Online Library Database Design

Part 1 – MongoDB

Zubeka Dane Dang

Project 1 information

This project is about NoSQL Document Database Design, Implementation, and Deployment Project and use MongoDB to complete.

Database application:
Select an application from the list provided below and try to create minimum 5-10 entities and 5-10 interesting relationships among the entities from the description that you will provide in the written submission. You should be familiar with the data requirements of the selected application. Designing more entities and relationship among them will show more benefits of your Document Database.

Project 1:

1. Application Description:

Literacy is a critical factor in economic and social participation, which helps in removing barriers to education and employment. About 40 per cent of libraries provide adult literacy activities or English language programs. Libraries are the primary providers of early years' literacy programs for children aged 0-3 and their families. Libraries also support digital literacy through the provision of training and free access to computers and the internet. Today's libraries are vibrant neighbourhood hubs offering social interaction and shared cultural experiences. The library is a resourceful institution that enables members to sign up to get access to books. This library store application is a collection of different books that cuts across different genres. The aim of this library store application is to provide customers with a better and improved library experience. Customers can easily find materials to read and make easy payments to either rent, purchase or download books on various devices. In this project, focus will be on 6 entities: Authors, Publishers, Members, Books, Books Borrowing and Categories with an additional 2 bridge entities which are Book/ category and Book /author.

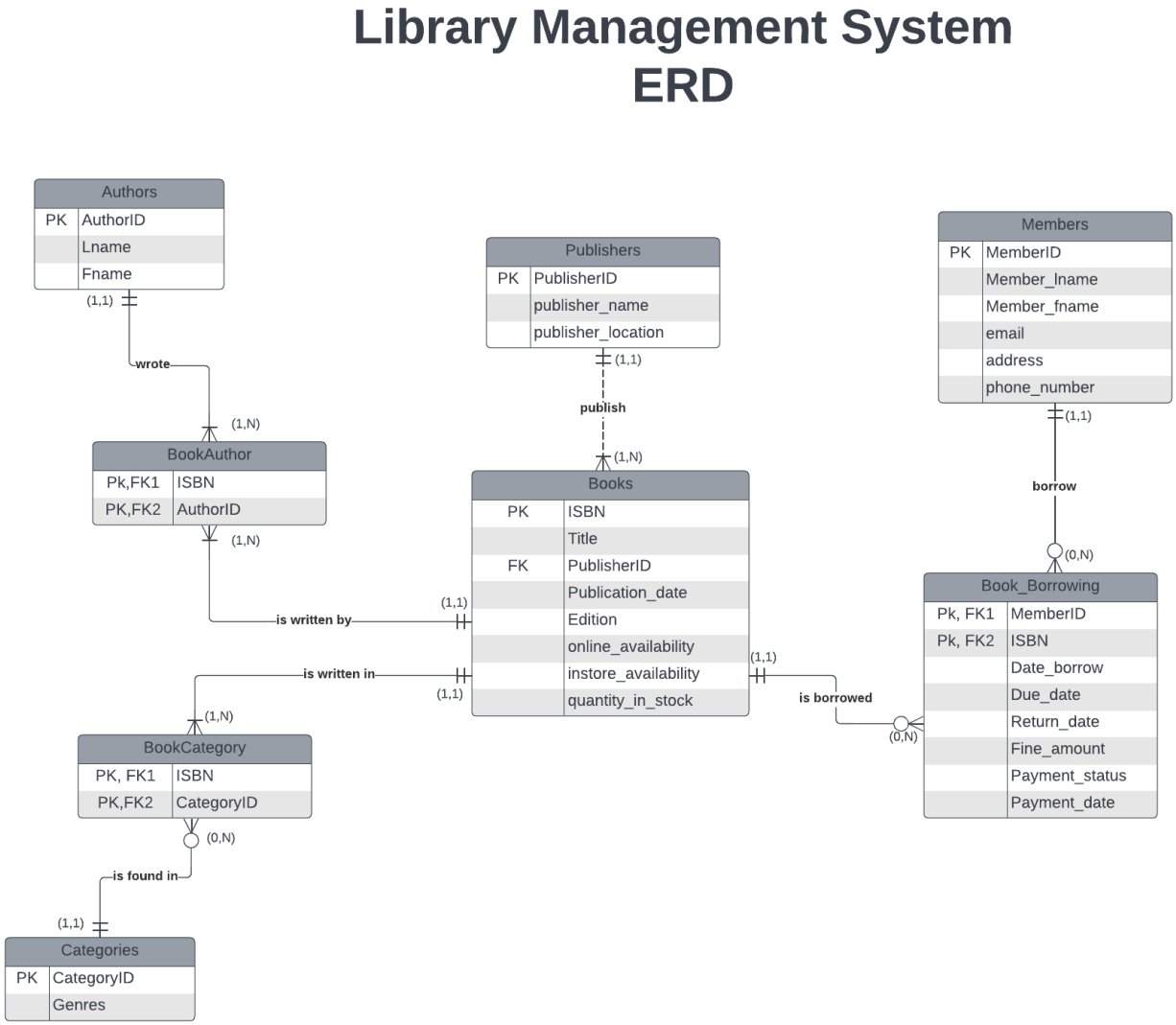
In the functionality of the library project, each member will have various attributes like member first and last name, member ID, email, address, and phone number. A member who borrows a book would be required to have a membersID under which the book borrowed will be recorded. The ISBN, date borrowed and returned are to regulate the periods when these books are to be borrowed and returned as well as if an erring member is to be fined when he/she defaults. Each book information will include the author's information, the publisher's information, if the book is available as a hardcopy or an online copy, the book's edition, and the category. The category includes the genre of books in stock like science, law etc. When a member goes to borrow a book, the application checks the memberID from the members entity. It then checks for the books ISBN number from the Books entity. It goes through all the attributes of the book entity connecting them to other entities as the need may be. The book borrowed will have information from the publisher's entity using the publisherID as the primary key, information on the books author is also gotten from the Authors entity through the bridge entity - bridge/author. The books entity also gets information from the categories entity to show what category the book belongs, and this is done through another bridge entity called the books/categories. The book entity shows if the book is available in store or online and reduces the quantity in stock whenever books are borrowed.

2. Design the ER model:

ERD model that has different type of entity relationships (1:1, 1:N, N:M) (2 marks)

Business Rules:

- A Member can borrow many books at the same time.
- Each book may have more than one copy.
- Each book may be borrowed by multiple Members.
- Each book can have one or multiple Authors.
- An Author may have more than one Book published.
- A book may belong to multiple Categories.
- Members may be fined if books are not returned on time.



3. Schema design:

Use MongoDB to create the collections derived from ER/EER diagram and submit the source code with screenshot of your collections

(1) Create Members Collection

```
> db.members.insert({
... memberID: 01,
... members_lname: 'Shahid',
... member_fname:"Mian",
... email: 'mmashahid@protonmail.com',
... phone_number: 6475625993,
... address: {
... street: '3495 morining star dr',
... city: 'Mississauga',
... province: 'Ontario',
... postal_code: "l4T 2E6" }
    ... })

    WriteResult({ "nInserted" : 1 })
```

(2) Create Categories Collection

```
> db.categories.insert({categoryID: 'cat01',
... genres: 'Fantasy'})

    WriteResult({ "nInserted" : 1 })
```

“(3) Create Authors Collection

```
> db.authors.insert({
... authorID: 'au01',
... author_lname: 'Sadalage',
... author_fname: 'Pramod'
... })
    WriteResult({ "nInserted" : 1 })
```

(4) Create Publishers Collection

```
> db.publishers.insert({
... publisherid: 'pub01',
... publisher_name: 'Course Technology',
... publisher_location: 'Boston, USA'
... })
    WriteResult({ "nInserted" : 1 })
```

(5) Create Books Collection

```
> db.books.insert({
... ISBN: 88001,
... title: "All of Our Demise",
... publisher_id: 'pub02',
... publication_date: 2022,
... edition: '1st',
... online_availability: false,
... instore_availability: true,
... quantity_in_stock: 3
... })
    WriteResult({ "nInserted" : 1 })
```

(6) Create BookArthor collection:

```
> db.bookauthor.insert({
... ISBN: 88001,
... author: [{authorID: 'au04'},{authorID: 'au05'}]})
    WriteResult({ "nInserted" : 1 })
```

(7) Create BookCategory collection:

```
> db.bookcategory.insert({
... ISBN: 88001,
... category: [{categoryID: 'ca01'}, {categoryID: 'ca04'}]})
    WriteResult({ "nInserted" : 1 })
```

(8) Create BookBorrowing collection:

```
> db.bookBorrowing.insert({
... memberID: 01,
... ISBN: 77001,
... date_borrow: new Date("2022-10-1"),
... due_date: new Date("2022-10-22"),
... return_date: new Date("2022-10-19"),
... fine_amount: 0.00,
... payment_status: 'n/a',
... Payment_date: 'n/a'})
    WriteResult({ "nInserted" : 1 })
```

4. Database construction:

Populate the collections of database by using MongoDB commands

(1) Populate the Members collection

```
> db.members.insertOne({
... memberID: 02,
... members_lname: 'Jan',
... member_fname:"Suhail",
... email: 'jansuahail@protonmail.com',
... phone_number: 6475625554,
... address: {
... street: '3126 etude dr',
... city: 'Mississauga',
... province: 'Ontario',
... postal_code: "l4T 2E7" }
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636fef4fad664e1ea22176bc")
}

> db.members.insertOne({
... memberID: 03,
... members_lname: 'Ahmad',
... member_fname:"Bilal",
... email: 'bilal@protonmail.com',
... phone_number: 6475622134,
... address: {
... street: '2024 mosque cres',
... city: 'Mississauga',
... province: 'Ontario',
... postal_code: "Mj7 2E7" }
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636fef6fad664e1ea22176bd")
}

> db.members.insertOne({
... memberID: 04,
... members_lname: 'Jahanzaib',
... member_fname:"Khan",
... email: 'jk@protonmail.com',
... phone_number: 6475622856,
... address: {
... street: '56 raxdale blvd',
... city: 'Mississauga',
```

```
... province: 'Ontario',
... postal_code: "Mj7 2k4" }
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636fef83ad664e1ea22176be")
}

> db.members.insertOne({
... memberID: 05,
... members_lname: 'Hassnain',
... member_fname:"Sajjad",
... email: 'hassnainsajjad@protonmail.com',
... phone_number: 6475622179,
... address: {
... street: '2022 derry road',
... city: 'Mississauga',
... province: 'Ontario',
... postal_code: "Mj7 2M9" }
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636fefa3ad664e1ea22176bf")
}

> db.members.insertOne({
... memberID: 06,
... members_lname: 'Dang',
... member_fname:"Luna",
... email: 'lunad@protonmail.com',
... phone_number: 3659986798,
... address: {
... street: '12 Carberry Cres',
... city: 'Brampton',
... province: 'Ontario',
... postal_code: "L6V 2E9" }
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636fefbbad664e1ea22176c0")
}
```

(2) Populate the Categories collection

```
> db.categories.insertMany([
... {categoryID: 'ca02',
... genres: 'Adventure'},
... {categoryID: 'ca03',
```

```
... genres: 'Romance'},
... {categoryID: 'ca04',
... genres: 'Horror'},
... {categoryID: 'ca05',
... genres: 'History'},
... {categoryID: 'ca06',
... genres: 'Science'},
... {categoryID: 'ca07',
... genres: 'Travel'},
... {categoryID: 'ca08',
... genres: 'Cookbook'},
... {categoryID: 'ca09',
... genres: 'Information system'},
... {categoryID: 'ca10',
... genres: 'Busisness'}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("636fefe4ad664e1ea22176c1"),
                ObjectId("636fefe4ad664e1ea22176c2"),
                ObjectId("636fefe4ad664e1ea22176c3"),
                ObjectId("636fefe4ad664e1ea22176c4"),
                ObjectId("636fefe4ad664e1ea22176c5"),
                ObjectId("636fefe4ad664e1ea22176c6"),
                ObjectId("636fefe4ad664e1ea22176c7"),
                ObjectId("636fefe4ad664e1ea22176c8"),
                ObjectId("636fefe4ad664e1ea22176c9")
        ]
}
```

(3) Populate the Publishers collection

```
> db.publishers.insertMany([
... {publisherid: 'pub02',
... publisher_name: 'Tom Doherty Association',
... publisher_location: 'Toronto, Canada'},
...
... {publisherid: 'pub03',
... publisher_name: 'Penguin Random House',
... publisher_location: 'London, UK'},
...
... {publisherid: 'pub04',
... publisher_name: 'Wesley',
... publisher_location: 'New York, USA'}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("636ff013ad664e1ea22176ca"),
```

```
            ObjectId("636ff013ad664e1ea22176cb"),
            ObjectId("636ff013ad664e1ea22176cc")
        ]
}
```

(4) Populate the Authors collection

```
> db.authors.insertMany([
... {authorID: 'au02',
... author_lname: 'Martin',
... author_fname: 'Fowler'},
...
... {authorID: 'au03',
... author_lname: 'Kenneth',
... author_fname: 'Lambert'},
...
... {authorID: 'au04',
... author_lname: 'Amanda',
... author_fname: 'Fooddy'},
...
... {authorID: 'au05',
... author_lname: 'Christine',
... author_fname: 'Lynn'},
...
... {authorID: 'au06',
... author_lname: 'Carlos',
... author_fname: 'Coronel'},
...
... {authorID: 'au07',
... author_lname: 'Steven',
... author_fname: 'Morris'},
...
... {authorID: 'au08',
... author_lname: 'Tahir',
... author_fname: 'Sabaa'}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("636ff033ad664e1ea22176cd"),
                ObjectId("636ff033ad664e1ea22176ce"),
                ObjectId("636ff033ad664e1ea22176cf"),
                ObjectId("636ff033ad664e1ea22176d0"),
                ObjectId("636ff033ad664e1ea22176d1"),
                ObjectId("636ff033ad664e1ea22176d2"),
                ObjectId("636ff033ad664e1ea22176d3")
        ]
}
```

(5) Populate the Books collection

```
> db.books.insertOne({
... ISBN: 77001,
... title: "Database Systems",
... publisher_id: 'pub01',
... publication_date: 2009,
... edition: '9th',
... online_availability: true,
... instore_availability: true,
... quantity_in_stock: 5})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ff44fad664e1ea22176f1")
}

> db.books.insertOne({
... ISBN: 77002,
... title: "Fundamentals of Python",
... publisher_id: 'pub01',
... publication_date: 2010,
... edition: '5th',
... online_availability: true,
... instore_availability: true,
... quantity_in_stock: 6})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ff466ad664e1ea22176f2")
}

> db.books.insertOne({
... ISBN: 77003,
... title: "Statistics for Management and Economics",
... publisher_id: 'pub01',
... publication_date: 2018,
... edition: '11th',
... online_availability: true,
... instore_availability: false,
... quantity_in_stock: 0
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ff47dad664e1ea22176f3")
}


> db.books.insertOne({
... ISBN: 88002,
```

```
... title: "All My Rage A Novel",
... publisher_id: 'pub03',
... publication_date: 2022,
... edition: '1st',
... online_availability: false,
... instore_availability: true,
... quantity_in_stock: 4})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ff490ad664e1ea22176f4")
}

> db.books.insertOne({
... ISBN: 77004,
... title: "NoSQL Distilled",
... publisher_id: 'pub04',
... publication_date: 2013,
... edition: '3rd',
... online_availability: true,
... instore_availability: true,
... quantity_in_stock: 6})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ff4a2ad664e1ea22176f5")
}
```

(6) Populate the BookAuthor collection

```
> db.bookauthor.insert({
... ISBN: 88001,
... author: [{authorID: 'au04'},{authorID: 'au05'}]})
WriteResult({ "nInserted" : 1 })
> db.bookcategory.insert({
... ISBN: 88001,
... category: [{categoryID: 'ca01'}, {categoryID: 'ca04'}]})
WriteResult({ "nInserted" : 1 })
> db.bookauthor.insertMany([
... {ISBN: 88002,
... author: {authorID: 'au08'}},
... {ISBN: 77001,
... author: [{authorID: 'au02'}, {authorID: 'au06'},{authorID: 'au07'}]},
... {ISBN: 77002,
... author: {authorID: 'au03'}},
... {ISBN: 77003,
... author: {authorID: 'au03'}},
... {ISBN: 77004,
... author: {authorID: 'au01'}}])
{
```

```
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("63706f71ad664e1ea2217710"),
                ObjectId("63706f71ad664e1ea2217711"),
                ObjectId("63706f71ad664e1ea2217712"),
                ObjectId("63706f71ad664e1ea2217713"),
                ObjectId("63706f71ad664e1ea2217714")
        ]
}
```

(7) Populate the BookCategory collection

```
> db.bookcategory.insertMany([
... {ISBN: 88002,
... category: [{categoryID: 'ca02'}, {categoryID: 'ca05'}]},
...
... {ISBN: 77001,
... category: {categoryID: 'ca09'}},
...
... {ISBN: 77002,
... category: [{categoryID: 'ca06'}, {categoryID: 'ca09'}]},
...
... {ISBN: 77003,
... category: [{categoryID: 'ca09'},{categoryID: 'ca10'}]},
...
... {ISBN: 77004,
... category:[{categoryID: 'ca09'},{categoryID: 'ca06'}]}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("63706f8bad664e1ea2217715"),
                ObjectId("63706f8bad664e1ea2217716"),
                ObjectId("63706f8bad664e1ea2217717"),
                ObjectId("63706f8bad664e1ea2217718"),
                ObjectId("63706f8bad664e1ea2217719")
        ]
}
```

(8) Populate the BookBorrowing collection

```
> db.bookBorrowing.insertOne({
... memberID: 02,
... ISBN: 88001,
... date_borrow: new Date("2022-10-12"),
... due_date: new Date("2022-11-1"),
... return_date: 'n/a',
... fine_amount: 1.00,
... payment_status: 'unpaid',
... Payment_date: 'n/a'})
{
```

```
        "acknowledged" : true,
        "insertedId" : ObjectId("636ffd68ad664e1ea2217703")
}

> db.bookBorrowing.insertOne({
... memberID: 03,
... ISBN: 88002,
... date_borrow: new Date("2022-9-10"),
... due_date: new Date("2022-9-30"),
... return_date: new Date("2022-10-15"),
... fine_amount: 2.00,
... payment_status: 'paid',
... Payment_date: new Date("2022-10-15")})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ffdb0ad664e1ea2217704")
}
> db.bookBorrowing.insertOne({
... memberID: 03,
... ISBN: 77004,
... date_borrow: new Date("2022-9-10"),
... due_date: new Date("2022-9-30"),
... return_date: new Date("2022-10-15"),
... fine_amount: 2.00,
... payment_status: 'paid',
... Payment_date: new Date("2022-10-15")
...
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ffdc3ad664e1ea2217705")
}

> db.bookBorrowing.insertOne({
... memberID: 04,
... ISBN: 88002,
... date_borrow: new Date("2022-10-10"),
... due_date: new Date("2022-10-31"),
... return_date: 'n/a',
... fine_amount: 3.00,
... payment_status: 'unpaid',
... Payment_date: 'n/a'})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ffdddad664e1ea2217706")
}
```

```
> db.bookBorrowing.insertOne({
... memberID: 05,
... ISBN: 77002,
... date_borrow: new Date("2022-10-29"),
... due_date: new Date("2022-11-12"),
... return_date: 'n/a',
... fine_amount: 0.00,
... payment_status: 'n/a',
... Payment_date: 'n/a'
...
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("636ffdefad664e1ea2217707")
}
```

5. Designing Simple Quarries:

Prepare at least 8 simple queries (at least one for each of your collection) and show the snapshots of their results. In general, you have to produce professional report results for queries. The query results should show records in collections with meaningful titles and proper format.

(1) Show all the collections:

```
> show collections

authors
bookBorrowing
bookauthor
bookcactegory
books
categories
members
publishers
```

(2) Retrieve all information on the users that live in 'Brampton'

```
> db.members.find({"address.city": "Brampton"}).pretty()

{
        "_id" : ObjectId("636fefbbad664e1ea22176c0"),
        "memberID" : 6,
        "members_lname" : "Dang",
        "member_fname" : "Luna",
        "email" : "lunad@protonmail.com",
        "phone_number" : 3659986798,
        "address" : {
```

```
            "street" : "12 Carberry Cres",
            "city" : "Brampton",
            "province" : "Ontario",
            "postal_code" : "L6V 2E9"
        }
}
```

(3) Retrieve all information on all of the books published after 2018

```
> db.books.find({"publication_date": {"$gt": 2018}}).pretty()

{
        "_id" : ObjectId("636ff415ad664e1ea22176f0"),
        "ISBN" : 88001,
        "title" : "All of Our Demise",
        "publisher_id" : "pub02",
        "publication_date" : 2022,
        "edition" : "1st",
        "online_availability" : false,
        "instore_availability" : true,
        "quantity_in_stock" : 3
}
{
        "_id" : ObjectId("636ff490ad664e1ea22176f4"),
        "ISBN" : 88002,
        "title" : "All My Rage A Novel",
        "publisher_id" : "pub03",
        "publication_date" : 2022,
        "edition" : "1st",
        "online_availability" : false,
        "instore_availability" : true,
        "quantity_in_stock" : 4
}
```

(4) Find all members who borrow the book "All My Rage A Novel", show their memberID, firstname and last name.

```
> db.bookBorrowing.aggregate([{$lookup: {from: 'members', localField: 'memberID',
foreignField: 'memberID', as: 'Member'}},
. ..... {$lookup: {from: 'books', localField: 'ISBN', foreignField: 'ISBN', as: 'Book'}}, {$match:
{"Book.title" : "All My Rage A Novel"}},
... {$project:{"_id": 0, "Member.memberID":1, "Member.member_fname":1,
"Member.members_lname":1, "Book.title":1, "ISBN":1}}]).pretty()
{
        "ISBN" : 88002,
        "Member" : [
                {
                        "memberID" : 3,
                        "members_lname" : "Ahmad",
                        "member_fname" : "Bilal"
```

```
                }
            ],
            "Book" : [
                {
                    "title" : "All My Rage A Novel"
                }
            ]
        }
        {
            "ISBN" : 88002,
            "Member" : [
                {
                    "memberID" : 4,
                    "members_lname" : "Jahanzaib",
                    "member_fname" : "Khan"
                }
            ],
            "Book" : [
                {
                    "title" : "All My Rage A Novel"
                }
            ]
        }
```

(5) Concatenates Authors' first name and last name, but do not show _id:

```
> db.authors.aggregate({$project:{_id: 0, authorName: {$concat: ["$author_fname"," ",
"$author_lname"]}}})

{ "authorName" : "Pramod Sadalage" }
{ "authorName" : "Fowler Martin" }
{ "authorName" : "Lambert Kenneth" }
{ "authorName" : "Fooddy Amanda" }
{ "authorName" : "Lynn Christine" }
{ "authorName" : "Coronel Carlos" }
{ "authorName" : "Morris Steven" }
{ "authorName" : "Sabaa Tahir" }
```

(6) Retrieve all information on the publishers that are not in 'Toronto, Canada'

```
> db.publishers.find({"publisher_location": {$ne: 'Toronto, Canada'}}).pretty()

{
    "_id" : ObjectId("636febf9ad664e1ea22176b7"),
    "publisherid" : "pub01",
    "publisher_name" : "Course Technology",
    "publisher_location" : "Boston, USA"
}
{
    "_id" : ObjectId("636ff013ad664e1ea22176cb"),
```

```
        "publisherid" : "pub03",
        "publisher_name" : "Penguin Random House",
        "publisher_location" : "London, UK"
}
{
        "_id" : ObjectId("636ff013ad664e1ea22176cc"),
        "publisherid" : "pub04",
        "publisher_name" : "Wesley",
        "publisher_location" : "New York, USA"
}
```

(7) Count the number of books written by the author with authorID as 'au03'

```
> db.bookauthor.aggregate([{$lookup: {from: 'authors', 'localField': 'author.authorID',
foreignField: 'authorID', as: "Author"}},{$match: {'Author.authorID': "au03"}}, {$count:
"ISBN"}]).pretty()

{ "ISBN" : 2 }
```

(8) Find the total fine amount that members who have borrowed books have had to pay.

```
 > db.bookBorrowing.aggregate([{$group: {_id: '$memberID', totalFine:{$sum: "$fine_amount"}}}])

 { "_id" : 2, "totalFine" : 1 }
 { "_id" : 4, "totalFine" : 3 }
 { "_id" : 3, "totalFine" : 4 }
 { "_id" : 5, "totalFine" : 0 }
{ "_id" : 1, "totalFine" : 0 }
```

(9) Find title and genres of books

```
 > db.bookcategory.aggregate([{$lookup: {from: 'categories', localField: 'category.categoryID',
 foreignField: 'categoryID', as: 'Category'}},
 ..... {$lookup: {from: 'books', localField: 'ISBN', foreignField: 'ISBN', as: 'Book'}},
 ... {$project:{"_id": 0,"ISBN":1, "Category.categoryID":1, "Category.genres":1,
 "Book.title":1}}]).pretty()
 {
        "ISBN" : 88001,
        "Category" : [
                {
                        "categoryID" : "ca04",
                        "genres" : "Horror"
                }
        ],
        "Book" : [
                {
                        "title" : "All of Our Demise"
                }
        ]
 }
 {
```

```
        "ISBN" : 88002,
        "Category" : [
            {
                "categoryID" : "ca02",
                "genres" : "Adventure"
            },
            {
                "categoryID" : "ca05",
                "genres" : "History"
            }
        ],
        "Book" : [
            {
                "title" : "All My Rage A Novel"
            }
        ]
    }
    {
        "ISBN" : 77001,
        "Category" : [
            {
                "categoryID" : "ca09",
                "genres" : "Information system"
            }
        ],
        "Book" : [
            {
                "title" : "Database Systems"
            }
        ]
    }
    {
        "ISBN" : 77002,
        "Category" : [
            {
                "categoryID" : "ca06",
                "genres" : "Science"
            },
            {
                "categoryID" : "ca09",
                "genres" : "Information system"
            }
        ],
        "Book" : [
            {
                "title" : "Fundamentals of Python"
            }
        ]
    }
    {
        "ISBN" : 77003,
        "Category" : [
            {
                "categoryID" : "ca09",
```

```
                "genres" : "Information system"
            },
            {
                "categoryID" : "ca10",
                "genres" : "Busisness"
            }
        ],
        "Book" : [
            {
                "title" : "Statistics for Management and Economics"
            }
        ]
 }
 {
        "ISBN" : 77004,
        "Category" : [
            {
                "categoryID" : "ca06",
                "genres" : "Science"
            },
            {
                "categoryID" : "ca09",
                "genres" : "Information system"
            }
        ],
        "Book" : [
            {
                "title" : "NoSQL Distilled"
            }
        ]
}
```


6. Designing Advanced Quarries:

Prepare at least 6 advanced queries to retrieve data from two or more collections. All queries should have clear and nice formatted results.

(1) Retrieve all information on the books that have borrowing due date is today.

```
> db.bookBorrowing.aggregate([{$lookup: {from: 'books', localField: 'ISBN',
  foreignField: 'ISBN', as: 'Book'}}, {$match: {"due_date":{$gte:ISODate("2022-11-
  12")}}}]) .pretty()
```


```
{
    "_id" : ObjectId("636ffdefad664e1ea2217707"),
    "memberID" : 5,
    "ISBN" : 77002,
    "date_borrow" : ISODate("2022-10-29T00:00:00Z"),
    "due_date" : ISODate("2022-11-12T00:00:00Z"),
    "return_date" : "n/a",
    "fine_amount" : 0,
    "payment_status" : "n/a",
```

```
        "Payment_date" : "n/a",
        "Book" : [
                {
                        "_id" : ObjectId("636ff466ad664e1ea22176f2"),
                        "ISBN" : 77002,
                        "title" : "Fundamentals of Python",
                        "publisher_id" : "pub01",
                        "publication_date" : 2010,
                        "edition" : "5th",
                        "online_availability" : true,
                        "instore_availability" : true,
                        "quantity_in_stock" : 6
                }
        ]
}
```


(2) List the book were borrowed between Oct 5, 2022 and Oct 15, 2022

```
> db.bookBorrowing.aggregate([{$lookup:{from: "books", localField: "ISBN",
  foreignField:"ISBN", as:"bookBorrowing"}}, {$match: {date_borrow:{$gte: new
  Date("2022-10-5"), $lt: new Date("2022-10-15")}}}]).pretty()
```


```
{
        "_id" : ObjectId("636ffd68ad664e1ea2217703"),
        "memberID" : 2,
        "ISBN" : 88001,
        "date_borrow" : ISODate("2022-10-12T00:00:00Z"),
        "due_date" : ISODate("2022-11-01T00:00:00Z"),
        "return_date" : "n/a",
        "fine_amount" : 1,
        "payment_status" : "unpaid",
        "Payment_date" : "n/a",
        "bookBorrowing" : [
                {
                        "_id" : ObjectId("636ff415ad664e1ea22176f0"),
                        "ISBN" : 88001,
                        "title" : "All of Our Demise",
                        "publisher_id" : "pub02",
                        "publication_date" : 2022,
                        "edition" : "1st",
                        "online_availability" : false,
                        "instore_availability" : true,
                        "quantity_in_stock" : 3
                }
        ]
}
{
        "_id" : ObjectId("636ffdddad664e1ea2217706"),
        "memberID" : 4,
```

```
        "ISBN" : 88002,
        "date_borrow" : ISODate("2022-10-10T00:00:00Z"),
        "due_date" : ISODate("2022-10-31T00:00:00Z"),
        "return_date" : "n/a",
        "fine_amount" : 3,
        "payment_status" : "unpaid",
        "Payment_date" : "n/a",
        "bookBorrowing" : [
                {
                        "_id" : ObjectId("636ff490ad664e1ea22176f4"),
                        "ISBN" : 88002,
                        "title" : "All My Rage A Novel",
                        "publisher_id" : "pub03",
                        "publication_date" : 2022,
                        "edition" : "1st",
                        "online_availability" : false,
                        "instore_availability" : true,
                        "quantity_in_stock" : 4
                }
        ]
}
```

(3) Find the allowed borrowing period of the books that have been borrowed.

```
> db.bookBorrowing.aggregate([{$lookup: {from: 'books', localField: 'ISBN', foreignField: 'ISBN',
as: 'Book'}}, {$project: {"_id": 0, "Book.title":1,"date_borrow": 1, "due_date": 1,
"borrowingPeriod": {$divide: [{$subtract: [ "$due_date", "$date_borrow" ]},
1000*60*60*24]}}}]).pretty()
{
        "date_borrow" : ISODate("2022-10-01T00:00:00Z"),
        "due_date" : ISODate("2022-10-22T00:00:00Z"),
        "Book" : [
                {
                        "title" : "Database Systems"
                }
        ],
        "borrowingPeriod" : 21
}
{
        "date_borrow" : ISODate("2022-10-12T00:00:00Z"),
        "due_date" : ISODate("2022-11-01T00:00:00Z"),
        "Book" : [
                {
                        "title" : "All of Our Demise"
                }
        ],
        "borrowingPeriod" : 20
}
{
```

```
        "date_borrow" : ISODate("2022-09-10T00:00:00Z"),
        "due_date" : ISODate("2022-09-30T00:00:00Z"),
        "Book" : [
                {
                        "title" : "All My Rage A Novel"
                }
        ],
        "borrowingPeriod" : 20
}
{
        "date_borrow" : ISODate("2022-09-10T00:00:00Z"),
        "due_date" : ISODate("2022-09-30T00:00:00Z"),
        "Book" : [
                {
                        "title" : "NoSQL Distilled"
                }
        ],
        "borrowingPeriod" : 20
}
{
        "date_borrow" : ISODate("2022-10-10T00:00:00Z"),
        "due_date" : ISODate("2022-10-31T00:00:00Z"),
        "Book" : [
                {
                        "title" : "All My Rage A Novel"
                }
        ],
        "borrowingPeriod" : 21
}
{
        "date_borrow" : ISODate("2022-10-29T00:00:00Z"),
        "due_date" : ISODate("2022-11-12T00:00:00Z"),
        "Book" : [
                {
                        "title" : "Fundamentals of Python"
                }
        ],
        "borrowingPeriod" : 14
}
```

(4) Find the books with their title that have not been return yet

```
> db.bookBorrowing.aggregate([{$lookup: {from: 'books', localField: 'ISBN', foreignField: 'ISBN', as: 'Book'}}, {$project: {"_id": 0, "Book.title": 1, "return_date": 1}},  {$match: {"return_date": 'n/a'}}]).pretty()

{
        "return_date" : "n/a",
        "Book" : [
                {
                        "title" : "All of Our Demise"
                }
        ]
```

```
 }
 {
        "return_date" : "n/a",
        "Book" : [
                {
                        "title" : "All My Rage A Novel"
                }
        ]
 }
 {
        "return_date" : "n/a",
        "Book" : [
                {
                        "title" : "Fundamentals of Python"
                }
        ]
}
```


(5) Count the number of books that have category as 'Information system'

```
> db.bookcategory.aggregate([{$lookup: {from: 'books', localField: 'ISBN',
  foreignField: 'ISBN', as: 'Book'}},{$lookup:{from: 'categories', localField:
  'category.categoryID', foreignField: 'categoryID', as: 'Category'}}, {$match:
  {'Category.genres': 'Information system'}}, {$count: 'ISBN'}]).pretty()

  { "ISBN" : 4 }
```

(6) Retrieve all information on the books that have title contain "python"

```
> db.books.find({title: {$regex: /ython/, $options: 'p'}}).pretty()

{
        "_id" : ObjectId("636ff466ad664e1ea22176f2"),
        "ISBN" : 77002,
        "title" : "Fundamentals of Python",
        "publisher_id" : "pub01",
        "publication_date" : 2010,
        "edition" : "5th",
        "online_availability" : true,
        "instore_availability" : true,
        "quantity_in_stock" : 6
}
```

Find the Book ID, title and due date of all the book check out by member with last name "Bilal"

What is the total fine amount that member named "Luna" has to pay.
List the names, surnames and the names of the authors who wrote "Drama" type.
List the name and surname of the members and count the number of books they borrowed.
What is the oldest book in the library.

List all member name, lastname, the title of the taken book, the taken date, the book's categories and the name and surname of the author

7. Updating and deleting database content:

Prepare at least 4 commands for updating different collections and the same for the other 4 deleting database content commands. (2 marks)

(1) Update location of publisher "Course Technology" as "Florida, USA"

```
> db.publishers.update({"publisher_name":'Course Technology'}, {$set:
{'publisher_location': 'Florida, USA'}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

> db.publishers.find({"publisher_name":'Course Technology'})

{ "_id" : ObjectId("636febf9ad664e1ea22176b7"), "publisherid" : "pub01", "publisher_name"
: "Course Technology", "publisher_location" : "Florida, USA" }
```

(2) Update new email of member "Luna Dang" as "lunad@email.ca"

```
> db.members.update({"member_fname":"Luna","members_lname":"Dang"},   {$set:
{"email": "lunadang@cmail.ca"}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

> db.members.find({email:  "lunadang@cmail.ca"}).pretty()

{
        "_id" : ObjectId("636fefbbad664e1ea22176c0"),
        "memberID" : 6,
        "members_lname" : "Dang",
        "member_fname" : "Luna",
        "email" : "lunadang@cmail.ca",
        "phone_number" : 3659986798,
        "address" : {
                "street" : "12 Carberry Cres",
                "city" : "Brampton",
                "province" : "Ontario",
                "postal_code" : "L6V 2E9"
        }
}
```

(3) Update Fine amount for books that have due date is today but have not been returned yet.

```
    > db.bookBorrowing.update({"due_date": new Date("2022-11-
        12")},{$set:{"fine_amount": 1.00, "payment_status": "unpaid"}})
    WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 })

    > db.bookBorrowing.find({"due_date": new Date("2022-11-12")}).pretty()
```

```
{
        "_id" : ObjectId("636ffdefad664e1ea2217707"),
        "memberID" : 5,
        "ISBN" : 77002,
        "date_borrow" : ISODate("2022-10-29T00:00:00Z"),
        "due_date" : ISODate("2022-11-12T00:00:00Z"),
        "return_date" : "n/a",
        "fine_amount" : 1,
        "payment_status" : "unpaid",
        "Payment_date" : "n/a"
    }
```

(4) Update the payment status as Paid, Payment_date and return_date as today for the member with memberID: 04 cand the book ISBN as 88002

```
> db.bookBorrowing.update({memberID: 04, ISBN: 88002}, {$set: {return_date: new
    Date(), payment_status: "paid", Payment_date: new Date()}})

    WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

> db.bookBorrowing.find({memberID: 04, ISBN: 88002}).pretty()
{
        "_id" : ObjectId("636ffdddad664e1ea2217706"),
        "memberID" : 4,
        "ISBN" : 88002,
        "date_borrow" : ISODate("2022-10-10T00:00:00Z"),
        "due_date" : ISODate("2022-10-31T00:00:00Z"),
        "return_date" : ISODate("2022-11-13T05:02:10.437Z"),
        "fine_amount" : 3,
        "payment_status" : "paid",
        "Payment_date" : ISODate("2022-11-13T05:02:10.437Z")
    }
```

(5) Delete quantity_in_stock field of the books that are not available in store.

```
> db.books.update({instore_availability: false}, {$unset: {quantity_in_stock: ""}})
    WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

> db.books.find({instore_availability: false}).pretty()
{
        "_id" : ObjectId("636ff47dad664e1ea22176f3"),
        "ISBN" : 77003,
        "title" : "Statistics for Management and Economics",
        "publisher_id" : "pub01",
        "publication_date" : 2018,
        "edition" : "11th",
        "online_availability" : true,
        "instore_availability" : false
    }
```

(6) Delete the genres “Fantasy” in Categories collection

```
> db.categories.deleteOne({genres: "Fantasy"})
    { "acknowledged" : true, "deletedCount" : 1 }
```

(7) Delete the book borrowing record where fine was paid

```
> db.bookBorrowing.deleteMany({payment_status: "paid"})
{ "acknowledged" : true, "deletedCount" : 3 }
> db.bookBorrowing.find().pretty()
{
        "_id" : ObjectId("636ffc91ad664e1ea2217702"),
        "memberID" : 1,
        "ISBN" : 77001,
        "date_borrow" : ISODate("2022-10-01T00:00:00Z"),
        "due_date" : ISODate("2022-10-22T00:00:00Z"),
        "return_date" : ISODate("2022-10-19T00:00:00Z"),
        "fine_amount" : 0,
        "payment_status" : "n/a",
        "Payment_date" : "n/a"
}
{
        "_id" : ObjectId("636ffd68ad664e1ea2217703"),
        "memberID" : 2,
        "ISBN" : 88001,
        "date_borrow" : ISODate("2022-10-12T00:00:00Z"),
        "due_date" : ISODate("2022-11-01T00:00:00Z"),
        "return_date" : "n/a",
        "fine_amount" : 1,
        "payment_status" : "unpaid",
        "Payment_date" : "n/a"
}
{
        "_id" : ObjectId("636ffdefad664e1ea2217707"),
        "memberID" : 5,
        "ISBN" : 77002,
        "date_borrow" : ISODate("2022-10-29T00:00:00Z"),
        "due_date" : ISODate("2022-11-12T00:00:00Z"),
        "return_date" : "n/a",
        "fine_amount" : 1,
        "payment_status" : "unpaid",
        "Payment_date" : "n/a"
}
    >
```

(8) Find member who has not borrowed books and delete them

```
> db.members.aggregate([{$lookup:{from: "bookBorrowing", localField: "memberID",
    foreignField:"memberID", as: "Borrowing"}},{$match:{"Borrowing": [ ]}},
    {$project:{_id:0, "memberID": 1, "Borrowing": 1}}])
```

```
    { "memberID" : 3, "Borrowing" : [ ] }
    { "memberID" : 4, "Borrowing" : [ ] }
    { "memberID" : 6, "Borrowing" : [ ] }

> db.members.deleteMany({memberID: {$in: [3,4,6]}})
    { "acknowledged" : true, "deletedCount" : 3 }
```

****