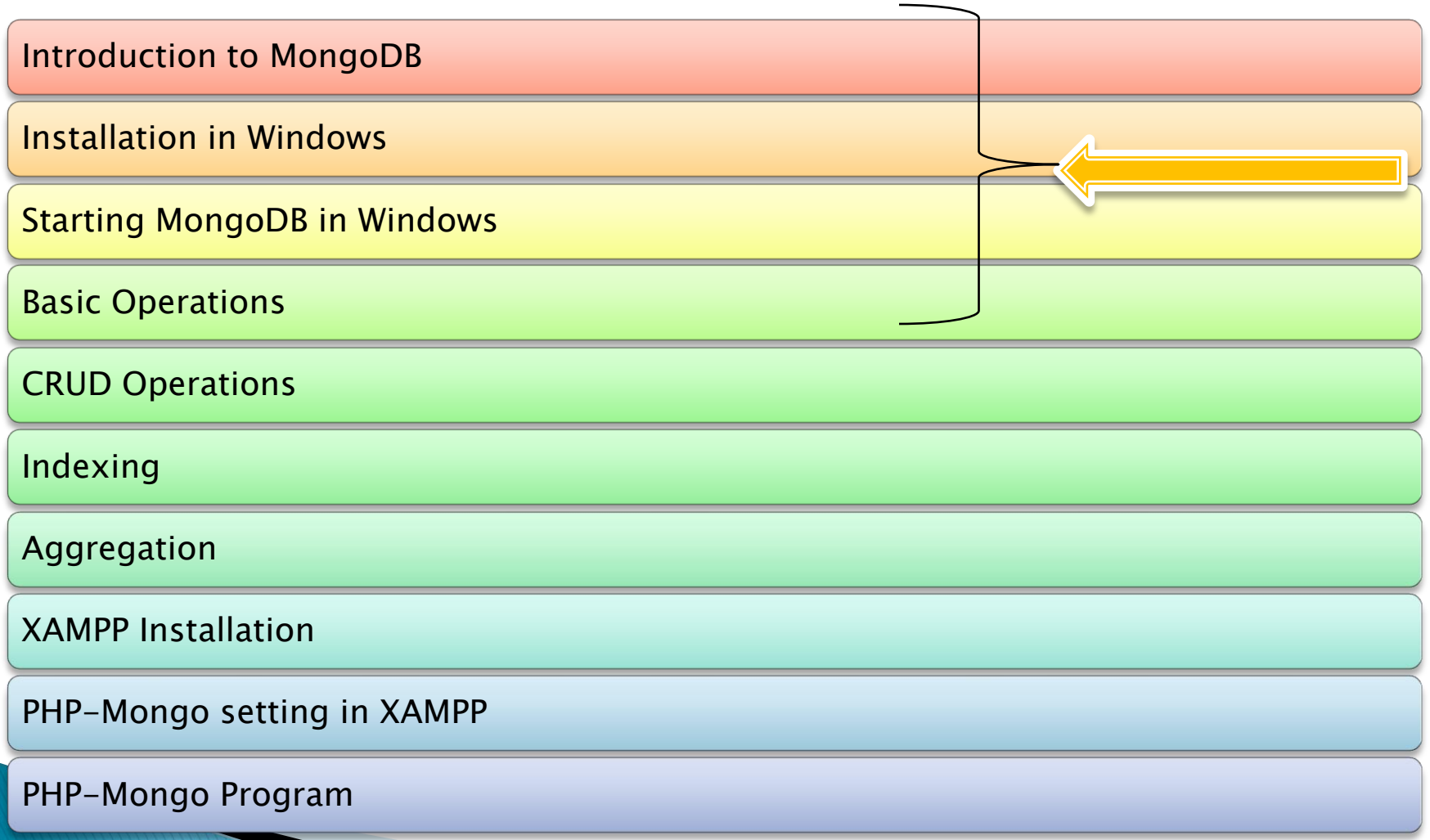
MongoDB and Php

By Prof. Bhavana A.Khivsara

Outline

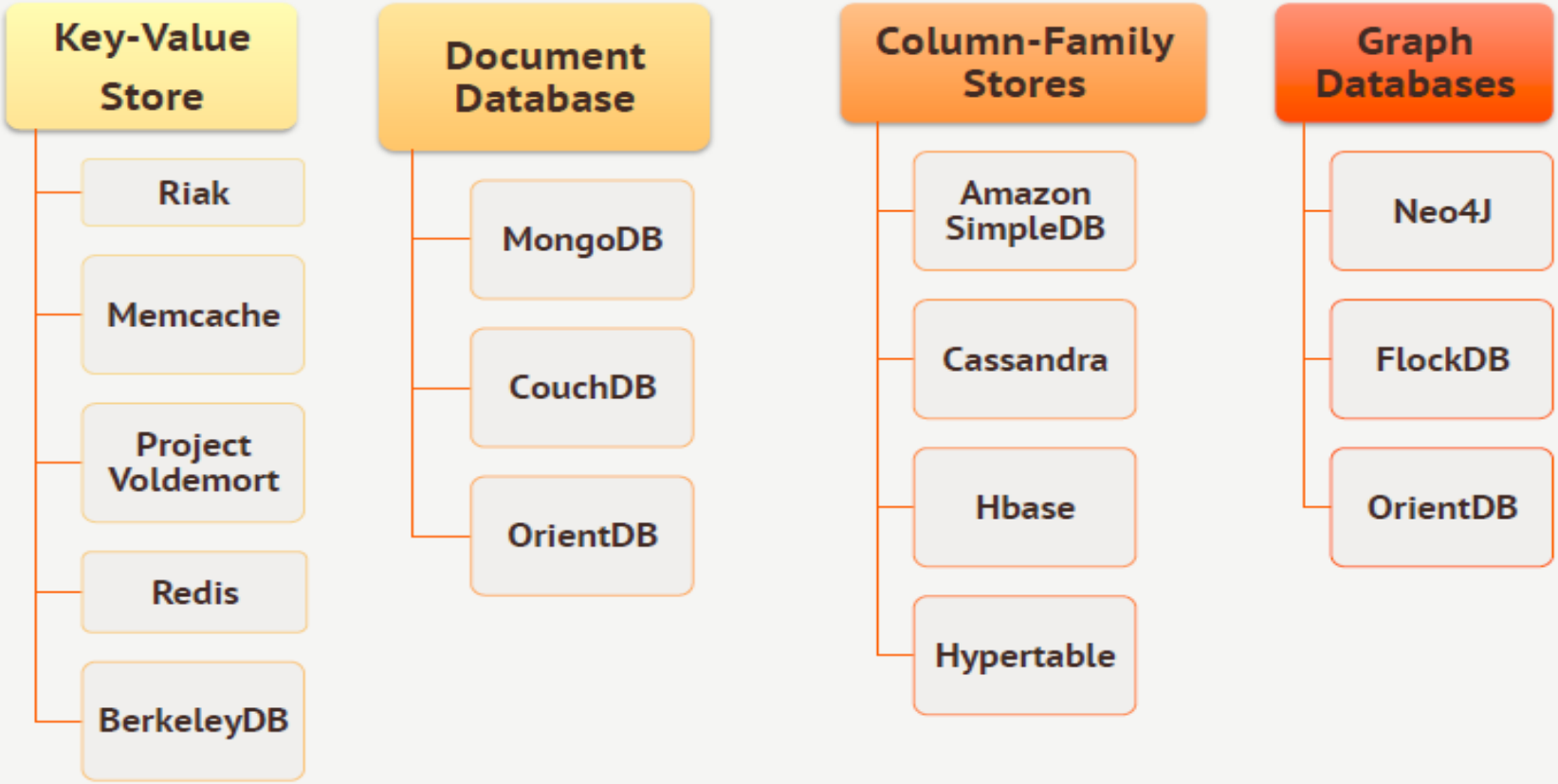

Introduction to MongoDB
Installation in Windows
Starting MongoDB in Windows
Basic Operations
CRUD Operations
Indexing
Aggregation
XAMPP Installation
PHP-Mongo setting in XAMPP
PHP-Mongo Program

Introduction of NoSQL

- Stands for **N**ot **O**nly SQL.
- Johan Oskarsson introduced the term ***NoSQL*** in early 2009
- open source distributed, non relational databases
- Handle large volumes of structured, and unstructured data. (eg.Google)
- Easy and frequent changes to DB
- Fast development

Types of NoSQL

1. NoSQL Data Model



Document store

RDBMS		MongoDB
Database	⇨	Database
Table, View	⇨	Collection
Row	⇨	Document (JSON, BSON)
Column	⇨	Field
Index	⇨	Index
Join	⇨	Embedded Document
Foreign Key	⇨	Reference
Partition	⇨	Shard

Introduction of MongoDB

- MongoDB is a _____ database
  - ✓Document
  - ✓Open source
  - ✓High performance
  - ✓High Availability
  - ✓Horizontally scalable
  - ✓Full featured

In Production

10gen | the MongoDB company

http://www.mongodb.org/about/production-deployments/

Advantages of MongoDB

- Schema less : Number of fields, content and size of the document can be differ from one document to another.
- No complex joins
- Data is stored as JSON style
- Index on any attribute
- Replication and High availability

JSON

- “JavaScript Object Notation”
- Easy for humans to write/read, easy for computers to parse/generate
- Objects can be nested
- Built on
  - name/value pairs
  - Ordered list of values

http://json.org/

BSON

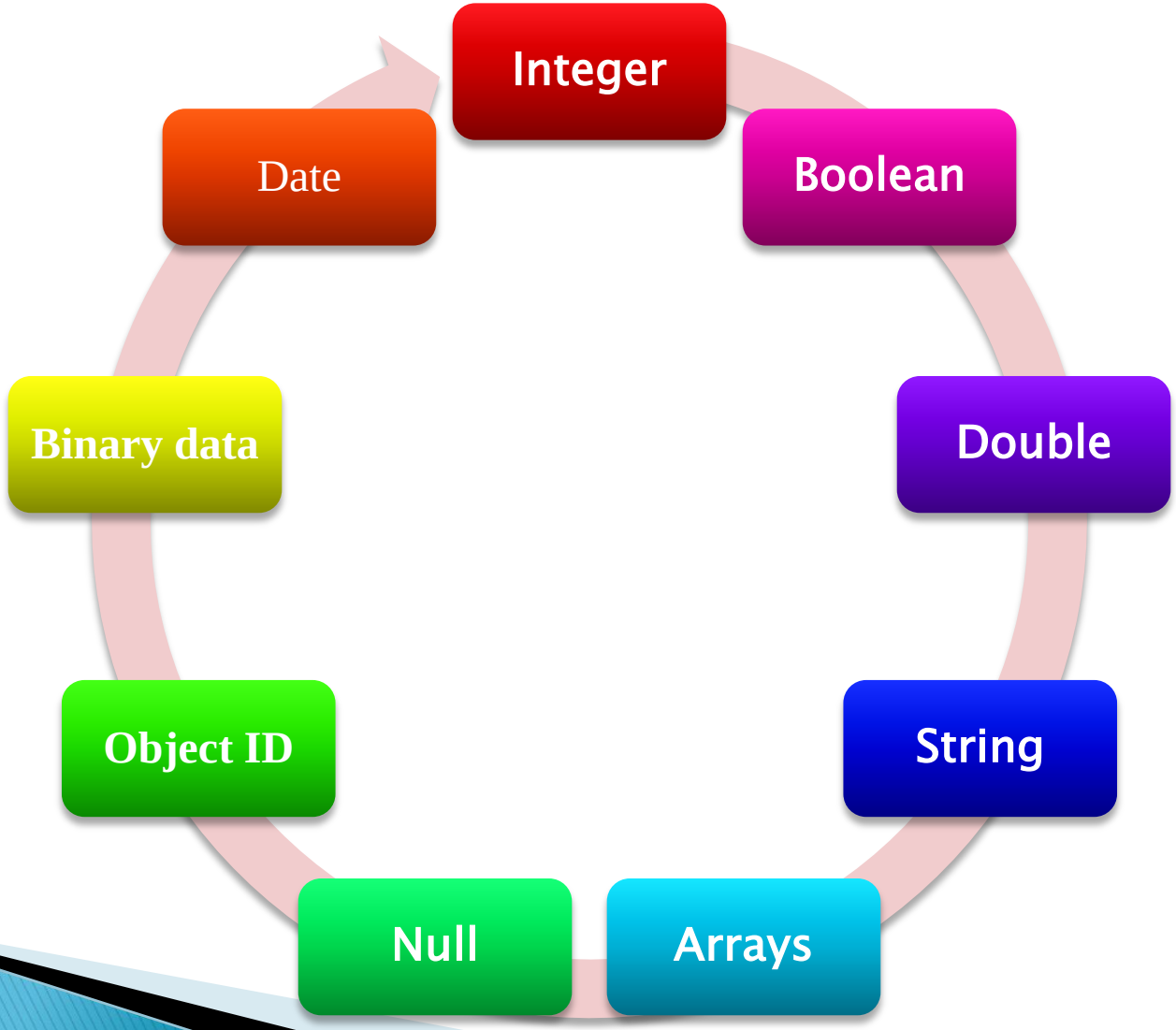
- “Binary JSON”
- Binary-encoded serialization of JSON-like docs
- Also allows “referencing”
- Embedded structure reduces need for joins
- Goals
  - Lightweight
  - Traversable
  - Efficient (decoding and encoding)

http://bsonspec.org/

BSON Example

```
{
"_id" :         "37010"
"city" :        "ADAMS",
"pop" :         2660,
"state" :       "TN",
“councilman” : {
                    name: “John Smith”
                    address: “13 Scenic Way”
                }
}
```

Data Types of MongoDB



Data Types

- **String** : This is most commonly used datatype to store the data. String in mongodb must be UTF-8 valid.
- **Integer** : This type is used to store a numerical value. Integer can be 32 bit or 64 bit depending upon your server.
- **Boolean** : This type is used to store a boolean (true/ false) value.
- **Double** : This type is used to store floating point values.
- **Min/ Max keys** : This type is used to compare a value against the lowest and highest BSON elements.
- **Arrays** : This type is used to store arrays or list or multiple values into one key.
- **Timestamp** : ctimestamp. This can be handy for recording when a document has been modified or added.
- **Object** : This datatype is used for embedded documents.

Data Types

- **Null** : This type is used to store a Null value.
- **Symbol** : This datatype is used identically to a string however, it's generally reserved for languages that use a specific symbol type.
- **Date** : This datatype is used to store the current date or time in UNIX time format. You can specify your own date time by creating object of Date and passing day, month, year into it.
- **Object ID** : This datatype is used to store the document's ID.
- **Binary data** : This datatype is used to store binay data.
- **Code** : This datatype is used to store javascript code into document.
- **Regular expression** : This datatype is used to store regular expression

Find version of Windows

enter the following commands in the *Command Prompt* or *Powershell*:

wmic os get caption

wmic os get osarchitecture

Steps for MongoDB in Windows

Create one folder eg SNJB in bin folder of MongoDB

Goto command prompt

Goto bin dir of MongoDB and write following command

mongod --storageEngine=mmapv1 --dbpath SNJB

(Server will started and listen at 27017 port)

Open another command prompt and give command mongo

(Client will be started)

Basic Database Operations

Database

collection

Basic Database Operations- Database

>use db1	• switched to db db1
>db db1	• To check your currently selected database use the command **db**
>show dbs local 0.78125GB test 0.23012GB	• If you want to check your databases list, then use the command show dbs.
db.dropDatabase()	• To Drop the database

Basic Database Operations- Collection

db.createCollection (name, options)	• >db.createCollection(“ Stud")
>show collections	• You can check the created collection by using the command **show collections**
>db.stud.insert ({"name" : “Jiya"})	• In mongodb you don't need to create collection. MongoDB creates collection automatically, when you insert some document.
>db.Stud.drop()	• MongoDB's **db.collection.drop()** is used to drop a collection from the database.

Outline

- Introduction to MongoDB
- Installation in Windows
- Starting MongoDB in Windows
- Basic Operations
- CRUD Operations
- Indexing
- Aggregation
- XAMPP Installation
- PHP-Mongo setting in XAMPP
- PHP-Mongo Program

CRUD Operations

Insert

Find

Update

Delete

CRUD Operations – Insert

- **The insert() Method**:- To insert data into MongoDB collection, you need to use MongoDB's **insert()** or **save()**method.
- **Syntax**

>db.COLLECTION_NAME.insert(document)

- **Example**

>db.mycol.insert({name: “Jiya”, age:15})

CRUD Operations – Insert

- **_id Field**
- If the document does not specify an *_id* field, then MongoDB will add the _id field and assign a unique *ObjectId* for the document before inserting.
- The _id value must be unique within the collection to avoid duplicate key error.

_Id field

_id is 12 Byte field

- 4 Bytes – Current time stamp
- 3 Bytes- Machine Id
- 2 Bytes- Process id of MongoDB Server
- 3 Bytes- Incremental Value.

CRUD Operations – Insert

- **Insert a Document without Specifying an _id Field**
- db.products.insert({ item: "card", qty: 15 })
- { "_id" : "5063114bd386d8fadbd6b004”, "item" : "card", "qty" : 15 }
- **Insert a Document Specifying an _id Field**
- db.products.insert({ _id: 10, item: "box", qty: 20 })
- { "_id" : 10, "item" : "box", "qty" : 20 }

CRUD Operations – Insert

- **Insert Multiple Documents**

```
db.products.insert
( [
 { item: "pencil", qty: 50, type: "no.2" },
 { item: "pen", qty: 20 },
 { item: "eraser", qty: 25 }
 ] )
```

CRUD Operations – Insert

- **Insert Multivalued attribute**

```
db.inventory.insert(
   {
     item: "ABC1",
     details: { model: "14Q3",
        manufacturer: "XYZ Company” },
    stock: [ { size: "S", qty: 25 }, { size: "M", qty: 50 } ],
     category: "clothing"
   }
)
```

```
db.bios.insert(
    {
      name: { first: 'John', last: 'McCarthy' },
      birth: new Date('Sep 04, 1927'),
      death: new Date('Dec 24, 2011'),
      contribs: [ 'Lisp', 'Artificial Intelligence', 'ALGOL' ],
      awards: [
                  {
                    award: 'Turing Award',
                    year: 1971,
                    by: 'ACM'
                  },
                  {
                    award: 'Kyoto Prize',
                    year: 1988,
                    by: 'Inamori Foundation'
                  },
                  {
                    award: 'National Medal of Science',
                    year: 1990,
                    by: 'National Science Foundation'
                  }
                ]
    }
)
```

CRUD Operations – Insert

db.source.copyTo(target)

Copies all documents from collection into newCollection using server-side JavaScript. If newCollection does not exist, MongoDB creates it.

CRUD Operations

Insert

Find

Update

Delete

CRUD Operations – Find

- **The find() Method**– To query data from MongoDB collection. **D**isplays all the documents in a non structured way.
- **Syntax**

>db.COLLECTION_NAME.find()

- **The pretty() Method**– To display the results in a formatted way, you can use **pretty()** method.
- **Syntax**:

>db. COLLECTION_NAME.find().pretty()

CRUD Operations – Find

Select All Documents in a Collection

- db.inventory.find({})
- OR
- db.inventory.find()

Specify Equality Condition

- use the query document { <field>: <value> }
- Examples:
- db.stud.find(name: “Jiya" })
- db.stud.find({ _id: 5 })

CRUD Operations – Find
Comparison Operators

Operator	Description
$eq	Matches values that are equal to a specified value.
$gt	Matches values that are greater than a specified value.
$gte	values that are greater than or equal to a specified value.
$lt	Matches values that are less than a specified value.
$lte	Matches values that are less than or equal to a specified value.
$ne	Matches all values that are not equal to a specified value.
$in	Matches any of the values specified in an array.
$nin	Matches none of the values specified in an array.

CRUD Operations – Find Examples

```
db.stud.find( { qty: { $gt: 25 } } )
```

```
db.stud.find( { score: { $gt: 0, $lt: 2 } } )
```

```
db.stud.find({name: “Jiya”},{age:1})
```

```
Db.stud.find({name: “jiya”},{_id:0,age:1})
```

```
db.stud.find().sort( { age: 1 } )
```

```
db.stud.find().sort( { age: -1 } )
```

CRUD Operations – Find Examples

db.stud.find().count()

db.stud.find({age:20}). count()

db.stud.find().limit(2)

db.stud.find().skip(5)

db.stud.findOne()

db.stud.find({“addr.city”: “Pune”})

db.stud.find({name: “Riya”,age:20})

CRUD Operations – Find Examples

db.stud.find({name:{$in:[“riya”,”jiya”]}})

db.stud.find({age:{$nin:[20,25]}})

db.stud.distinct(“age”)

db.stud.find({city:/^n/})

db.stud.find({city:/n/})

db.stud.find({city:/n$/})

CRUD Operations – Find Examples

db.collection.stats()

db.collection.explain().find()

db.collection.explain().find().help()

```
>var cur=db.stud.find()
>cur.forEach(printjson)
```

CRUD Operations

- Insert
- Find
- Update
- Delete

CRUD Operations – Update

- **Syntax**

```
db.collection.update(
   <query/Condition>,
   <update with $set or $unset>,
   {
     upsert: <boolean>,
     multi: <boolean>,
   }
)
```

CRUD Operations – Update

upsert

- If set to *True*, creates new document if no matches found.

multi

- If set to *True*, updates multiple documents that matches the query criteria

CRUD Operations – Update Examples

```
db.stud.update(
   { _id: 100 },
   { age: 25})

db.stud.update(
   { _id: 100 },
   { $set:{age: 25}})

db.stud.update(
   { _id: 100 },
   { $unset:{age: 1}})
```

CRUD Operations – Update Examples

```
db.stud.update(
   { _id: 100 },
   { $set: { “marks.dmsa": 50} })

db.stud.update(
   { class: “TE” },
   { $set: { “marks.dmsa": 50} } ,
   { multi: true } )

db.stud.update(
   { class: “TE” },
   { $set: { “marks.dmsa": 50} } ,
   { upsert: true } )
```

CRUD Operations – Update

Examples

```
db.stud.update(
    { },{ $inc:{age: 5}})

db.stud.update(
    { },
    { $set:{cadd: “Pune”}},
    {multi:true})

db.stud.update(
    { },
    { $rename:{“age”: “Age”}},
    {multi:true})
```

CRUD Operations

- Insert
- Find
- Update
- Delete

CRUD Operations – Remove

- **Remove All Documents**

```
db.inventory.remove({})
```

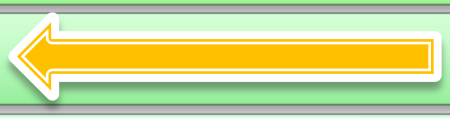
- **Remove Documents that Match a Condition**

```
db.inventory.remove( { type : "food" } )
```

- **Remove a Single Document that Matches a Condition**

```
db.inventory.remove( { type : "food" }, 1 )
```

Outline

- Introduction to MongoDB
- Installation in Windows
- Starting MongoDB in Windows
- Basic Operations
- CRUD Operations
- Indexing 
- Aggregation
- XAMPP Installation
- PHP-Mongo setting in XAMPP
- PHP-Mongo Program

Indexing

- Indexes support the efficient execution of queries in MongoDB.

Indexing Types

- *Single Field Indexes* A single field index only includes data from a single field of the documents in a collection.
- *Compound Indexes* A compound index includes more than one field of the documents in a collection.
- *Multikey Indexes* A multikey index is an index on an array field, adding an index key for each value in the array.
- *Geospatial Indexes and Queries* Geospatial indexes support location-based searches.
- *Text Indexes* Text indexes support search of string content in documents.
- *Hashed Index* Hashed indexes maintain entries with hashes of the values of the indexed field and are used with sharded clusters to support hashed shard keys.
- *Unique Indexes* A unique index causes MongoDB to reject all documents that contain a duplicate value for the indexed field

Index Creation

1 for Ascending Sorting

-1 for Descending Sorting

Using CreateIndex

- Single: db.stud.createIndex({ zipcode: 1})
- Compound: db.stud.createIndex({ dob: 1, zipcode: -1 })
- Unique: db.stud.createIndex({ rollno: 1 }, { unique: true })
- Sparse: db.stud.createIndex({ age: 1 }, { sparse: true })

Using ensureIndex

- Single: db.stud.ensureIndex({“name":1})

Index Display

- **db.collection.getIndexes()**

  Returns an array that holds a list of documents that identify and describe the existing indexes on the collection.

- **db.collection.getIndexStats()**

  Displays a human-readable summary of aggregated statistics about an index's B-tree data structure.

  db.<collection>.getIndexStats({ index : "<index name>" })

Index Drop

Syntax

- db.collection.dropIndexes()
- db.collection.dropIndex(*index*)

Example

- db.stud.dropIndexes()
- db.stud.dropIndex({ “name" : 1 })

Indexing and Querying

- create an ascending index on the field ***name*** for a collection records:

```
db.records.createIndex( { name: 1 } )
```

- This index can support an ascending sort on ***name*** :

```
db.records.find().sort( { name: 1 } )
```

- The index can also support descending sort

```
db.records.find().sort( { a: -1 } )
```

Indexing and Querying

db.stud.findOne({rno:2}), using index {rno:1}

db.stud.find ({rno:5}), using index {rno:1}

db.stud.find({rno:{$in:[2,3]}}), using index {rno:1}

db.stud.find({age:{$gt:15}}), using index {age:1}

db.stud.find({age :{$gt:2,$lt:5}}), using index {age :1}

db.stud.count({age:19}) using index {age:1}

db.stud.distinct({branch: “Computer”}) using index {branch:1}

Indexing and Querying

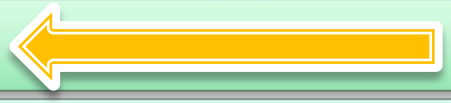
db.stud.find({}, {name:1,age:1}), using index {name:1,age:1}

db.c.find().sort({name:1,age:1}), using index {name:1,age:1}

db.stud.update({age:20}, {age:19}) using index {age:1}

db.stud.remove({name: “Jiya”}) using index {name:1}

Outline

- Introduction to MongoDB
- Installation in Windows
- Starting MongoDB in Windows
- Basic Operations
- CRUD Operations
- Indexing
- Aggregation 
- XAMPP Installation
- PHP-Mongo setting in XAMPP
- PHP-Mongo Program

Aggregation

- Aggregations operations process data records and return computed results.
- Aggregation operations group values from multiple documents together, and can perform a variety of operations on the grouped data
- For aggregation in mongodb use **aggregate()** method.
- **Syntax:**
- **>db.COLLECTION_NAME.aggregate(AGGREGATE_OPERATION)**

Aggregation

Operations that process data records and return computed results.

MongoDB provides aggregation operations

Pipelines

Provides:

- *filters* that operate like queries
- *document transformations* that modify the form of the output document.

Provides tools for:

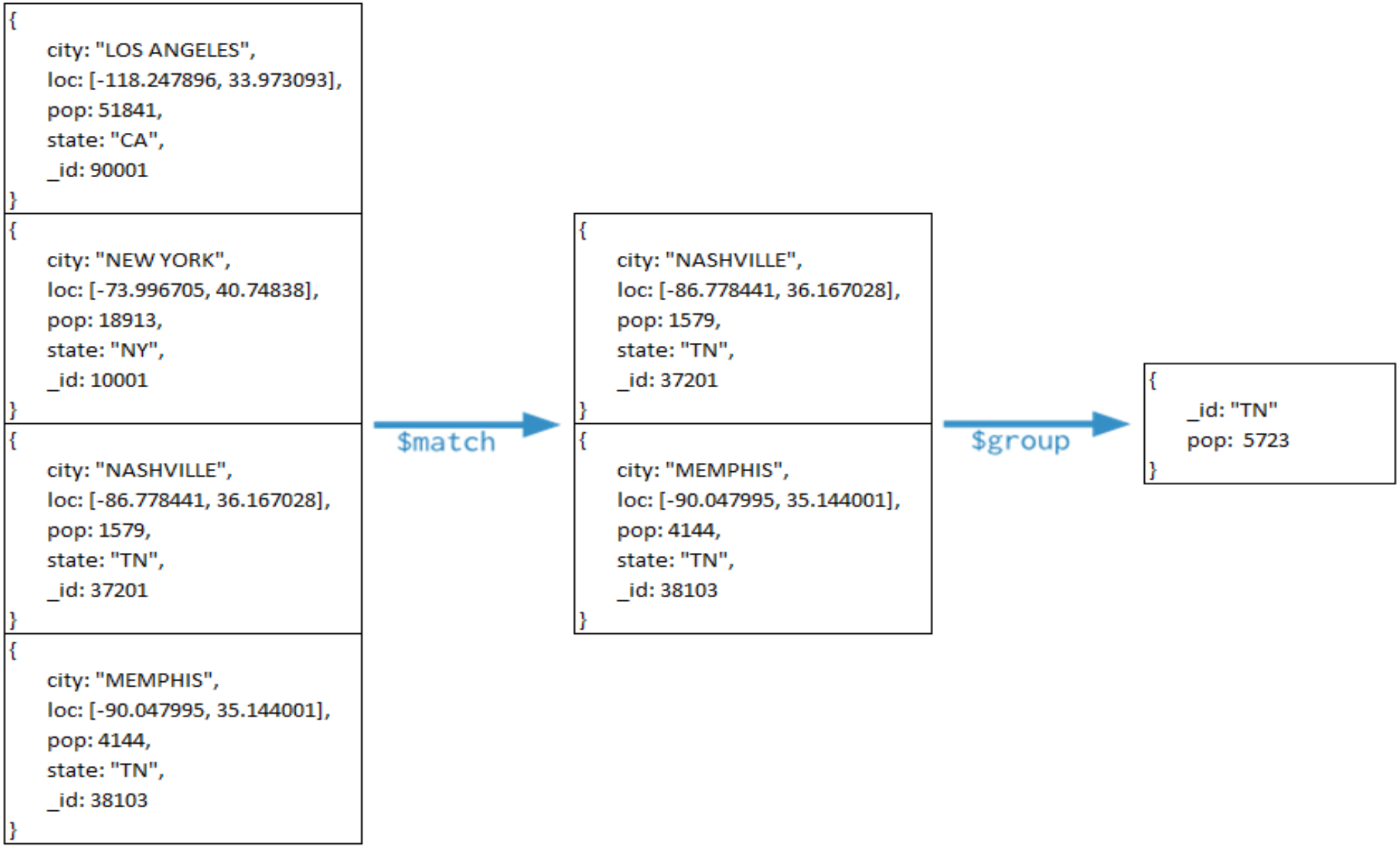
- grouping and sorting by field
- aggregating the contents of arrays, including arrays of documents

Can use operators for tasks such as calculating the average or concatenating a string.

Aggregation Framework Operators

- $project
- $match
- $limit
- $skip
- $sort
- $unwind
- $group

```
db.zips.aggregate(
                    { $match: { state: "TN" } },
                    { $group: {_id: "TN", pop: { $sum: "$pop" } } }
                  );
```



$group

- **$avg**
- **$first**
  - Returns a value from the first document for each group
- **$last**
  - Returns a value from the last document for each group
- **$max**
- **$min**
- **$push**
  - Returns an array of expression values for each group
- **$sum**

Pipeline Aggregate Examples

- db.stud.aggregate([{$match :{name:”b”}},
  {$group:{_id:null,
  total:{$sum:”$age”}}}])

- db.stud.aggregate([{$match :{name:”b”}},
  {$group:{_id:null,
  max:{$max:”$age”}}}])

- db.stud.aggregate([{$group:{_id: “$age”,
  total:{$sum:1}}}])

A Document Example

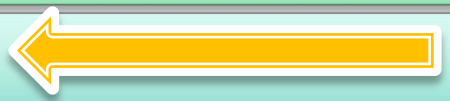
```
{
_id: ObjectId(“33667997ab01”)
  course: {code: “SWEN432”,
            title: “Advanced DB”},
  year: 2014,
  coordinator: “Pavle”,
  no_of_st: 11,
  prerequisites: [“SWEN304”, “COMP261”,
  “NWEN304"],
ratings: [1.5, 1.8, 1.3],
last_modified: ISODate("2014-02-
  13T02:14:37.948Z")
}
```

Pipeline Aggregate Examples

- Find the number of all students enrolled in `year: 2015` courses

```
db.myclasses.aggregate([
{$match: {year: 2015} },
{$group: {_id: null,
total_enrollment: {$sum: "$no_of_st"
  }}}
])
```

Outline

- Introduction to MongoDB
- Installation in Windows
- Starting MongoDB in Windows
- Basic Operations
- CRUD Operations
- Indexing
- Aggregation
- XAMPP Installation 
- PHP-Mongo setting in XAMPP
- PHP-Mongo Program

Xampp Installation on Windows

In your web browser go to
https://www.apachefriends.org/index.html

Select version for Windows 32 bit or 64 bit
and Click on the download link for XAMPP.

When prompted for the download, click
"Save" and wait for your download to finish.

click on "Run“ ->Next->Finish (select C:\)

Start the XAMPP Control Panel and Start the
Apache

Xampp Installation on Windows

Goto c:\Xampp\htdoc folder

Delete index.php file

Create new file ,write simple php code and save with **index.php** name at same location

```
<?php
echo “Welcome to PHP";
?>
```

Go to browser and type **localhost** if output of index.php is shown means xampp is ready to run any php code

Outline

- Introduction to MongoDB
- Installation in Windows
- Starting MongoDB in Windows
- Basic Operations
- CRUD Operations
- Indexing
- Aggregation
- XAMPP Installation
- PHP-Mongo setting in XAMPP
- PHP-Mongo Program

PHP-Mongo setting in XAMPP

Download php-mongo driver zip file and Extract it on your machine

Copy php_mongo-1.6.8-5.5-vc11.dll and paste it to c:\Xampp\php\ext folder

(5.5 is php version and vc11 is compiler MSCV11- you can find it from localhost-phpinfo())

Rename it with php_mongo.dll

Open php.ini (Configuration) file- under Windows Extension write *extension=php_mongo.dll*

PHP-Mongo setting in XAMPP

Then goto Xampp conrol panel stop Apache

Again start Apache

Goto browser-> type localhost-> phpinfo… you will find mongo

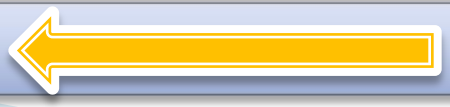
Goto command prompt start mongo server

Write a code for php mongo connectivity in index.php

Goto browser and type localhost …output of program will be shown

Outline

- Introduction to MongoDB
- Installation in Windows
- Starting MongoDB in Windows
- Basic Operations
- CRUD Operations
- Indexing
- Aggregation
- XAMPP Installation
- PHP-Mongo setting in XAMPP
- PHP-Mongo Program



PHP- Mongo connectivity Example

```
<?php
   // connect to mongodb
   $m = new MongoClient();
   echo "Connection to database successfully";
      // select a database
   $db = $m->bhavana;
   echo "Database mydb selected";
   $collection = $db->stud;
   echo "Collection selected succsessfully";
  ?>
```

Find Example

```
$cursor = $collection->find();
   // iterate cursor to display title of documents

   foreach ($cursor as $document) {
      echo $document["name"] . "\n“;
      echo $document["age"] . "\n“;
```

Insert Example

```
$doc = array(
    "name" => "MongoDB",
    "age" => 35
);

$collection->insert($doc);
```

Update Example

```
$collection->update(array("title"=>"MongoDB"),
  array('$set'=>array("title"=>"MongoDB Tutorial")));
```

Remove Example

```
$collection->remove(array("title"=>"MongoDB
  Tutorial"),false);
```

Var_Dump Example

- <?php

  $c->insert(array ("firstname" => "Bob));

  $newdata = array('$set' => array("address" => ”Smith Lane"));

  $c->update(array ("firstname" => "Bob"), $newdata);

  var_dump($c->findOne (array("firstname" => "Bob")));

  ?>

Find Examples for Array and Embedded Documents

```
<?php
   $m = new MongoClient();
   $db = $m->selectDB('test');
   $collection = new MongoCollection($db, 'produce');
   // search for Array
   $fruitQuery = array('Type' => 'Fruit');
   $cursor = $collection->find($fruitQuery);
   foreach ($cursor as $doc) {
      var_dump($doc);
   }
   // search for Embedded document
   $sweetQuery = array('Details.Taste' => 'Sweet');
   echo "Sweet\n";
   $cursor = $collection->find($sweetQuery);
   foreach ($cursor as $doc) {
      var_dump($doc);
   }
   ?>
```

Dynamic Program by taking values from user.

```
$userN = $_POST['uname'];

$userPass = $_POST[‘Password'];

 $user = $db->$collection->findOne (array

 ('username'=> $userN, 'password'=>

  $ userPass));
```

References

- https://docs.mongodb.org/
- www.tutorialspoint.com/mongodb/mongodb_php.htm
- www.wikihow.com/Install-XAMPP-for-Windows