



# Mongo DB

Made by : Abdul Hafeez  
[www.codehafeez.com](http://www.codehafeez.com)

- ▶ MongoDB is NoSQL language.
- ▶ MongoDB is a cross-platform, document oriented database that provides, high performance, high availability, and easy scalability. MongoDB works on concept of collection and document.

- **MongoDB** is a free and open-source cross-platform document-oriented database program.
- Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemas.

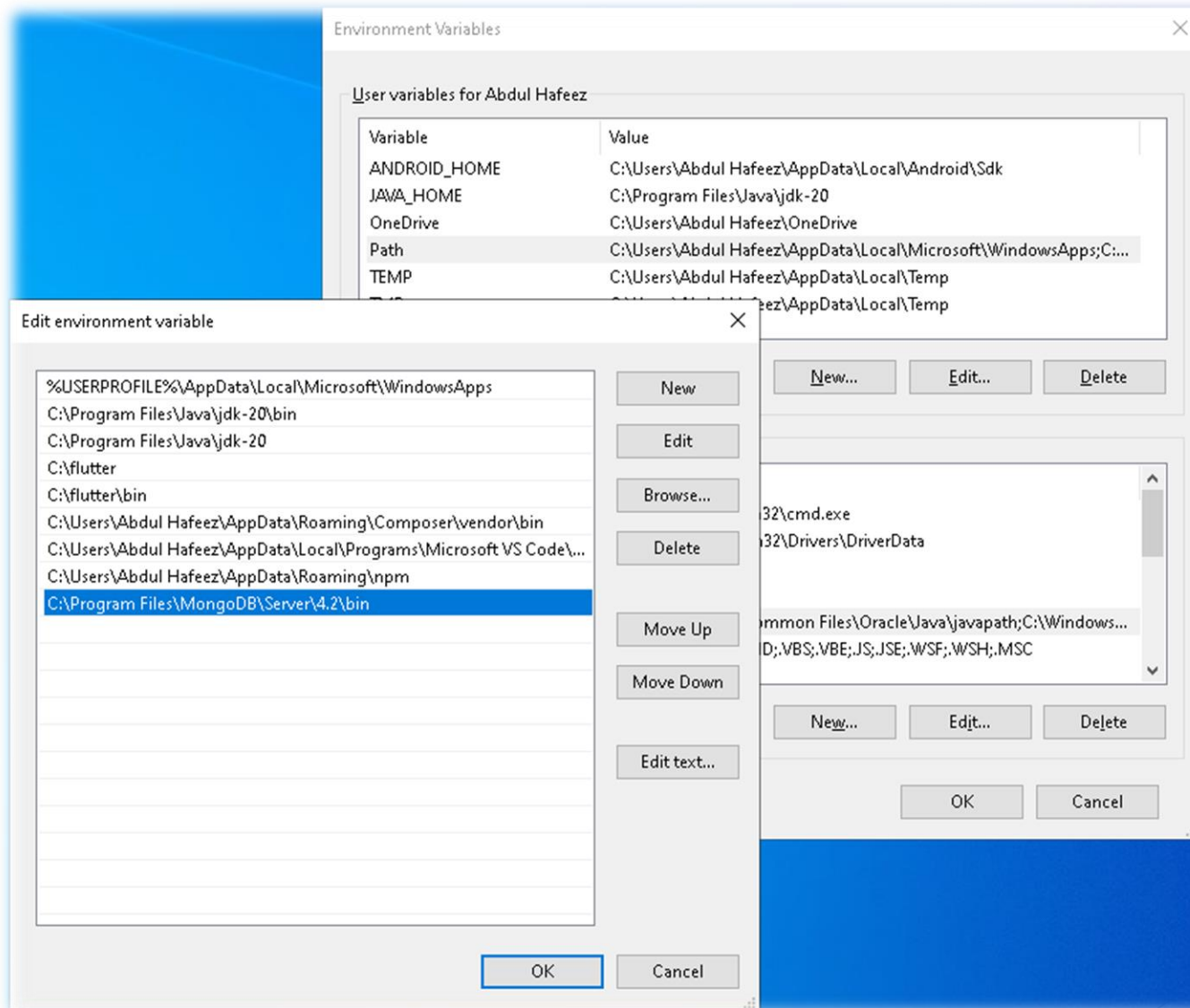
- ▶ Download IDE MongoDB

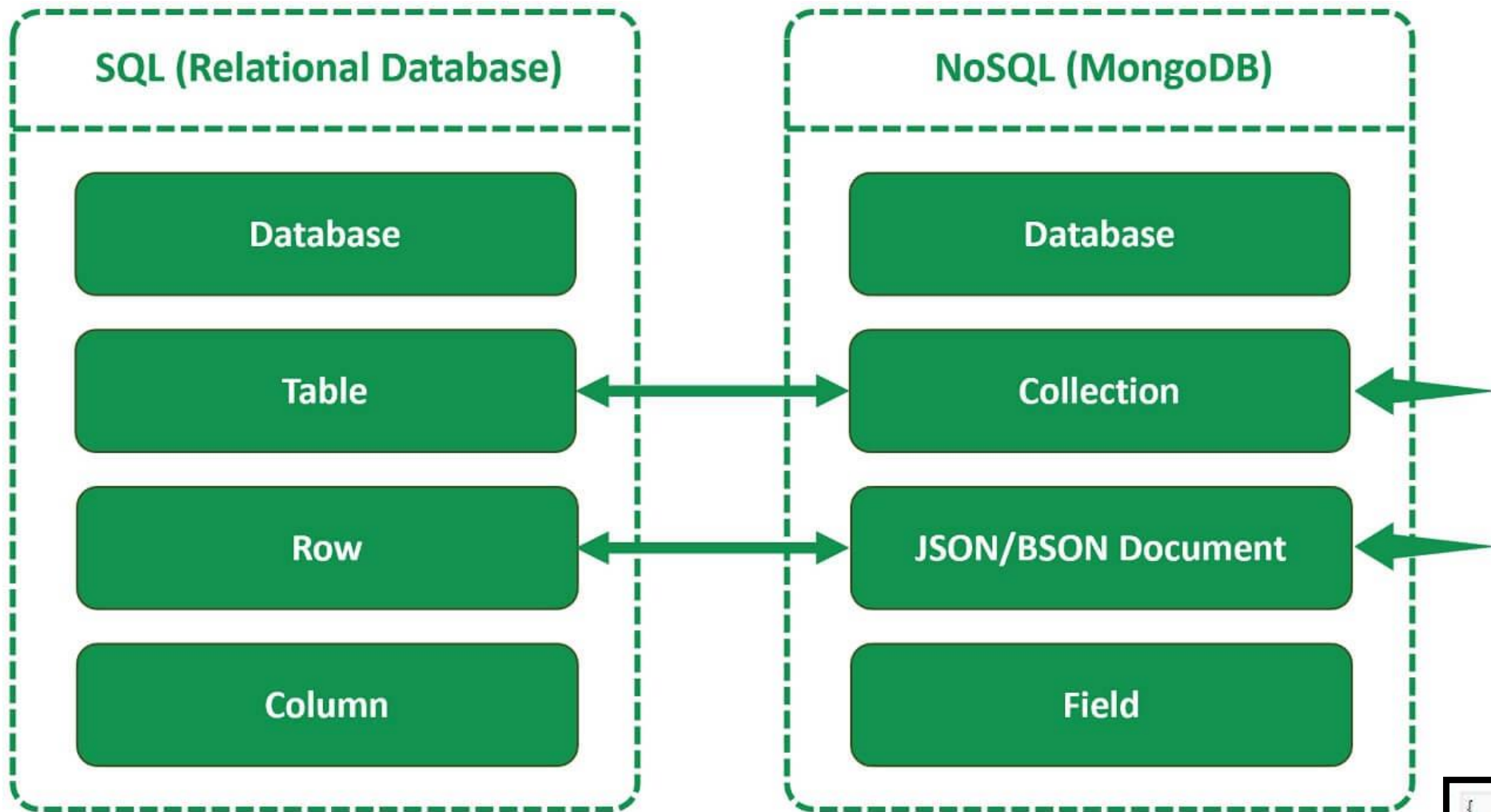
<https://www.mongodb.com/try/download/community>

- ▶ `mongo --version`

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by MongoDB itself)

# Install & Set Environment Variables





```
{
  person: {
    first_name: "Peter",
    last_name: "Peterson",
    addresses: [
      {street: "123 Peter St"},
      {street: "504 Not Peter St"}
    ],
  },
}
```

# Insert Single Data

The image shows a screenshot of the MongoDB Compass application interface. The main window is titled "MongoDB Compass - localhost:27017/hafeez\_db". The left sidebar shows the database structure with "admin", "config", and "hafeez\_db" databases. The "hafeez\_db" database is expanded, showing "employees" and "students" collections. The "students" collection is selected, and its storage size is 4.10 kB. The "Collections" tab is active, showing a list of collections. The "employees" collection is also visible with a storage size of 4.10 kB. The "Documents" tab is selected for the "students" collection, showing a list of documents. An "Insert Document" dialog is open, showing the JSON document to be inserted into the "students" collection. The document contains the following fields: "employee\_id", "employee\_name", "employee\_email", "employee\_phone", "employee\_dob", "employee\_age", and "employee\_address". The "Insert" button is highlighted in green.

**Insert Document**

To collection hafeez\_db.students

```
1 {
2   "employee_id": 4,
3   "employee_name": "Abdul Hafeez",
4   "employee_email": "hafeez@example.com",
5   "employee_phone": "555-12-4567",
6   "employee_dob": "1988-11-01",
7   "employee_age": 33,
8   "employee_address": "7812 Oak Road, CityC, CountryZ"
9 }
```

Cancel Insert

# Insert Multiple Data

MongoDB Compass - localhost:27017/hafeez\_db.employees

Connect Edit View Collection Help

localhost:27017 Documents hafeez\_db.emplo...

My Queries

Databases

Search

admin

config

hafeez\_db

employees

students

local

hafeez\_db

Documents

Filter

ADD DATA

Insert Document

To collection hafeez\_db.employees

VIEW

```
1 [
2   {
3     "employee_id": 1,
4     "employee_name": "John Doe",
5     "employee_email": "john.doe@example.com",
6     "employee_phone": "123-456-7890",
7     "employee_dob": "1985-08-15",
8     "employee_age": 37,
9     "employee_address": "123 Main Street, CityA, Count
10  },
11  {
12    "employee_id": 2,
13    "employee_name": "Jane Smith",
14    "employee_email": "jane.smith@example.com",
15    "employee_phone": "987-654-3210",
16    "employee_dob": "1990-03-20",
17    "employee_age": 32,
18    "employee_address": "456 Park Avenue, CityB, Count
19  },
20 ]
```

Cancel Insert

0 DOCUMENTS 1 INDEXES

Explain Reset Find Options

0 - 0 of 0

# View Data

MongoDB Compass - localhost:27017/hafeez\_db.students

Connect Edit View Collection Help

localhost:27017 Documents hafeez\_db.stude...

My Queries Databases Search

- admin
- config
- hafeez\_db
  - employees
  - students**
- local

### hafeez\_db.students

3 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options

ADD DATA EXPORT DATA 1 - 3 of 3

```
{ "_id": ObjectId('64be9cb9066fe416c2a44c84'), "student_id": 1, "student_name": "John Doe", "student_email": "john.doe@example.com", "student_phone": "123-456-7890", "student_dob": "1995-10-15", "student_age": 27, "student_address": "123 Main Street, CityA, CountryX" }
```

```
{ "_id": ObjectId('64be9cb9066fe416c2a44c85'), "student_id": 2, "student_name": "Jane Smith", "student_email": "jane.smith@example.com" }
```

MongoDB Compass - localhost:27017/hafeez\_db.students

Connect Edit View Help

localhost:27017 Documents hafeez\_db.stude...

My Queries Databases Search

- admin
- config
- hafeez\_db
  - employees
  - students**
- local

### hafeez\_db.students

3 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options

ADD DATA EXPORT DATA 1 - 3 of 3

	_id ObjectId	student_id Int32	student_name String	student_email String	student_phone String	student_dob String	
1	ObjectId('64bed775d87dcf7b7801...')	1	"John Doe"	"john.doe@example.com"	"123-456-7890"	"1995-10-15"	
2	ObjectId('64bed775d87dcf7b7801...')	2	"Jane Smith"	"jane.smith@example.com"	"987-654-3210"	"1998-05-20"	
3	ObjectId('64bed775d87dcf7b7801...')	3	"Michael Johnson"	"michael.johnson@example.com"	"555-123-4567"	"1997-02-08"	

{student\_email:'michael.johnson@example.com'}

The screenshot shows the MongoDB Compass interface. The left sidebar displays the database structure: localhost:27017, hafeez\_db, and the students collection. The main panel shows the hafeez\_db.students collection with 3 documents and 1 index. A filter query is applied: {student\_email:'michael.johnson@example.com'}. Below the filter, there are buttons for 'ADD DATA' and 'EXPORT DATA'. The document being displayed is:

```
{
  "_id": ObjectId('64be9cb9066fe416c2a44c86'),
  "student_id": 3,
  "student_name": "Michael Johnson",
  "student_email": "michael.johnson@example.com",
  "student_phone": "555-123-4567",
  "student_dob": "1997-02-08",
  "student_age": 26,
  "student_address": "789 Oak Road, CityC, CountryZ"
}
```

# Filter Query Single Value



MongoDB Compass - localhost:27017/hafeez\_db.students

Connect Edit View Help

localhost:27017

Documents  
hafeez\_db.stude...

My Queries

Databases

Search

admin

config

hafeez\_db

employees

students

local

### hafeez\_db.students

Documents Aggregations Schema Indexes Validation

3 DOCUMENTS 1 INDEXES

Filter

```
{
  student_name: "John Doe",
  student_email: "john.doe@example.com"
}
```

INSIGHT Explain Reset Find </> Options

ADD DATA EXPORT DATA

1-1 of 1

```
{
  "_id": { },
  "student_id": 1,
  "student_name": "John Doe",
  "student_email": "john.doe@example.com",
  "student_phone": "123-456-7890",
  "student_dob": "1995-10-15",
  "student_age": 27,
  "student_address": "123 Main Street, CityA, CountryX"
}
```

# Filter Query Multiple Values

# View Data Sort

The screenshot shows the MongoDB Compass interface for the `localhost:27017/hafeez_db.employees` collection. The left sidebar shows the database structure with `employees` selected. The main panel displays the `Documents` tab for `hafeez_db.employees`, showing 3 documents and 1 index. The `Filter` field is empty, and the `Sort` field is set to `{employee_id: -1}`. The `Options` button is highlighted with a red arrow. Below the query fields, the `Collation` is set to `{ locale: 'simple' }`. The `MaxTimeMS` is 60000, `Skip` is 0, and `Limit` is 0. The bottom of the interface shows the first document in the collection:

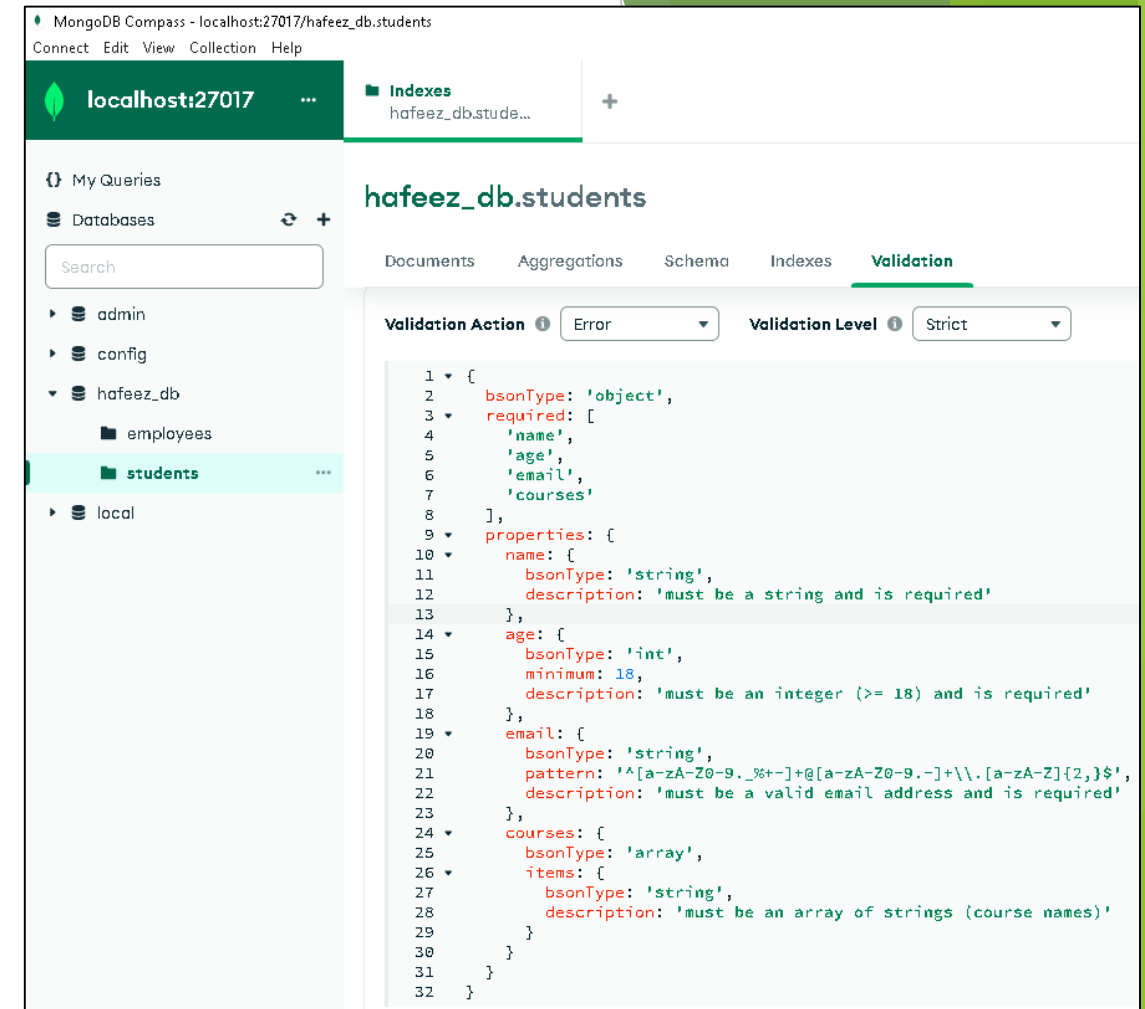
```
{ "_id": ObjectId('64bea5e7066fe416c2a44c96'),  
  "employee_id": 3,  
  "employee_name": "Michael Johnson",  
  "employee_email": "michael.johnson@example.com",  
  "employee_phone": "555-123-4567",  
  "employee_dob": "1988-12-08",  
  "employee_age": 33,  
  "employee_address": "789 Oak Road, CityC, CountryZ"
```

Buttons for `CANCEL` and `UPDATE` are visible at the bottom right.

```

{
  "bsonType": "object",
  "required": ["name", "age", "email", "courses"],
  "properties": {
    "name": {
      "bsonType": "string",
      "description": "must be a string and is required",
    },
    "age": {
      "bsonType": "int",
      "minimum": 18,
      "description": "must be an integer (>= 18) and is required",
    },
    "email": {
      "bsonType": "string",
      "pattern": "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$",
      "description": "must be a valid email address and is required",
    },
    "courses": {
      "bsonType": "array",
      "items": {
        "bsonType": "string",
        "description": "must be an array of strings (course names)",
      },
    },
  },
}

```



## Insert Data - Validation Rules

```
[
  {
    "student_id": 1,
    "student_name": "John Doe",
    "student_email": "john.doe@example.com",
    "student_phone": "123-456-7890",
    "student_dob": "1995-10-15",
    "student_age": 27,
    "student_address": "123 Main Street, CityA, CountryX"
  },
  {
    "student_id": 2,
    "student_name": "Jane Smith",
    "student_email": "jane.smith@example.com",
    "student_phone": "987-654-3210",
    "student_dob": "1998-05-20",
    "student_age": 24,
    "student_address": "456 Park Avenue, CityB, CountryY"
  },
  {
    "student_id": 3,
    "student_name": "Michael Johnson",
    "student_email": "michael.johnson@example.com",
    "student_phone": "555-123-4567",
    "student_dob": "1997-02-08",
    "student_age": 26,
    "student_address": "789 Oak Road, CityC, CountryZ"
  }
]
```



**students.json**

```
[
  {
    "employee_id": 1,
    "employee_name": "John Doe",
    "employee_email": "john.doe@example.com",
    "employee_phone": "123-456-7890",
    "employee_dob": "1985-08-15",
    "employee_age": 37,
    "employee_address": "123 Main Street, CityA, CountryX"
  },
  {
    "employee_id": 2,
    "employee_name": "Jane Smith",
    "employee_email": "jane.smith@example.com",
    "employee_phone": "987-654-3210",
    "employee_dob": "1990-03-20",
    "employee_age": 32,
    "employee_address": "456 Park Avenue, CityB, CountryY"
  },
  {
    "employee_id": 3,
    "employee_name": "Michael Johnson",
    "employee_email": "michael.johnson@example.com",
    "employee_phone": "555-123-4567",
    "employee_dob": "1988-12-08",
    "employee_age": 33,
    "employee_address": "789 Oak Road, CityC, CountryZ"
  }
]
```



employees.json

```
{
  bsonType: 'object',
  required: [
    'name',
    'age',
    'email',
    'courses'
  ],
  properties: {
    name: {
      bsonType: 'string',
      description: 'must be a string and is required'
    },
    age: {
      bsonType: 'int',
      minimum: 18,
      description: 'must be an integer (>= 18) and is required'
    },
    email: {
      bsonType: 'string',
      pattern: '^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$',
      description: 'must be a valid email address and is required'
    },
    courses: {
      bsonType: 'array',
      items: {
        bsonType: 'string',
        description: 'must be an array of strings (course names)'
      }
    }
  }
}
```



students-validation-rules.json

# Using CMD use MongoDB

- ▶ mongo
- ▶ show dbs
- ▶ use mydatabase
- ▶ show collections

```
C:\Windows\system32\cmd.exe - mongo
Microsoft Windows [Version 10.0.19045.3208]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Abdul Hafeez>mongo
MongoDB shell version v4.2.13
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("adc5159e-a459-443f-b2d8-ddef1cf2ab34") }
MongoDB server version: 6.0.8
WARNING: shell and server versions do not match
Server has startup warnings:
{"t":{"$date":"2023-07-24T16:54:21.230+05:00"},"s":"W", "c":"CONTROL", "id":22120, "ctx":"initandlisten","msg":"Access control is not enabled for the database. Read and write access to data and configuration is unrestricted","tags":["startupWarnings"]}

> show dbs
admin          0.000GB
config         0.000GB
hafeez_db      0.000GB
local          0.000GB
> use hafeez_db
switched to db hafeez_db
> show collections
employees
students
>
```

# Using CMD use MongoDB

- ▶ `mongo`
- ▶ `use hafeez_db`
- ▶ `db.createCollection("users")`
- ▶ `db.users.insertOne({ name: "John Doe", age: 30, email: "john@example.com" })`
- ▶ `db.users.insertMany([  
 { name: "Alice", age: 25, email: "alice@example.com" },  
 { name: "Bob", age: 28, email: "bob@example.com" },  
 { name: "Eve", age: 22, email: "eve@example.com" }  
])`
- ▶ `db.users.find()`
- ▶ `db.users.find().limit(2)`
- ▶ `db.users.find({ name: "John Doe" })`
- ▶ `db.users.updateOne({ name: "John Doe" }, { $set: { age: 31 } })`
- ▶ `db.users.updateMany({ name: "John Doe" }, { $set: { age: 31 } })`
- ▶ `db.users.deleteOne({ age: 30 })`
- ▶ `db.users.deleteMany({ age: 30 })`
- ▶ `db.users.find().sort({ age: 1 }) // 1=> ASC & -1 => DESC`
- ▶ `db.users.find().sort({ age: 1, name: -1 })`



```
mongo
use hafeez_db;

db.createCollection("students", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["name", "age", "email", "courses"],
      properties: {
        name: {
          bsonType: "string",
          description: "must be a string and is required",
        },
        age: {
          bsonType: "int",
          minimum: 18,
          description: "must be an integer (>= 18) and is required",
        },
        email: {
          bsonType: "string",
          pattern: "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$",
          description: "must be a valid email address and is required",
        },
        courses: {
          bsonType: "array",
          items: {
            bsonType: "string",
            description: "must be an array of strings (course names)",
          },
        },
      },
    },
  },
});
```

## MongoDB Validation Rules Using CMD

### Example 01

```
db.runCommand({
  collMod: "students",
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["name", "age", "email", "courses"],
      properties: {
        name: {
          bsonType: "string",
          description: "must be a string and is required",
        },
        age: {
          bsonType: "int",
          minimum: 18,
          description: "must be an integer (>= 18) and is required",
        },
        email: {
          bsonType: "string",
          pattern: "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$",
          description: "must be a valid email address and is required",
        },
        courses: {
          bsonType: "array",
          items: {
            bsonType: "string",
            description: "must be an array of strings (course names)",
          },
        },
      },
    },
  },
  validationLevel: "strict",
  validationAction: "error",
});
```

## MongoDB Validation Rules Using CMD

### Example 02