.**Part A. Creating a document database**

**A1. Creating the data model**

**A1.1 Schema design**

The schema given below provides a basic structure for a book store with user comments, book descriptions, and sample page images. The users table stores information about registered users, including their usernames, email addresses, and hashed passwords. The books table stores information about each book, including its title, author, and description, cover image URL, and URL for sample pages. Finally, the comments table stores information about each comment, including the book and user IDs, the content of the comment, and the time it was created.

**User Collection Example Document:**

{

"\_id": ObjectId("623456789012345678901234"),

“user\_id”: “77c30696-73e3-4c6c-8653-94278ab759f9”

"username": "johndoe",

"email": "johndoe@example.com",

"password": "password123",

"created\_at": ISODate("2023-03-11T12:00:00Z")

}

**Book Collection Example Document**

{

book\_id: "3cc261e1-df82-4888-9804-470728648181",

title: "Book 1",

category: "Fiction",

author: "Author 1",

publicationYear: 1951,

storageQuantity: 20,

description: "This book is to be added to database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

}

**Comments Collection Example Document**

{

comment\_id: "bbea4ead-92b7-4c5a-b23a-75afdb8392dc",

book\_id: "3cc261e1-df82-4888-9804-470728648181",

user\_id: "5220ef31-b384-4ece-a00a-3d2ea7176ca0",

date: new Date("2022-01-01"),

text: "Sample Comment One for Book One by UserOne"

}

**A1.2 Justification**

The choice of collections and data modelling approach is based on the requirements of the book store application. The books collection contains information about the books, which includes book details like title, category, author, publication year, storage quantity, description, and sample image pages, and also allows for comments on the book. This is a good choice because it helps to organize and manage the book information and provides a way to store all relevant information in a single document. In addition, it allows for easy updates to the book information and comments. The user’s collection contains information about the store users who can add new books, update the storage of a book, and leave comments on books. This is a good choice because it provides a way to store and manage user information separately from the book information. In addition, it allows for easy updates to user information and user authentication when performing actions like adding new books or leaving comments. Overall, the use of references, embedding or any other data modelling approach is not required in this scenario since the book and user information is relatively straightforward and can be effectively represented in a single collection for books and a single collection for users.

**A2. Creating the database**

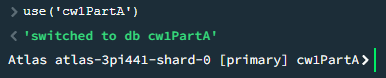
1. **To create and switch to database named as “cw1PartA”**

Code:

use('cw1PartA')

Output:

'switched to db cw1PartA'



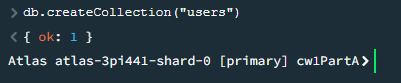
1. **To create and populate “Users” collection:**

**Code:**

db.createCollection("users")

**Output:**

{ ok: 1 }



**Code:**

db.users.insertMany([

{

user\_id: "5220ef31-b384-4ece-a00a-3d2ea7176ca0",

username: "userOne",

email: "userOne@email.xyz",

password: "passwordOne"

},

{

user\_id: "70c8ed4a-f9a6-4b40-a112-0a0af6a4134f",

username: "UserTwo",

email: "userTwo@email.xyz",

password: "passwordTwo"

},

{

user\_id: "014bb153-8619-45bf-a771-c9b9634b1ea8",

username: "userThree",

email: "userThree@email.xyz",

password: "passwordThree"

},

{

user\_id: "a6b5bfae-b77e-4a92-9fab-033b7fb9597d",

username: "userFour",

email: "userFour@email.xyz",

password: "passwordFour"

},

{

user\_id: "be7a0fc1-4a73-43b8-b16e-e75ffd62d039",

username: "userFive",

email: "userFive@email.xyz",

password: "passwordFive"

}

])

**Output:**

{

acknowledged: true,

insertedIds: {

'0': ObjectId("640d7839cbdb3c9723203794"),

'1': ObjectId("640d7839cbdb3c9723203795"),

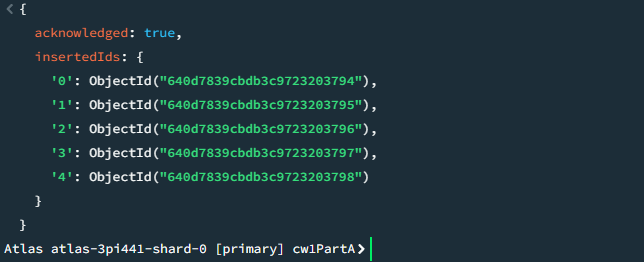
'2': ObjectId("640d7839cbdb3c9723203796"),

'3': ObjectId("640d7839cbdb3c9723203797"),

'4': ObjectId("640d7839cbdb3c9723203798")

}

}



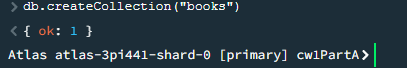
1. **To create and populate “Books” collection:**

**Code:**

db.createCollection("books")

**Output:**

{ ok: 1 }



**Code:**

db.books.insertMany([

{

book\_id: "3cc261e1-df82-4888-9804-470728648181",

title: "Book 1",

category: "Fiction",

author: "Author 1",

publicationYear: 1951,

storageQuantity: 20,

description: "This is the first book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "b95f894e-1df6-4787-9da5-4d46a5e2bb3f",

title: "Book 2",

category: "Fiction",

author: "Author 2",

publicationYear: 1952,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "caec6098-b56d-48f1-be95-a7af4c991bcc",

title: "Book 3",

category: "Fiction",

author: "Author 3",

publicationYear: 1953,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose ",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "19f4b1b6-6000-4c96-9ed2-6b0a908a7d9f",

title: "Book 4",

category: "Science",

author: "Author 4",

publicationYear: 1954,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "1f3eb7ea-10cc-4483-ab59-9c33e1d689ba",

title: "Book 5",

category: "Science",

author: "Author 5",

publicationYear: 1955,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "6fed3b9d-40fa-4fb9-b68d-5e3ec2752e5f",

title: "Book 6",

category: "Political Science",

author: "Author 6",

publicationYear: 1956,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "6f3a07b1-72ba-410f-a155-f5e49c6fc713",

title: "Book 7",

category: "Political Science",

author: "Author 7",

publicationYear: 1957,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "4a4fba06-0d80-4759-b1f8-7c4cda815127",

title: "Book 8",

category: "Arts",

author: "Author 8",

publicationYear: 1958,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "c66ba3d2-3aac-4d8b-b4ea-f9d712cefcbe",

title: "Book 9",

category: "Fiction",

author: "Author 9",

publicationYear: 1959,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

},

{

book\_id: "d3bcc5f2-4a4d-4bc6-a526-615b387bc54f",

title: "Book 10",

category: "Fiction",

author: "Author 10",

publicationYear: 1951,

storageQuantity: 20,

description: " This is the book to be added in the database for assignment purpose.",

samplePagesImages: ["https://www.exaxmple.com/page1.jpg", "https://www.exaxmple.com/page1.jpg"]

}

])

**Output:**

{

acknowledged: true,

insertedIds: {

'0': ObjectId("640d79c0cbdb3c9723203799"),

'1': ObjectId("640d79c0cbdb3c972320379a"),

'2': ObjectId("640d79c0cbdb3c972320379b"),

'3': ObjectId("640d79c0cbdb3c972320379c"),

'4': ObjectId("640d79c0cbdb3c972320379d"),

'5': ObjectId("640d79c0cbdb3c972320379e"),

'6': ObjectId("640d79c0cbdb3c972320379f"),

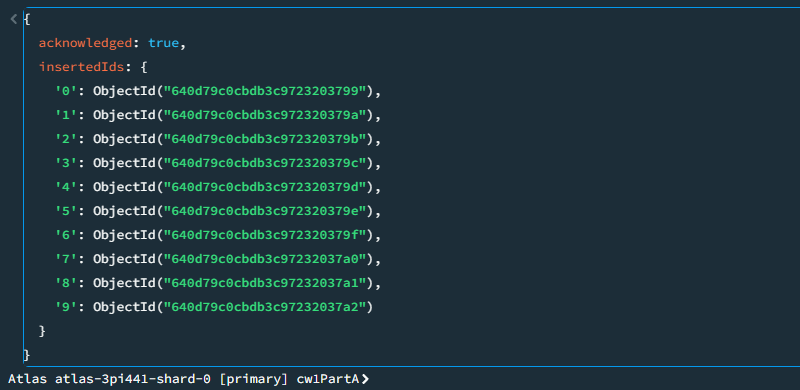
'7': ObjectId("640d79c0cbdb3c97232037a0"),

'8': ObjectId("640d79c0cbdb3c97232037a1"),

'9': ObjectId("640d79c0cbdb3c97232037a2")

}

}



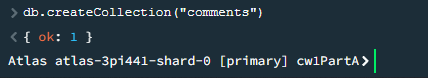
1. **To create and populate Comments collection:**

**Code:**

db.createCollection("comments")

**Output:**

{ ok: 1 }



**Code:**

db.comments.insertMany([

{

comment\_id: "bbea4ead-92b7-4c5a-b23a-75afdb8392dc",

book\_id: "3cc261e1-df82-4888-9804-470728648181",

user\_id: "5220ef31-b384-4ece-a00a-3d2ea7176ca0",

date: new Date("2022-01-01"),

text: "Sample Comment One for Book One by UserOne"

},

{

comment\_id: "84f5437d-2bb7-4bbb-954c-75c93488f3fa",

book\_id: "3cc261e1-df82-4888-9804-470728648181",

user\_id: "70c8ed4a-f9a6-4b40-a112-0a0af6a4134f",

date: new Date("2022-01-01"),

text: "Sample Comment Two for Book One by UserTwo"

},

{

comment\_id: "9893c9da-8f68-4e8d-8930-4e2cfdaf9fb3",

book\_id: "b95f894e-1df6-4787-9da5-4d46a5e2bb3f",

user\_id: "014bb153-8619-45bf-a771-c9b9634b1ea8",

date: new Date("2022-01-01"),

text: "Sample Comment Three for Book Two by UserThree"

},

{

comment\_id: "d2ce2e37-fd64-482d-bb04-8d6994d7190a",

book\_id: "b95f894e-1df6-4787-9da5-4d46a5e2bb3f",

user\_id: "a6b5bfae-b77e-4a92-9fab-033b7fb9597d",

date: new Date("2022-01-01"),

text: "Sample Comment Four for Book Two by UserFour"

},

{

comment\_id: "b0c1cdda-97a1-4e85-92c9-fd7895745d7d",

book\_id: "caec6098-b56d-48f1-be95-a7af4c991bcc",

user\_id: "be7a0fc1-4a73-43b8-b16e-e75ffd62d039",

date: new Date("2022-01-01"),

text: "Sample Comment Five for Book Three by UserFive"

},

{

comment\_id: "7b992fef-1e92-499c-80a6-192ad7be3cd9",

book\_id: "c66ba3d2-3aac-4d8b-b4ea-f9d712cefcbe",

user\_id: "be7a0fc1-4a73-43b8-b16e-e75ffd62d039",

date: new Date("2022-01-01"),

text: "Sample Comment Six for Book Nine by UserFive"

},

{

comment\_id: "218990f4-ea95-431e-b6e4-9a9e937b7107",

book\_id: "d3bcc5f2-4a4d-4bc6-a526-615b387bc54f",

user\_id: "be7a0fc1-4a73-43b8-b16e-e75ffd62d039",

date: new Date("2022-01-01"),

text: "Sample Comment Seven for Book Ten by UserFive"

}

])

**Output:**

{

acknowledged: true,

insertedIds: {

'0': ObjectId("640d7ab0cbdb3c97232037a3"),

'1': ObjectId("640d7ab0cbdb3c97232037a4"),

'2': ObjectId("640d7ab0cbdb3c97232037a5"),

'3': ObjectId("640d7ab0cbdb3c97232037a6"),

'4': ObjectId("640d7ab0cbdb3c97232037a7"),

'5': ObjectId("640d7ab0cbdb3c97232037a8"),

'6': ObjectId("640d7ab0cbdb3c97232037a9")

}

}

****

**A3. Queries**

1. **Use Case 1 Query:**

**Code:**

db.books.aggregate([

{

$match: {

category: "Fiction",

$expr: {

$and: [

{ $gte: [ { $toInt: "$publicationYear" }, 1950 ] },

{ $lte: [ { $toInt: "$publicationYear" }, 1960 ] }

]

}

}

}

])

**Output:**

{

\_id: ObjectId("640d79c0cbdb3c9723203799"),

book\_id: '3cc261e1-df82-4888-9804-470728648181',

title: 'Book 1',

category: 'Fiction',

author: 'Author 1',

publicationYear: 1951,

storageQuantity: 20,

description: This book is to be added to database for assignment purpose..',

samplePagesImages: [

'https://www.exaxmple.com/page1.jpg',

'https://www.exaxmple.com/page1.jpg'

]

}

{

\_id: ObjectId("640d79c0cbdb3c972320379a"),

book\_id: 'b95f894e-1df6-4787-9da5-4d46a5e2bb3f',

title: 'Book 2',

category: 'Fiction',

author: 'Author 2',

publicationYear: 1952,

storageQuantity: 20,

description: This book is to be added to database for assignment purpose..',

samplePagesImages: [

'https://www.exaxmple.com/page1.jpg',

'https://www.exaxmple.com/page1.jpg'

]

}

{

\_id: ObjectId("640d79c0cbdb3c97232037a2"),

book\_id: 'd3bcc5f2-4a4d-4bc6-a526-615b387bc54f',

title: 'Book 10',

category: 'Fiction',

author: 'Author 10',

publicationYear: 1951,

storageQuantity: 20,

description: This book is to be added to database for assignment purpose..',

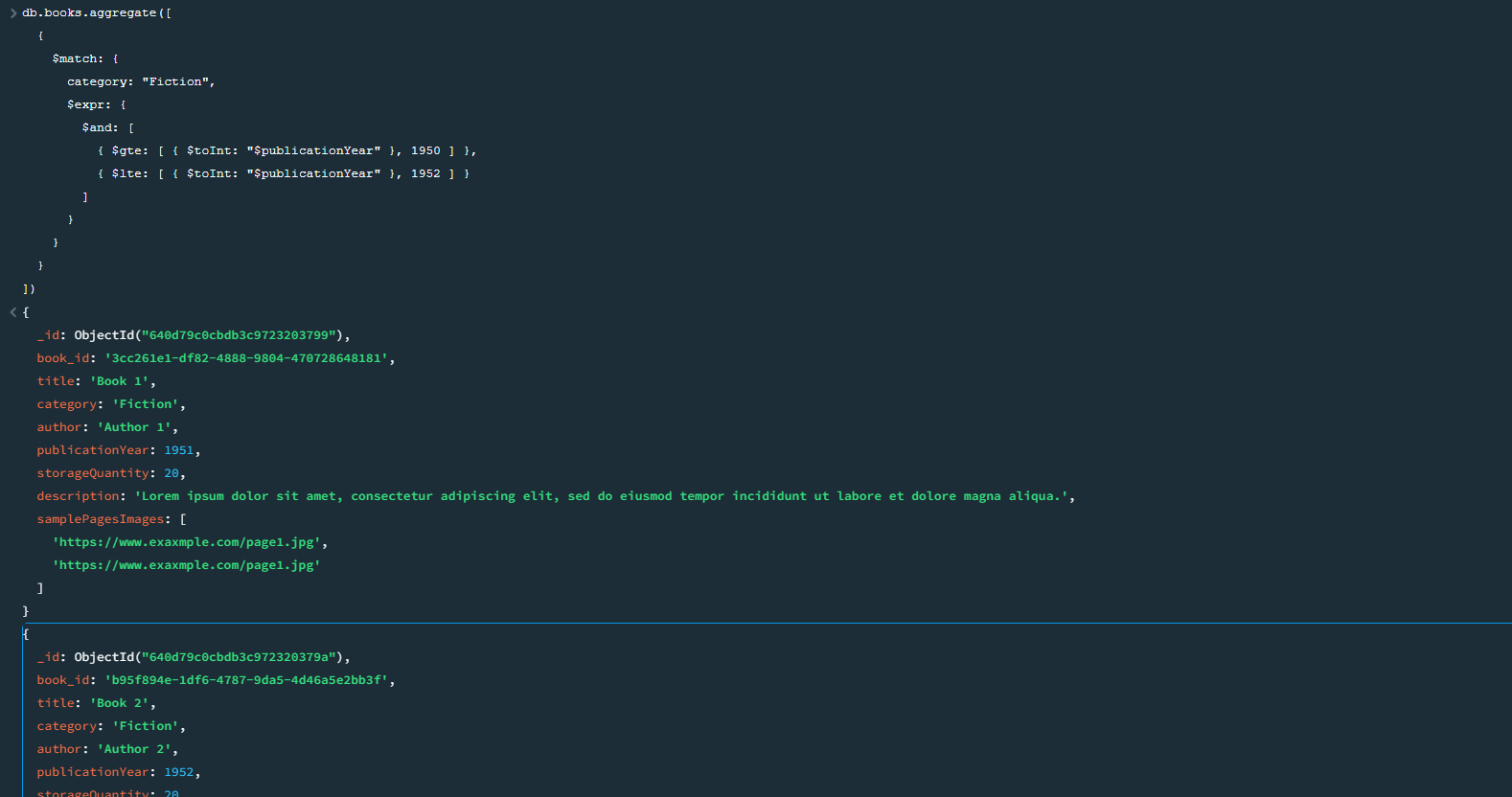
samplePagesImages: [

'https://www.exaxmple.com/page1.jpg',

'https://www.exaxmple.com/page1.jpg'

]

}

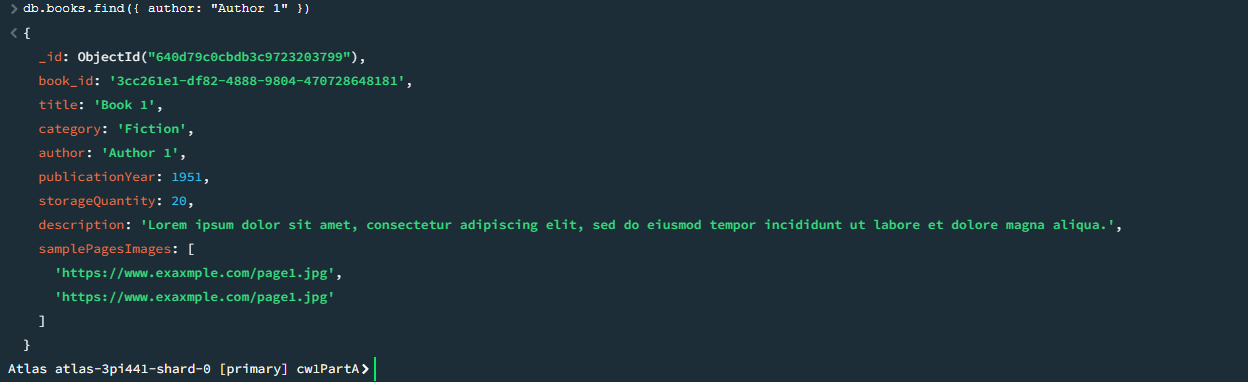


1. **Use Case 2 Query:**

**Code:**

db.books.find({ author: "Author 1" })

**Output:**

****

1. **Use Case 3 Query:**

**Code:**

db.comments.insert({

comment\_id: "7672efce-7285-4cd9-8094-cec51cbf10a4",

book\_id: "3cc261e1-df82-4888-9804-470728648181",

user\_id: "5220ef31-b384-4ece-a00a-3d2ea7176ca0",

date: new Date(),

text: "New Sample Comment"

})

**Output:**

{

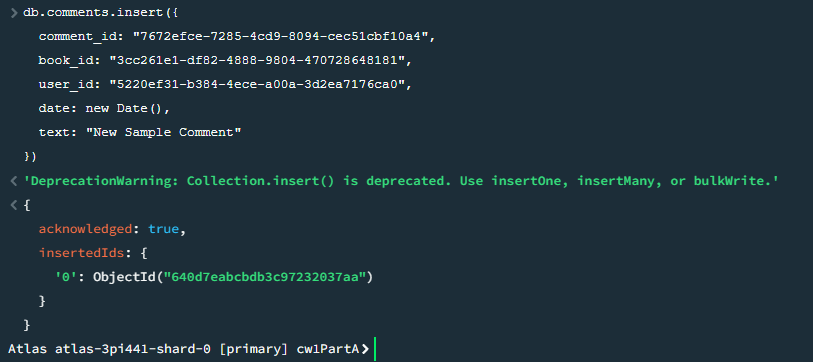
acknowledged: true,

insertedIds: {

'0': ObjectId("640d7eabcbdb3c97232037aa")

}

}



1. **Use Case 4 Query:**

db.books.aggregate([

{ $match: { description: / This book is to be added./i } },

{

$lookup: {

from: "comments",

localField: "\_id",

foreignField: "comment\_id",

as: "comments"

}

},

{ $unwind: { path: "$comments", preserveNullAndEmptyArrays: true } },

{

$lookup: {

from: "users",

localField: "user\_id",

foreignField: "user\_id",

as: "comments.user"

}

},

{

$group: {

\_id: "$\_id",

title: { $first: "$title" },

author: { $first: "$author" },

description: { $first: "$description" },

category: { $first: "$category" },

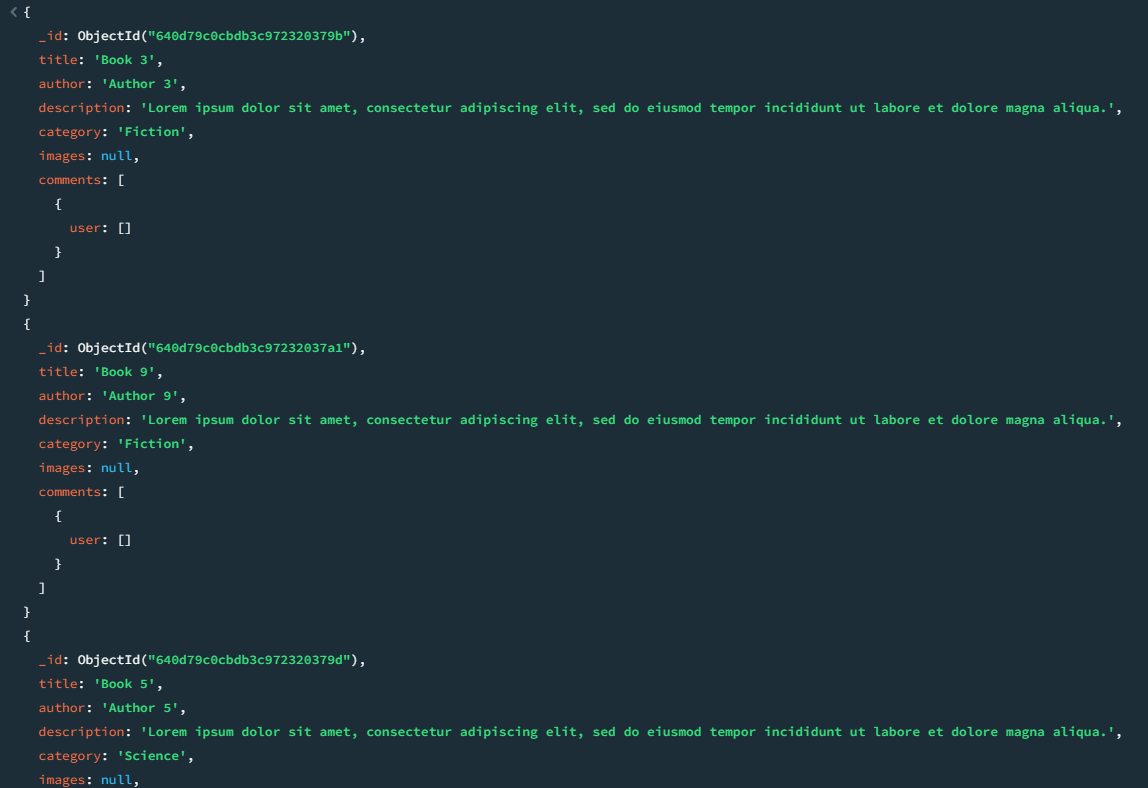
images: { $first: "$images" },

comments: { $push: "$comments" }

}

}

])



**Part B. Creating a graph database**

**B1. Creating the database**

1. **To create database cw1PartB:**

**Cypher:**

CREATE DATABASE cw1PartB

**Output:**

(1 system update, no records)

1. **To switch to databases cw1PartB:**

**Cypher:**

Use cw1PartB

**Output:**

Queries from this point and forward are using the database cw1partb as the target.

1. **To create Airport Nodes:**

**Cypher:**

CREATE (:Airport {name: "JFK"})

CREATE (:Airport {name: "LAX"})

CREATE (:Airport {name: "LHR"})

CREATE (:Airport {name: "CDG"})

CREATE (:Airport {name: "HND"})

**Output:**

Added 5 labels, created 5 nodes, set 5 properties, completed after 12 ms.

1. **To create Flight nodes:**

**Cypher:**

CREATE (:Flight {airline: "AirlineOne", date: "2023-02-17", distance: 15, duration: 50})

CREATE (:Flight {airline: "AirlineTwo", date: "2023-02-18", distance: 16, duration: 55})

CREATE (:Flight {airline: "AirlineThree", date: "2023-02-19", distance: 17, duration: 60})

CREATE (:Flight {airline: "AirlineFour", date: "2023-02-20", distance: 18, duration: 65})

CREATE (:Flight {airline: "AirlineFive", date: "2023-02-21", distance: 10, duration: 25})

CREATE (:Flight {airline: "AirlineSix", date: "2023-02-22", distance: 15, duration: 35})

CREATE (:Flight {airline: "AirlineSeven", date: "2023-02-23", distance: 18, duration: 38})

CREATE (:Flight {airline: "AirlineEight", date: "2023-02-24", distance: 20, duration: 38})

CREATE (:Flight {airline: "AirlineNine", date: "2023-02-25", distance: 30, duration: 48})

CREATE (:Flight {airline: "AirlineTen", date: "2023-02-26", distance: 39, duration: 38})

**Output:**

Added 10 labels, created 10 nodes, set 40 properties, completed after 31 ms.

1. **To create Ticket nodes:**

**Cypher:**

CREATE (:Ticket {class: "FirstClass", price: 20})

CREATE (:Ticket {class: "Economy", price: 10})

1. **To create relationship between Flights and Tickets:**

**Cypher:**

MATCH (f:Flight {date: "2023-02-17"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 109 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-17"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 15 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-18"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 18 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-18"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 18 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-19"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 15 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-19"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 23 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-20"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 23 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-20"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 18 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-21"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 16 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-21"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 16 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-22"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 18 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-22"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 10 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-23"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 22 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-23"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 25 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-24"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 18 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-24"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 19 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-25"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 10 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-25"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 17 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-26"})

MATCH (t:Ticket {class: "FirstClass"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 18 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-26"})

MATCH (t:Ticket {class: "Economy"})

CREATE (f)-[:HAS]->(t)

**Output:**

Created 1 relationship, completed after 21 ms.

1. **To create relation between Flights and Airports**

**Cypher:**

MATCH (f:Flight {date: "2023-02-17"})

MATCH (a:Airport {name: "JFK"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 20 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-18"})

MATCH (a:Airport {name: "LAX"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 40 ms.

**Cypher:**

MATCH (f:Flight {date: "2023-02-19"})

MATCH (a:Airport {name: "LHR"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 23 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-20"})

MATCH (a:Airport {name: "CDG"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 21 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-21"})

MATCH (a:Airport {name: "HND"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 16 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-22"})

MATCH (a:Airport {name: "JFK"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 28 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-23"})

MATCH (a:Airport {name: "LAX"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 26 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-24"})

MATCH (a:Airport {name: "LHR"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 28 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-25"})

MATCH (a:Airport {name: "CDG"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 18 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-26"})

MATCH (a:Airport {name: "HND"})

CREATE (f)-[:ORIGIN]->(a)

**Output:**

Created 1 relationship, completed after 20 ms

//DESTINATION

**Cypher:**

MATCH (f:Flight {date: "2023-02-17"})

MATCH (a:Airport {name: "LAX"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 21 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-18"})

MATCH (a:Airport {name: "LHR"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 17 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-19"})

MATCH (a:Airport {name: "CDG"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 23 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-20"})

MATCH (a:Airport {name: "HND"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 16 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-21"})

MATCH (a:Airport {name: "JFK"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 18 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-22"})

MATCH (a:Airport {name: "LAX"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 16 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-23"})

MATCH (a:Airport {name: "LHR"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 110 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-24"})

MATCH (a:Airport {name: "CDG"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 23 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-25"})

MATCH (a:Airport {name: "HND"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 26 ms

**Cypher:**

MATCH (f:Flight {date: "2023-02-26"})

MATCH (a:Airport {name: "JFK"})

CREATE (f)-[:DESTINATION]->(a)

**Output:**

Created 1 relationship, completed after 29 ms

**B2. Queries**

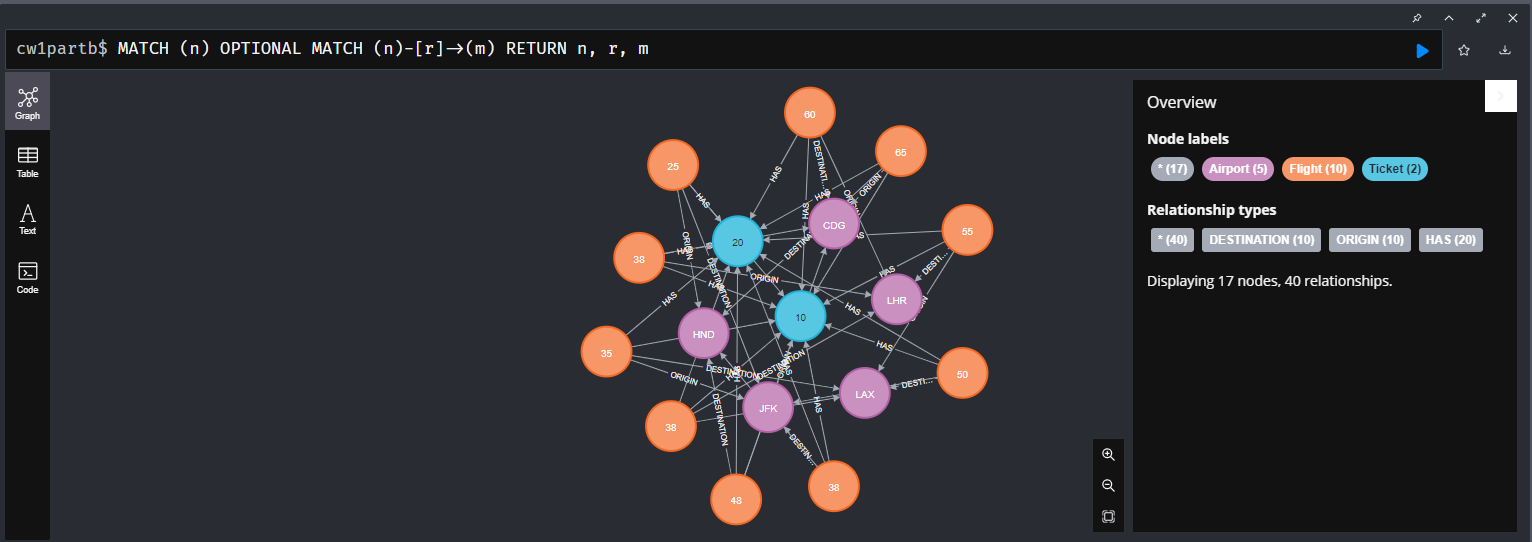
**Query 1:**

MATCH (n)

OPTIONAL MATCH (n)-[r]->(m)

RETURN n, r, m

**Output:**



**Query 2:**

MATCH (a:Airport {name: "JFK"})<-[:ORIGIN]-(f:Flight)

RETURN f.airline, f.date, f.distance, f.duration

**Output:**

╒════════════╤════════════╤════════════╤════════════╕

│"f.airline" │"f.date" │"f.distance"│"f.duration"│

╞════════════╪════════════╪════════════╪════════════╡

│"AirlineSix"│"2023-02-22"│15 │35 │

├────────────┼────────────┼────────────┼────────────┤

│"AirlineOne"│"2023-02-17"│15 │50 │

└────────────┴────────────┴────────────┴────────────┘



**Query 3:**

MATCH (origin:Airport {name: "JFK"})<-[:ORIGIN]-(f:Flight)-[:DESTINATION]->(destination:Airport)

RETURN destination.name

**Output:**

╒══════════════════╕

│"destination.name"│

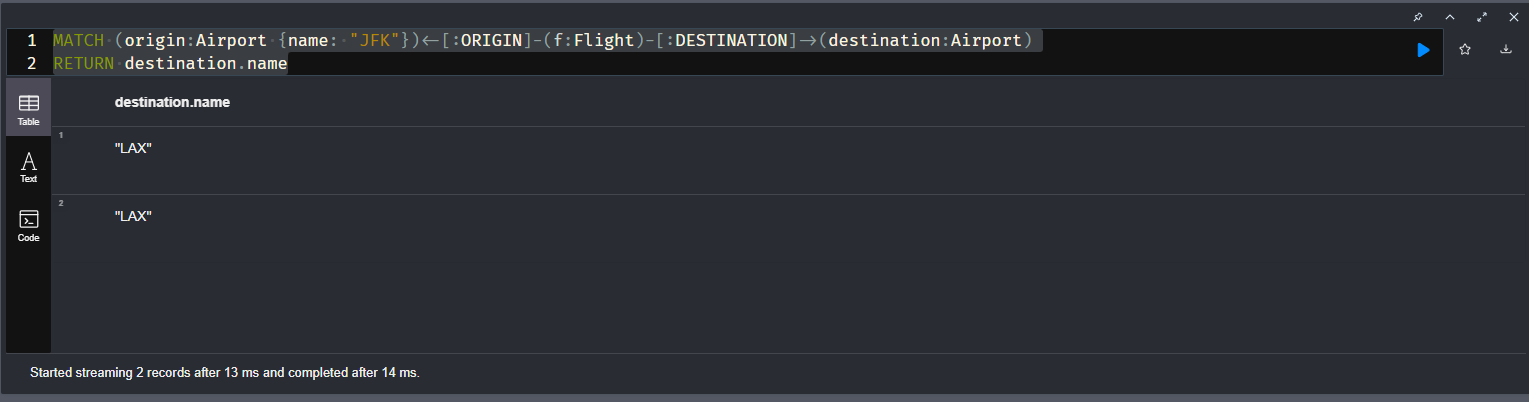
╞══════════════════╡

│"LAX" │

├──────────────────┤

│"LAX" │

└──────────────────┘



**Query 4:**

MATCH (f:Flight {airline: "AirlineOne"})

CREATE (f)-[:HAS]->(:Ticket {class: "Business", price: 100})

**Output:**

Added 1 label, created 1 node, set 2 properties, created 1 relationship, completed after 16 ms.

**Query 5:**

MATCH (destination:Airport {name: "LAX"})<-[to:DESTINATION]-(f:Flight)-[has:HAS]->(t:Ticket)

RETURN t.class, t.price

**Output:**

╒════════════╤═════════╕

│"t.class" │"t.price"│

╞════════════╪═════════╡

│"Economy" │10 │

├────────────┼─────────┤

│"FirstClass"│20 │

├────────────┼─────────┤

│"Business" │100 │

├────────────┼─────────┤

│"Economy" │10 │

├────────────┼─────────┤

│"FirstClass"│20 │

└────────────┴─────────┘

