

Basics of NoSQL Databases - MongoDB

सीबीएसई पाठ्यक्रम पर आधारित
कक्षा -11



द्वारा:
संजीव भदौरिया
स्नातकोत्तर शिक्षक (संगणक विज्ञान)
के० वि० बाराबंकी (लखनऊ संभाग)

परिचय

- अभी तक हमने जितने databases के बारे में जान वे सभी SQL आधारित databases हैं, जिनमे एक table, row, fields ,records इत्यादि होते हैं ।
- लेकिन बिना structure या record के भी databases संभव हैं - NoSQL अर्थात Not Only SQL Databases ऐसे ही database होते हैं ।
- इस अध्याय में हम ऐसे ही NoSQL databases के बारे में जानेंगे ।

NoSQL Databases

- ये एक प्रकार के non-relational database होते हैं जिनका कोई प्रतिबंधित(Strict) या दृढ़ (rigid) आकार नहीं होता |
- ये पारंपरिक table के अधर पर record को नहीं store करते हैं |
- ये clusters में run करते हैं और web के पैमाने पर data को store कर सकते हैं | इनकी scalability बहुत होती है | ये एक प्रकार से सामान्य भाषा में bigdata कहे जाते हैं|
- इस प्रकार के databases को प्रयोग करने वाले अपने बहुत से app या web apps देखी होंगी जैसे Google Mail, Google Earth, Ebay, LinkedIn, facebook, Amazon इत्यादि |
- पूरे विश्व के users को बहुत fast response time मिलता है |
- सभी प्रकार के डाटा को handle करने में सक्षम है वह भी बिना किसी रोकटोक के |
- नए feature और fast update को तुरंत adopt कर लेता है |
- इसमें down time नहीं आता अर्थात हमेशा performance देता है |

NoSQL Databases के प्रकार

1. Key-value Databases
2. Document Databases
3. Column family stores Databases
4. Graph Databases

Key-Value databases

- ठीक वैसे ही होते हैं जैसे python dictionary.
- ये बहुत ही साधारण और flexible होते हैं |
- इनके उदाहरण – *Cassandra, Amazon DyanmoDB, ATS (Azure Table Storage, Riak, BerkeleyDB* हैं |

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

Document Databases

- ये key-value databases के आगे के रूप होते हैं |
- इसमें key-value pair एक structured या semi-structured रूप में document के फॉर्म में store होती है |
- इसमें keys सदैव string के रूप में होती हैं और values किसी भी प्रकार की हो सकती हैं|
- यह MS office document, PDFs, XML, JSON ,BSON के रूप में हो सकती है |
- JSON (JavaScript Object Notation) तथा BSON (Binary JSON)
- JSON एक open, मानव तथा मशीन द्वारा समझा जाने वाला standard होता है और modern web पर data interchange के लिए इसका मुख्य फॉर्मेट XML होता है |
- JSON का प्रयोग हमने Python dictionaries अध्याय में पढ़ा था|
- इसके उदहारण - MongoDB, Couch DB DocumentDB इत्यादि है |

```
{  
  "Title": "The Cuckoo's Calling",  
  "Author": "Robert Galbraith",  
  "Genre": "classic crime novel",  
  "Detail": {  
    "Publisher": "Little Brown",  
    "Publication_Year": 2013,  
    "ISBN-13": 9781408704004,  
    "Language": "English",  
    "Pages": 494  
  },  
  "Price": [  
    {  
      "type": "Hardcover",  
      "price": 16.65  
    },  
    {  
      "type": "Kidle Edition",  
      "price": 7.03  
    }  
  ]  
}
```

Column Family Store Database

- इन्हें column store या column family database कहा जाता है तथा यह column oriented मॉडल होता है।
- Column family एक storage mechanism होती है जिसमें –
 - कई rows हो सकती हैं।
 - प्रत्येक row में कई column हो सकते हैं।
 - इसमें एक row key होती है जिसके अंतर्गत उस row में कई कोलुम्न्स हो सकते हैं जैसे की बगल वाले चित्र में दिखाया गया है।
 - इसके उदाहरण - Hbase, Cassandra, HyperTable इत्यादि हैं।

UserProfile

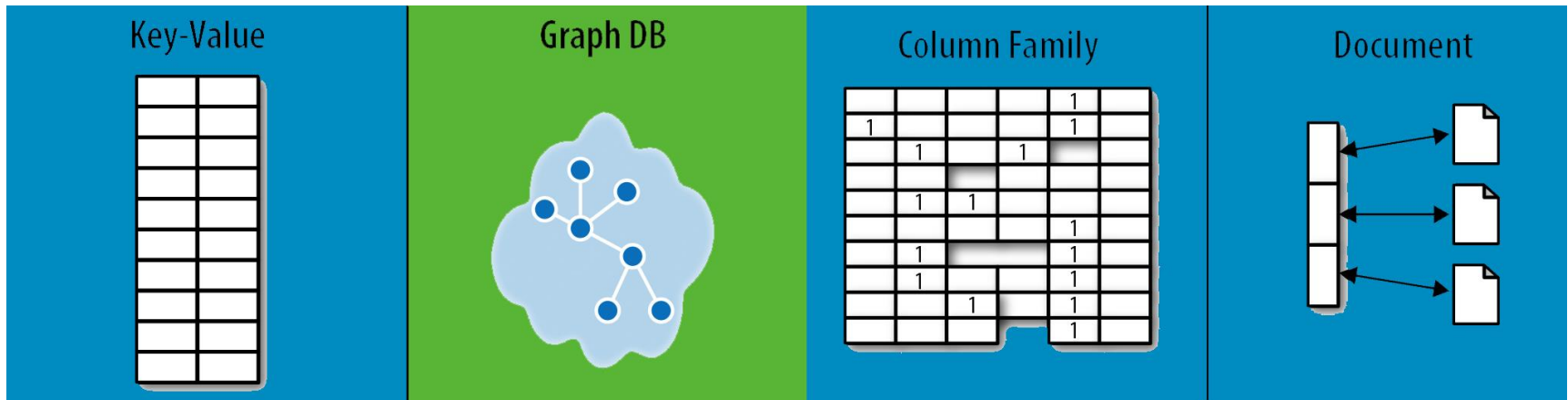
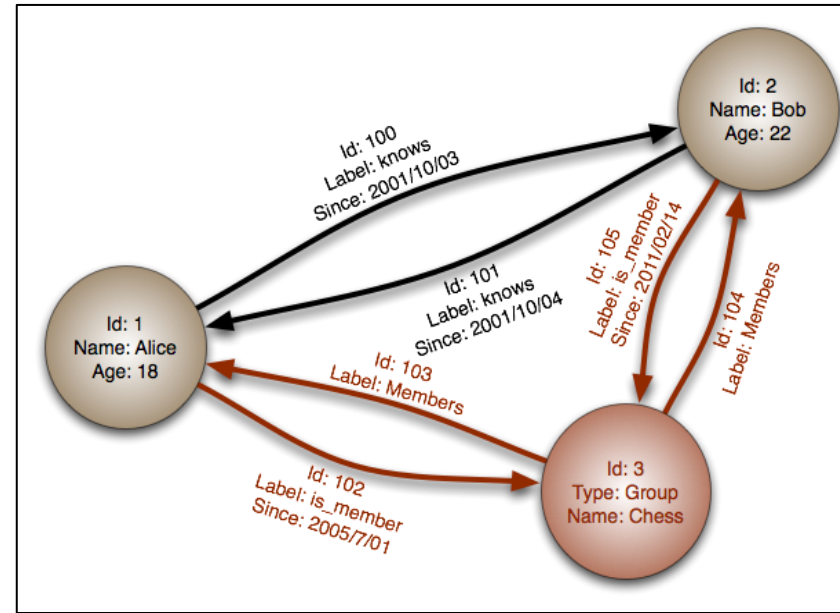
	emailAddress	gender	age
Bob	bob@example.com	male	35
	1465676582	1465676582	1465676582

	emailAddress	gender
Britney	brit@example.com	female
	1465676432	1465676432

	emailAddress	country	hairColor
Tori	tori@example.com	Sweden	Blue
	1435636158	1435636158	1465633654

Graph Database

- यह डाटा को store करने के लिए grafical मॉडल का प्रयोग करता है |
- यहाँ object को प्रदर्शित करने के लिए **nodes** का प्रयोग किया जाता है जबकि **edges** का प्रयोग उन nodes के मध्य relation दर्शाने के लिए किया जाता है |
- इसके उदहारण - Neo4j, Blazegraph, Titan इत्यादि है |



NoSQL Databases के Advantages और Disadvantages

•Advantages:

–Flexible Data Model

ये बहुत flexible database होते हैं जिनमे किसी भी type का डाटा store किया जा सकता है ।

–Evolving Data Model

बिना system को down किये आप schema में बदलाव कर सकते हैं।

–Elastic Scalability

कम कीमत पर बहुत बड़ा database store किया जा सकता है ।

–High Performance

इसका throughput और latency दोनों का समय बहुत कम होता है ।

–Open Source

इसके लिए किसी प्रकार का भुगतान करने की आवश्यकता नहीं और अपने अनुसार बदलाव भी संभव है ।

•Disadvantages:

–Lack of Standardization

NoSQL database को मानक प्रदान करने के कोई नियम नहीं हैं ।

–Backup of Database

NoSQL databases में सबसे बड़ी कमी backup की है हालाँकि MongoDB backup के लिए tool प्रदान करता है परन्तु वह अभी उतना अच्छा नहीं है ।

–Consistency

NoSQL database, consistency को performance और scalability के आगे गंभीरता से नहीं लेता है । अर्थात यहाँ डाटा की duplicacy आसानी से हो जाती है ।

MongoDB के साथ कार्य करना

- MongoDB एक document-oriented NoSQL database है |
- यह dynamic schemas को सपोर्ट करता है जोकि JSON फॉर्मेट में डाटा को प्रदर्शित करता है |
- यह एक free open source software है जो high scalability और high performance देता है |

MongoDB Terminology

MongoDB Term	Description	SQL Term
Field	एक name-value pair जो एक प्रकार की information रखती है	Column
Document	Locally related fields का समूह	Row/record
Collection	Related documents का समूह	Table
Database	Collections का container. एक MongoDB server में कई database हो सकते हैं	Database
Primary key	Unique field जो document को identify करती है	Primary key

MongoDB को install करना

- निम्न लिंक को ब्राउज़र पर पेस्ट करें या इसको open करें |

<https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/#install-mdb-edition>

The screenshot shows the MongoDB Download Center website. The URL in the browser is <https://www.mongodb.com/download-center/community?jmp=docs>. The page features a navigation bar with links for DOCS, LEARN, WHAT'S MONGODB?, BLOG, LOGIN, SOLUTIONS, CLOUD, CUSTOMERS, RESOURCES, and ABOUT US. A green button labeled "Get MongoDB" is visible in the top right. The main heading is "MongoDB Download Center". Below this, there are tabs for "Cloud", "Servers", and "Tools", with "Servers" being the active tab. The text "Select the server you would like to run:" is displayed. Two server options are shown: "MongoDB Community Server" (FEATURE RICH. DEVELOPER READY.) and "MongoDB Enterprise Server" (ADVANCED FEATURES. PERFORMANCE GRADE.). Below the server selection, there are dropdown menus for "Version" (4.0.3 (current release)) and "OS" (Windows 64-bit x64). A "Package" dropdown menu is set to "ZIP". A green "Download" button is present. A blue callout box with white text points to the "Package" dropdown menu, containing the text "यहाँ से msi version डाउनलोड करें।". A list of links is visible on the right side of the page, including "Release notes", "Changelog", "All version binaries", "Installation Instructions", "Download source (tgz)", and "Download source (zip)".

MongoDB को install करना

- MSI file को open करके MongoDB install करिए |
- जब इनस्टॉल हो जाये तो निम्न path पर जाकर check कर लें कि mongod.exe file और mongo.exe file हैं या नहीं।

`C:\Program Files\MongoDB\Server\4.0\bin`

- उसके बाद निम्न `c:\` पर data फोल्डर और उसके अन्दर db फोल्डर बनायें | अर्थात् **`c:\data\db`**
- अब वापस command window से `C:\Program Files\MongoDB\Server\4.0\bin` location पर जाकर mongod को run करें और जब mongod run हो जाये तो उसे बंद न करें |
- अब दूसरी command window open कर उसी path पर mongo को run कर दीजिये |

MongoDB को start करना

 mongodb-win32-x86_64-2008plus-ssl-4.0.3-signed 10/30/2018 8:57 AM Windows Installer ... 191,781 KB

ये है mongodb का installer जो लगभग 190 MB का है इसको इनस्टॉल करने पर यह program files में install हो जाता है |
मेरे कंप्यूटर में यह "C:\Program Files\MongoDB\Server\4.0\bin" path पर इनस्टॉल हुआ है आप अपने system में path देख लीजिये | इसे हम windows के path में भी जोड़ सकते हैं |

```
mongod
te access to data and configuration is unrestricted.
2018-10-30T09:14:31.414-0700 I CONTROL [initandlisten]
2018-10-30T09:14:31.415-0700 I CONTROL [initandlisten] ** WARNING: This server
is bound to localhost.
2018-10-30T09:14:31.416-0700 I CONTROL [initandlisten] ** Remote system
ms will be unable to connect to this server.
2018-10-30T09:14:31.417-0700 I CONTROL [initandlisten] ** Start the se
rver with --bind_ip <address> to specify which IP
2018-10-30T09:14:31.417-0700 I CONTROL [initandlisten] ** addresses it
should serve responses from, or with --bind_ip_all to
2018-10-30T09:14:31.418-0700 I CONTROL [initandlisten] ** bind to all
interfaces. If this behavior is desired, start the
2018-10-30T09:14:31.419-0700 I CONTROL [initandlisten] ** server with
--bind_ip 127.0.0.1 to disable this warning.
2018-10-30T09:14:31.419-0700 I CONTROL [initandlisten]
2018-10-30T09:14:31.669-0700 W FTDC [initandlisten] Failed to initialize Per
formance Counters for FTDC: WindowsPdhError: PdhExpandCounterPathW failed with '
The specified object was not found on the computer.' for counter '\Memory\Availa
ble Bytes'
2018-10-30T09:14:31.669-0700 I FTDC [initandlisten] Initializing full-time d
iagnostic data capture with directory 'C:\data\db\diagnostic.data'
2018-10-30T09:14:31.671-0700 I NETWORK [initandlisten] waiting for connections
on port 27017
```

```
mongo
Server has startup warnings:
2018-10-30T08:59:52.729-0700 I CONTROL [initandlisten]
2018-10-30T08:59:52.729-0700 I CONTROL [initandlisten] ** WARNING: Access contr
ol is not enabled for the database.
2018-10-30T08:59:52.730-0700 I CONTROL [initandlisten] ** Read and writ
te access to data and configuration is unrestricted.
2018-10-30T08:59:52.730-0700 I CONTROL [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive an
d display
metrics about your deployment (disk utilization, CPU, operation statistics, etc)
.
The monitoring data will be available on a MongoDB website with a unique URL acc
essible to you
and anyone you share the URL with. MongoDB may use this information to make prod
uct
improvements and to suggest MongoDB products and deployment options to you.
To enable free monitoring, run the following command: db.enableFreeMonitoring(<>).
To permanently disable this reminder, run the following command: db.disableFreeM
onitoring(<>)
---
```

Mongo को run होने के लिए mongod का पहले से running state में होना आवश्यक है | अब आप mongo पर commands देने के लिए तैयार हैं |

MongoDB Data Types

S.N.	Data Type	Data Type Number	S.N.	Data Type	Data Type Number
1.	Double	1	10.	Null	11
2.	String	2	11.	Regular Expression	12
3.	Object	3	12.	JavaScript	13
4.	Array	4	13.	Symbol	14
5.	Binary Data	5	14.	JavaScript with scope	15
6.	Undefined	6	15.	Integer	16 and 18
7.	Object Id	7	16.	Timestamp	10
8.	Boolean	9	17.	Min Key	255
9.	Date	10	18.	Max Key	127

MongoDB के basic commands

• Database create करना →

MongoDB में अलग से database create करने की आवश्यकता नहीं होती है | जैसे ही आप पहली information database में insert करते हैं तो database स्वतः तैयार हो जाता है |

• Current Database को प्रदर्शित करना →

>show dbs

यह database को शो करेगा

>show collections

यह current database में collections को शो करेगा

• Database को use करना →

>use mydb

• CRUD operations →

ये operations निम्न हैं -

Create

Read

Update

Delete

इन्हें CRUD operation कहा जाता है |

```
> db
test
> show dbs
admin    0.000GB
config  0.000GB
local   0.000GB
> db
test
> show collections
```

MongoDB के basic commands

• Save operation द्वारा database तैयार करना →

- MongoDB में अलग से database create करने की आवश्यकता नहीं होती है | जैसे ही आप पहली information database में insert करते हैं तो database स्वतः तैयार हो जाता है |
- आप collection में डाटा save या insert कमांड के द्वारा प्रविष्ट करा सकते हैं |
db.<collection-name>.save({<document details>})
- बाद में हम show collections कमांड चलाकर देख सकते हैं कि collection बना या नहीं |
- **>USE <DatabaseName>** से भी database बना सकते हैं |
- निम्न उदहारण में school database बनाकर उसमे 1 collection insert कराया गया है |

```
> use school
switched to db school
> db.student.save({name:'Pankaj'})
WriteResult({ "nInserted" : 1 })
> show collections
student
>
```

MongoDB के basic commands

• Save operation द्वारा database तैयार करना →

- इसके द्वारा हम एक साथ कई documents भी insert कर सकते हैं ।

```
> db.student.save([<name:'Suresh'>,<name:'Hari',age:23>])
BulkWriteResult<<
  "writeErrors" : [ ],
  "writeConcernErrors" : [ ],
  "nInserted" : 2,
  "nUpserted" : 0,
  "nMatched" : 0,
  "nModified" : 0,
  "nRemoved" : 0,
  "upserted" : [ ]
>>
```

ध्यान दीजिये की
यहाँ 2 document
insert हुए हैं ।

- जब आप document insert करते हैं तब mongoDB स्वतः एक field और जोड़ देता है “_id” जिसकी value वह स्वयं बढ़ते क्रम में set करता है । और यह प्रक्रिया हमें दिखाई नहीं देती है । यदि हम चाहें तो “_id” की value हम स्वयं insert करते समय provide कर सकते हैं ।

• Save या insert का प्रयोग करके यदि आप किसी document को insert करते हैं और आपके दिए गए database और collection से नाम नहीं मिलता है तो mongoDB इनके लिए नया database बना देता है ।

MongoDB के basic commands

• Insert operation द्वारा database तैयार करना →

- आप collection में डाटा insert कमांड के द्वारा भी प्रविष्ट करा सकते हैं।
`db.<collection-name>.insert({<document details>})`
- बाद में हम show collections कमांड चलाकर देख सकते हैं कि collection बना या नहीं।
- निम्न उदहारण में school database बनाकर उसमे 1 collection insert कराया गया है।

```
> db.teachers.insert([<name:'Roop Narayan'>])
BulkWriteResult<<
  "writeErrors" : [ ],
  "writeConcernErrors" : [ ],
  "nInserted" : 1,
  "nUpserted" : 0,
  "nMatched" : 0,
  "nModified" : 0,
  "nRemoved" : 0,
  "upserted" : [ ]
>>
```

- इसके द्वारा भी save की भांति multiple documents को insert करने के लिए निम्न कमांड प्रयोग करते हैं -

```
>db.teachers.insertMany([<name:'Ratan'>,<name:'Krishna',age:45>])
```

MongoDB के basic commands

- Object Create करके भी documents को insert किया जा सकता

है

```
> stud={name:'Ojas',age:12,city:'Barabanki'}
< "name" : "Ojas", "age" : 12, "city" : "Barabanki" >
> db.student.insert(stud)
WriteResult(<< "nInserted" : 1 >>
>
```

यहाँ stud एक valid mongoDB ऑब्जेक्ट है |

- एक object में कोई field ऐसी भी हो सकती है जो खुद एक object हो |

```
> addr={Hno:113, Vill:'Sangram Kheda',post:'Gulariha'}
< "Hno" : 113, "Vill" : "Sangram Kheda", "post" : "Gulariha" >
> stud={name:'Mohit',age:24,address:addr}
<
  "name" : "Mohit",
  "age" : 24,
  "address" : <
    "Hno" : 113,
    "Vill" : "Sangram Kheda",
    "post" : "Gulariha"
  >
>
> db.student.insert(stud)
WriteResult(<< "nInserted" : 1 >>
>
```

यहाँ stud में एक field है address जिसकी value एक ऑब्जेक्ट है addr |

MongoDB के basic commands

- Object में array भी हो सकते हैं | उदहारण के लिए –

Name: Himanshu

Class:11

Section: A

Subjects: English, Hindi, Maths, Physics, Chemistry

यहाँ Subjects एक array है |

```
> newstud={
... name:'Himanshu',
... class:11,
... Sec:'A',
... Subjects:['English','Hindi','Maths','Physics','Chemistry']
... }
{
  "name" : "Himanshu",
  "class" : 11,
  "Sec" : "A",
  "Subjects" : [
    "English",
    "Hindi",
    "Maths",
    "Physics",
    "Chemistry"
  ]
}
> db.student.save(newstud)
WriteResult({ "nInserted" : 1 })
```

MongoDB के basic commands

•Read Operation:

Read operation का प्रयोग database के collection से documents को access करने के लिए किया जाता है | जिसके लिए निम्न syntaxes ला प्रयोग करते हैं|

>db.<collection-name>.find() collection के सारे documents को show करेगा |

>db.<collection-name>.findOne() यह सिर्फ एक record show करेगा |

>db.<collection-name>.findOne({<key>:<value>}) यह search criteria के जैसे कार्य करता है |

```
> use school
switched to db school
> show collections
student
teachers
> db.student.find()
{ "_id" : ObjectId("5be1255c6b46969b847e3f5f"), "name" : "Pankaj" }
{ "_id" : ObjectId("5be446c3f29d9be663cbee3c"), "name" : "Suresh" }
{ "_id" : ObjectId("5be446c3f29d9be663cbee3d"), "name" : "Hari", "age" : 23 }
{ "_id" : ObjectId("5be44b54f29d9be663cbee3f"), "name" : "Ojas", "age" : 12, "ci
ty" : "Barabanki" }
{ "_id" : ObjectId("5be44ce9f29d9be663cbee40"), "name" : "Mohit", "age" : 24, "a
ddress" : { "Hno" : 113, "Vill" : "Sangram Kheda", "post" : "Gulariha" } }
{ "_id" : ObjectId("5be44ee2f29d9be663cbee41"), "name" : "Himanshu", "class" : 1
1, "Sec" : "A", "Subjects" : [ "English", "Hindi", "Maths", "Physics", "Chemistr
y" ] }
> db.student.findOne()
{ "_id" : ObjectId("5be1255c6b46969b847e3f5f"), "name" : "Pankaj" }
> db.student.findOne({name:'Suresh'})
{ "_id" : ObjectId("5be446c3f29d9be663cbee3c"), "name" : "Suresh" }
```

यदि कोई record match नहीं होता है तो null return होता है |

MongoDB के basic commands

•Read Operation:

```
> db.student.find().pretty()
<
  "_id" : ObjectId<"5be1255c6b46969b847e3f5f">, "name" : "Pankaj" >
<
  "_id" : ObjectId<"5be446c3f29d9be663cbee3c">, "name" : "Suresh" >
<
  "_id" : ObjectId<"5be446c3f29d9be663cbee3d">,
  "name" : "Hari",
  "age" : 23
>
<
  "_id" : ObjectId<"5be44b54f29d9be663cbee3f">,
  "name" : "Ojas",
  "age" : 12,
  "city" : "Barabanki"
>
<
  "_id" : ObjectId<"5be44ce9f29d9be663cbee40">,
  "name" : "Mohit",
  "age" : 24,
  "address" : {
    "Hno" : 113,
    "Vill" : "Sangram Kheda",
    "post" : "Gulariha"
  }
>
<
  "_id" : ObjectId<"5be44ee2f29d9be663cbee41">,
  "name" : "Himanshu",
  "class" : 11,
  "Sec" : "A",
  "Subjects" : [
    "English",
    "Hindi",
    "Maths",
    "Physics",
    "Chemistry"
  ]
1
>
>
```

pretty() documents को सही प्रकार से सही indentation में JSON फॉर्मेट में print कर देता है।

MongoDB के basic commands

•Read Operation:

```
> db.student.find(<>,{name:1})
< {"_id" : ObjectId("5be1255c6b46969b847e3f5f"), "name" : "Pankaj" }
< {"_id" : ObjectId("5be446c3f29d9be663cbee3c"), "name" : "Suresh" }
< {"_id" : ObjectId("5be446c3f29d9be663cbee3d"), "name" : "Hari" }
< {"_id" : ObjectId("5be44b54f29d9be663cbee3f"), "name" : "Ojas" }
< {"_id" : ObjectId("5be44ce9f29d9be663cbee40"), "name" : "Mohit" }
< {"_id" : ObjectId("5be44ee2f29d9be663cbee41"), "name" : "Himanshu" }
>
```

ऊपर के उदाहरण के अनुसार कमांड देने पर सिर्फ name field ही प्रदर्शित होगी वह भी “_id” के साथ ।

यदि “_id” प्रदर्शित नहीं करना चाहते तो निम्न प्रकार कमांड देना होगा ।

```
> db.student.find(<>,{name:1,_id:0})
< {"name" : "Pankaj" }
< {"name" : "Suresh" }
< {"name" : "Hari" }
< {"name" : "Ojas" }
< {"name" : "Mohit" }
< {"name" : "Himanshu" }
```

MongoDB के basic Operators

•Comparison Operator:

अन्य databases की तरह mongoDB भी operators प्रदान करता है ताकि हम सही से delete, read या update operation को perform कर पायें |

Operator Name	Meaning
\$eq	Equal to
\$gt	Greater than
\$gte	Greater than or equal to
\$lt	Less than
\$lte	Less than or equal to
\$ne	Not equal to

```
> db.student.find(<age:<$gt:23>>).pretty()
<
  "_id" : ObjectId("5be44ce9f29d9be663cbee40"),
  "name" : "Mohit",
  "age" : 24,
  "address" : <
    "Hno" : 113,
    "Vill" : "Sangram Kheda",
    "post" : "Gulariha"
  >
}
> db.student.find(<age:<$gte:23>>).pretty()
<
  "_id" : ObjectId("5be446c3f29d9be663cbee3d"),
  "name" : "Hari",
  "age" : 23
}
<
  "_id" : ObjectId("5be44ce9f29d9be663cbee40"),
  "name" : "Mohit",
  "age" : 24,
  "address" : <
    "Hno" : 113,
    "Vill" : "Sangram Kheda",
    "post" : "Gulariha"
  >
}
}
```

MongoDB के basic Operators

- Comparison Operator:

यदि conditional based या range देना हो तो निम्न प्रकार देंगे |

```
{field:{$gte:<lower value>, $lte:<upper value>}}
```

```
> db.student.find(<age:{$gte:23,$lte:24}>).pretty()
{
  "_id" : ObjectId("5be446c3f29d9be663cbee3d"),
  "name" : "Hari",
  "age" : 23
}
{
  "_id" : ObjectId("5be44ce9f29d9be663cbee40"),
  "name" : "Mohit",
  "age" : 24,
  "address" : {
    "Hno" : 113,
    "Vill" : "Sangram Kheda",
    "post" : "Gulariha"
  }
}
}
```


MongoDB के basic Operators

•Condition based on List/Array

```
{ field :{ $in : [ val1,val2, . . . . . ] } }
```

```
{ field :{ $nin : [ val1,val2, . . . . . ] } }
```

Operator Name	Meaning
\$in	In
\$nin	Not In

```
> db.student.find(<<Sec:<<$in:['A','C']>>>).pretty()
{
  "_id" : ObjectId("5be44ee2f29d9be663cbee41"),
  "name" : "Himanshu",
  "class" : 11,
  "Sec" : "A",
  "Subjects" : [
    "English",
    "Hindi",
    "Maths",
    "Physics",
    "Chemistry"
  ]
}
```

Section match करने पर डाटा show किया लेकिन match न करने पर डाटा नहीं दिखाया |

```
> db.student.find(<<Sec:<<$in:['B','C']>>>).pretty()
>
```

MongoDB के basic Operators

• Logical Query Operators

```
{ field : { $not : { <op-Exp> } }  
{ field : { $and : [ { <op-Exp> }, { <op-Exp> }, ... ] } }  
{ field : { $or : [ { <op-Exp> }, { <op-Exp> }, ... ] } }
```

Operator Name	Meaning
\$not	Logical NOT
\$and	Logical AND
\$or	Logical OR

```
> db.student.find(<<age:<<$not:<<$gt:23>>>>).pretty(<>  
< "_id" : ObjectId<"5be1255c6b46969b847e3f5f">, "name" : "Pankaj" >  
< "_id" : ObjectId<"5be446c3f29d9be663cbee3c">, "name" : "Suresh" >  
<  
<   "_id" : ObjectId<"5be446c3f29d9be663cbee3d">, >  
<   "name" : "Hari", >  
<   "age" : 23 >  
<  
<  
<   "_id" : ObjectId<"5be44b54f29d9be663cbee3f">, >  
<   "name" : "Ojas", >  
<   "age" : 12, >  
<   "city" : "Barabanki" >  
<  
<  
<   "_id" : ObjectId<"5be44ee2f29d9be663cbee41">, >  
<   "name" : "Himanshu", >  
<   "class" : 11, >  
<   "Sec" : "A", >  
<   "Subjects" : [ >  
<     "English", >  
<     "Hindi", >  
<     "Maths", >  
<     "Physics", >  
<     "Chemistry" >  
<   ] >  
<  
> db.student.find(<<$and:[<<age:<<$lt:23>>>, <<age:<<$gt:11>>>] >>).pretty(<>  
<  
<   "_id" : ObjectId<"5be44b54f29d9be663cbee3f">, >  
<   "name" : "Ojas", >  
<   "age" : 12, >  
<   "city" : "Barabanki" >  
<  
<
```

•Update Operation:

Update operation का प्रयोग दो तरह से किया जाता है |

>update/updateOne अथवा >updateMany (इनके साथ \$set operator प्रयोग किया जाता है)

```
>db.<CollectionName>.update/updateOne({query-exp},{ $set:{<field1>:<val1>, . . . }})
```

```
> db.student.update({age:12},{ $set:{name:'Hari Prakash'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.student.find().pretty()
{ "_id" : ObjectId<"5be1255c6b46969b847e3f5f">, "name" : "Pankaj" }
{ "_id" : ObjectId<"5be446c3f29d9be663cbee3c">, "name" : "Suresh" }
{
  "_id" : ObjectId<"5be446c3f29d9be663cbee3d">,
  "name" : "Hari",
  "age" : 23
}
{
  "_id" : ObjectId<"5be44b54f29d9be663cbee3f">,
  "name" : "Hari Prakash",
  "age" : 12,
  "city" : "Barabanki"
}
{
  "_id" : ObjectId<"5be44ce9f29d9be663cbee40">,
  "name" : "Mohit",
  "age" : 24,
  "address" : {
    "Hno" : 113,
    "Vill" : "Sangram Kheda",
    "post" : "Gulariha"
  }
}
{
  "_id" : ObjectId<"5be44ee2f29d9be663cbee41">,
  "name" : "Himanshu",
  "class" : 11,
  "Sec" : "A",
  "Subjects" : [
    "English",
    "Hindi",
    "Maths",
    "Physics",
    "Chemistry"
  ]
}
1
```

समझाने के लिए यह उदाहरण लिया है अन्यथा हमेशा बदलाव primary key के साथ किया जाना चाहिए | यहाँ जिसकी age 12 साल थी उसका नाम Hari Prakash हो गया है |

यदि सामान matching वाले एक से अधिक records में changes करने हैं तो updateMany() का प्रयोग करेंगे |

•Delete Operation:

Delete operation का प्रयोग दो तरह से किया जाता है। >deleteOne अथवा >deleteMany >db.<CollectionName>.deleteOne({<filter Exp>}) एक से ज्यादा matching होने पर भी एक record ही delete करेगा |

>db.<CollectionName>.deleteMany({<filter Exp>}) एक से ज्यादा matching होने पर एक से ज्यादा delete करेगा |

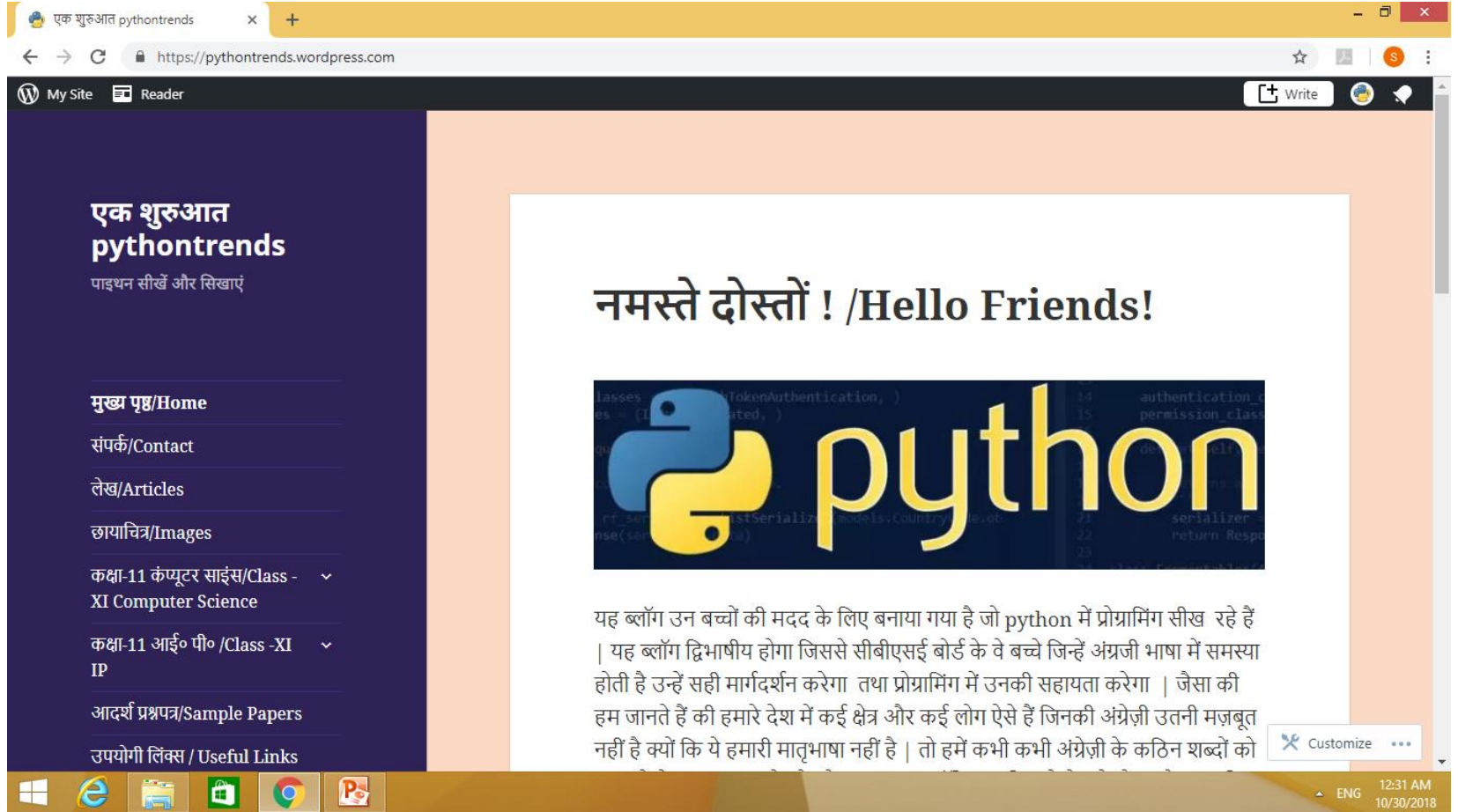
```
> db.student.deleteOne({name:'Hari Prakash'})
{
  "acknowledged" : true, "deletedCount" : 1 }
> db.student.find().pretty()
{
  "_id" : ObjectId<"5be1255c6b46969b847e3f5f">, "name" : "Pankaj" }
{
  "_id" : ObjectId<"5be446c3f29d9be663cbee3c">, "name" : "Suresh" }
{
  "_id" : ObjectId<"5be446c3f29d9be663cbee3d">,
  "name" : "Hari",
  "age" : 23
}
{
  "_id" : ObjectId<"5be44ce9f29d9be663cbee40">,
  "name" : "Mohit",
  "age" : 24,
  "address" : {
    "Hno" : 113,
    "Vill" : "Sangram Kheda",
    "post" : "Gulariha"
  }
}
{
  "_id" : ObjectId<"5be44ee2f29d9be663cbee41">,
  "name" : "Himanshu",
  "class" : 11,
  "Sec" : "A",
  "Subjects" : [
    "English",
    "Hindi",
    "Maths",
    "Physics",
    "Chemistry"
  ]
}
1
```

“Hari Prakash” का record delete हो गया

धन्यवाद

और अधिक पाठ्य-सामग्री हेतु निम्न लिंक पर क्लिक करें -

www.pythontrends.wordpress.com



एक शुरुआत pythontrends

पाठ्यन सीखें और सिखाएं

मुख्य पृष्ठ/Home

संपर्क/Contact

लेख/Articles

छायाचित्र/Images

कक्षा-11 कंप्यूटर साइंस/Class - XI Computer Science

कक्षा-11 आई० पी० /Class -XI IP

आदर्श प्रश्नपत्र/Sample Papers

उपयोगी लिंक्स / Useful Links

नमस्ते दोस्तों ! /Hello Friends!

python

यह ब्लॉग उन बच्चों की मदद के लिए बनाया गया है जो python में प्रोग्रामिंग सीख रहे हैं | यह ब्लॉग द्विभाषीय होगा जिससे सीबीएसई बोर्ड के वे बच्चे जिन्हें अंग्रेजी भाषा में समस्या होती है उन्हें सही मार्गदर्शन करेगा तथा प्रोग्रामिंग में उनकी सहायता करेगा | जैसा की हम जानते हैं की हमारे देश में कई क्षेत्र और कई लोग ऐसे हैं जिनकी अंग्रेजी उतनी मज़बूत नहीं है क्यों कि ये हमारी मातृभाषा नहीं है | तो हमें कभी कभी अंग्रेजी के कठिन शब्दों को

Customize ...

ENG 12:31 AM 10/30/2018