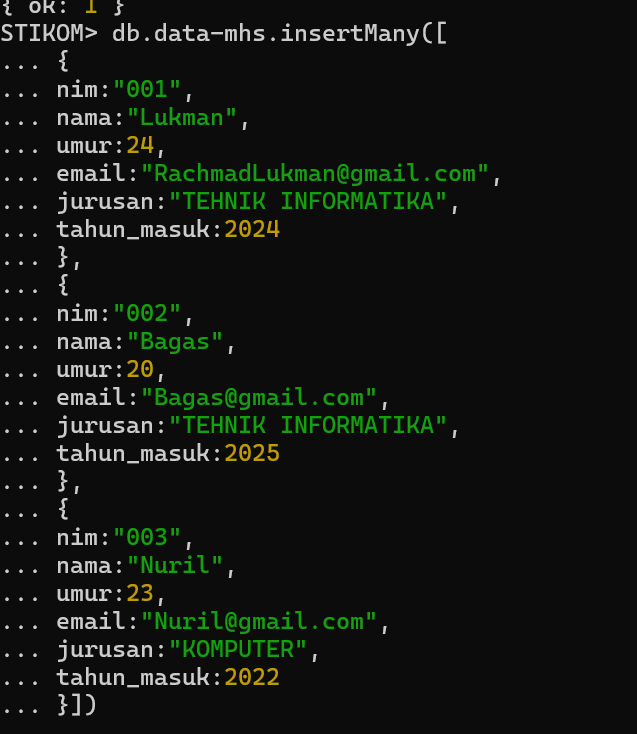
# Membuat Collection dan Melakukan Validasi



Deskripsi : 1. Kita membuat collection dengan nama **data\_mhs**. Validator ini digunakan untuk kita memasukkan **json schemanya** sebagai tempat validasinya. Kemudian, **required** ini digunakan untuk menentukan field dan **properties** ini digunakan untuk spesifikasi dari field tersebut. **BsonType** ini untuk menentukan tipe datanya apakah dibuat string atau int, dan description ini digunakan untuk memberi tahu kesalahan kita saat melakukan insert. Ketika berhasil maka aka nada notifikasi **{ok:1}.**

1. **Melakukan Insert Data Valid**

Kita diperintah untuk memasukkan 3 data yang valid.

****

**Deskripsi :** kemudian kita melakukan insert data seperti biasa menggunakan insert many dan memasukkan field yang sesuai dengan schema validasinya. Untuk NIM mengapa kita pakai tipe string karena kita sudah menentukan tipe datanya di schema validasi tadi. Jika hasilnya true maka data berhasil di masukkan.

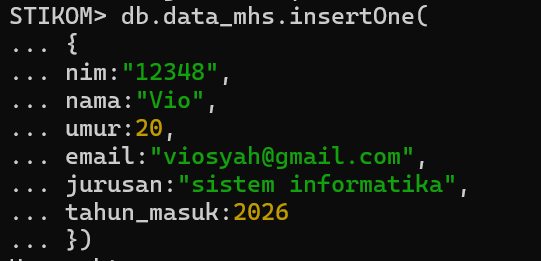
1. **Menampilkan Hasil Insert Data**

****

**Deskripsi:** hasilnya seperti di atas kita menampilkan collection data\_mhs menggunakan **db.data\_mhs.find().** Kita mempunyai 3 data yang valid.

1. **Melakukan Insert Data Invalid**

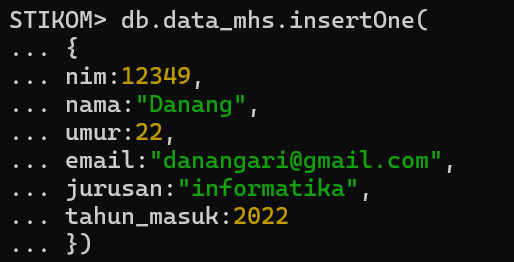
* **Data 1**

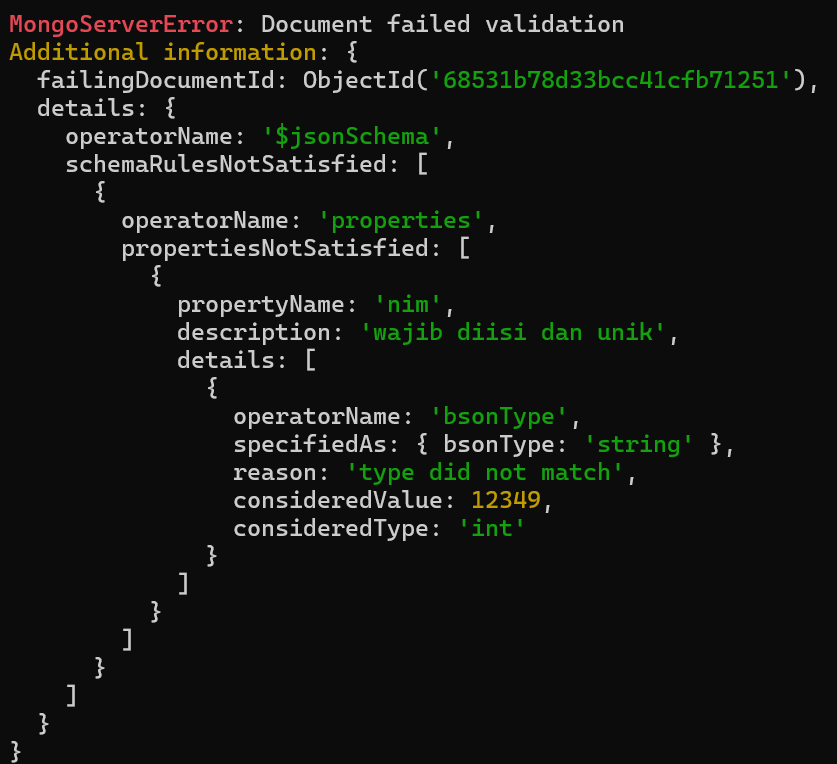
****

****

**Deskripsi :** kita mencoba untuk memasukkan data invalid secara sengaja, maka akan muncul notifikasi seperti diatas. Dimana kesalahannya terletak pada tahun masuk. Di dalam schema validasi tadi kita sudah menentukan maximal tahun masuknya yaitu 2025 tetapi kita memasukkan tahun masuknya 2026 yang berakibat invalid. Maka data dengan nama Vio tidak akan dimasukkan ke dalam collection data\_mhs.

# Data 2





**Deskripsi :** kita memasukkan data dengan nama Danang kemudian eror karena pada nim kita menggunakan tipe data integer, padahal kita sudah menentukan tipe datanya yaitu string yang ditandai dengan tanda petik dua (“ “). Data dengan nama Danang tidak dimasukkan ke dalam collection data\_mhs.

1. **Melakukan Update Collection dengan Validator**

**Pengembangan Validasi Lanjutan (Challenge)**

* Tambahkan validasi field status yang hanya boleh berisi aktif, cuti, lulus.
* Validasi nomor telepon dengan minimal 11 digit angka.
* Tambahkan aturan jika tahun\_masuk lebih dari 2023, maka status harus aktif.

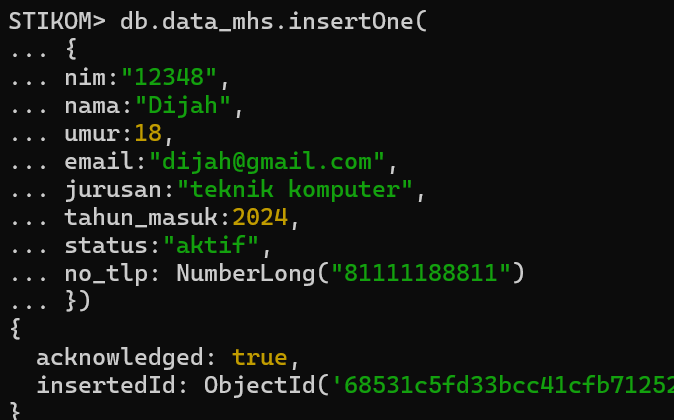


****

**Deskripsi :** kita diperintah untuk update collection dengan menambahkan field status yang hanya berisi [aktif, cuti, lulus] dan no\_tlp minimal 11 digit. Yang pertama kita akan menggunakan sintaks **db.runCommand** untuk update collectionnya. Kemudian kita menggunakan **collMod** untuk memanggil nama collection yang akan kita update. Setelah saya acari-cari di internet maka metode yang saya pilih yaitu metode yang menggunakan **operator logika $and** untuk menggabungkan validator dari **$jsonschema dan $expr** karena kedua validator ini tidak bisa digabung dan berdiri sendiri maka harus menggunakan operator logika $and. Validator **$jsonshcema** ini digunakan **untuk menentukan struktur datanya** sedangkan **validator $expr** ini **menentukan operasi logikanya.** Kenudian kita memasukkan sintaks seperti awal membuat validator dan kita juga menambahkan **field baru yaitu status dan no\_tlp.** Kemudian kita membuat sebuah operasi logika yaitu **jika tahun masuknya lebih dari 2023 maka status akan aktif.** Kemudian kita membuat validator $expr dan dialamnya kita kasih operator logika $and untuk menggabungkan dan **membandingkan no\_tlp dan status** dengan operator **$or,$gte,$lte, dan $eq.**

1. **Insert Data dengan Update Validator**

* **Data Valid**

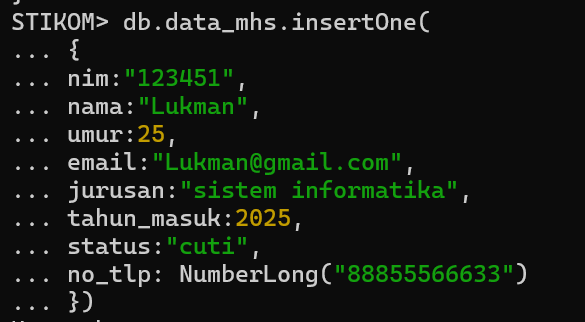
****

****

**Deskripsi :** kita akan memasukkan data yang validatornya sudah kita update dengan nama Dijah. Terlihat setelah di update field baru akan muncul yaitu status dan no\_tlp. Kemudian Ketika tahun masuknya 2024 dan statusnya aktif, serta no\_tlp nya lebih dari 11 digit maka hasilnya akan true karena sesuai dengan update validatornya dan datanya akan ditambahkan ke dalam collection data\_mhs.

**Note:** mengapa no\_tlp menggunakan **NumberLong?** Karena tipe datanya kita buat **“Long”.**

* **Data Invalid**

****

****

**Deskripsi :** kemudian kita masukkan data yang **invalid Dimana jika tahun masuknya 2025, maka status cuti**. Nah ini salah! Karena**, operasi logikanya jika tahun lebih 2023 maka status harus aktif yang artinya tahun di atas 2023 statusnya harus aktif tidak boleh cuti atau lulus.** Maka data tidak akan dimasukkan ke dalam collection data\_mhs.

## 1.5 Kesimpulan

Melalui praktikum ini, dapat disimpulkan bahwa MongoDB menggunakan operasi Schema Validasi yang tepat kita dapat menentukan Batasan terhadap data yang ada di dalam dokumen. Schema Validasi ini memiliki manfaat juga yaitu menjamin konsistensi data, menghindari data eror atau corrupt, membantu dokumentasi struktur data dan menyederhanakan validasi aplikasi. Kemudia pada materi update validasinya digunakan untuk update jika kita ingin menambah field baru dan menentukan Batasan pada datanya. Tetapi kita harus teliti terhadap tanda kurung yang banyak. Dengan memahami operasi Schema Validasi secara mendalam, pengguna MongoDB dapat membangun aplikasi yang skalabel dan cepat, dengan pengelolaan data yang efisien serta struktur data yang lebih natural dibanding basis data relasional.