Programming Lab 3

Assignment 07

Installation of MongoDB and Operations on MongoDB

Name : [Shantanu kantak](mailto:kantakshantanu20@gmail.com)

PRN : 21510001

Batch : T-8

Problem Statement 01:

1]First off, you need a database to connect to. MongoDB doesn’t have a “create

database” command. Instead, it is going to create one for you when you try to save

something into it.

 Install and Connect to the mongoDB.

 Create a collection called ‘games’. We’re going to put some games in it.

 Add 5 games to the database. Give each document the following properties: name,

genre, rating (out of 100). If you make some mistakes and want to clean it out, use

remove()on your collection.

 Write a query that returns all the games.

 Write a query to find one of your games by name without using limit(). Use the

findOne method. Look how much nicer it’s formatted!.

 Write a query that returns the 3 highest rated games.

 Update your two favourite games to have two achievements called ‘Game Master’

and ‘Speed Demon’, each under a single key. Show two ways to do this. Do the first

using update()and do the second using save(). Hint: for save, you might want to

query the object and store it in a variable first.

 Write a query that returns all the games that have both the ‘Game Maser’ and the

‘Speed Demon’ achievements.

 Write a query that returns only games that have achievements. Not all of your

games should have achievements, obviously.

Ans : Commands in the following order should be run once MongoSh Installation is done:

1]use dbs

2]db.createCollection(“games”)

3]db.Games.insertMany([

{ name: 'Game1', genre: 'Adventure', rating: 85 },

{ name: 'Game2', genre: 'Action', rating: 92 },

{ name: 'Game3', genre: 'RPG', rating: 88 },

{ name: 'Game4', genre: 'Sports', rating: 78 },

{ name: 'Game5', genre: 'Puzzle', rating: 95 }

])

4] db.Games.find({})

5] db.Games.findOne({ name: 'Game1' })

6] db.Games.find().sort({ rating: -1 }).limit(3)

7] // Using update() method

db.Games.update(

{ name: 'Game1' },

{ $set: { achievements: ['Game Master', 'Speed Demon'] }

})

let game2 = db.Games.findOne({ name: 'Game2' })

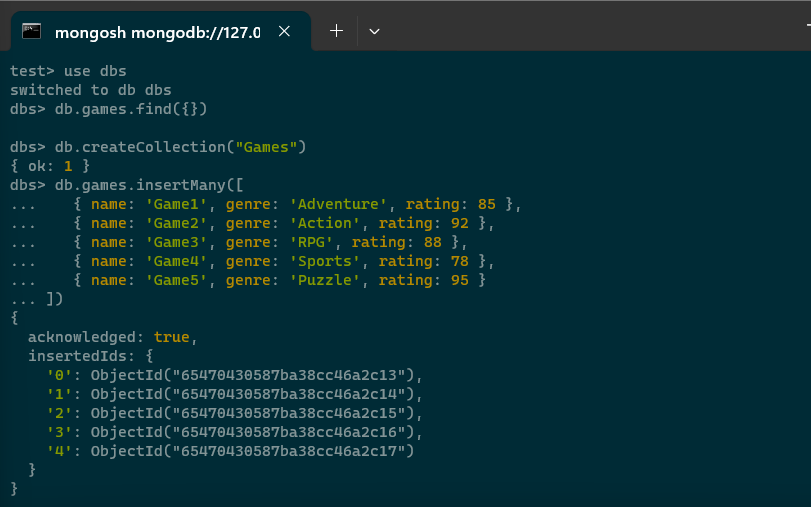
game2.achievements = ['Game Master', 'Speed Demon']

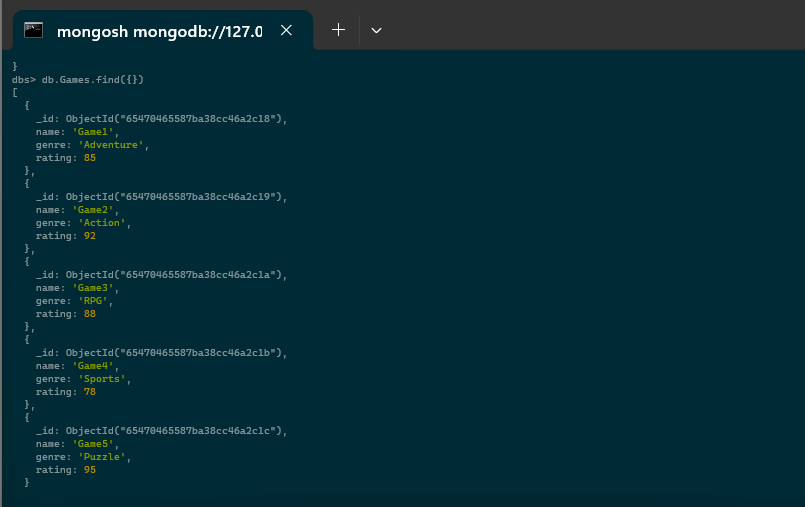
db.Games.save(game2)

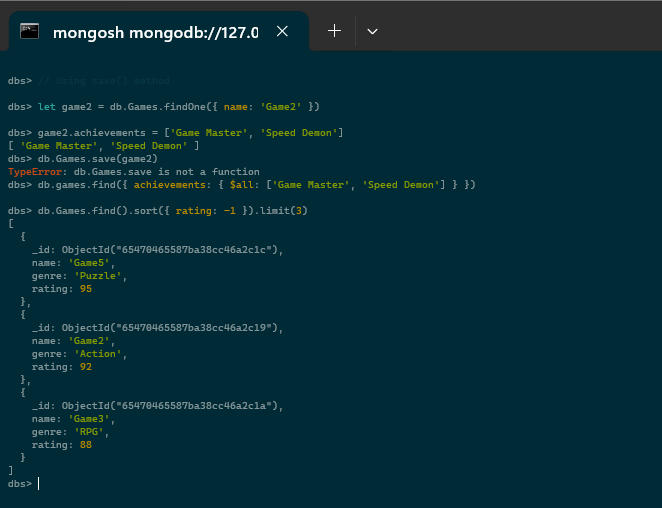
8] db.games.find({ achievements: { $all: ['Game Master', 'Speed Demon'] } })

9] db.games.find({ achievements: { $exists: true } })

Screenshots:

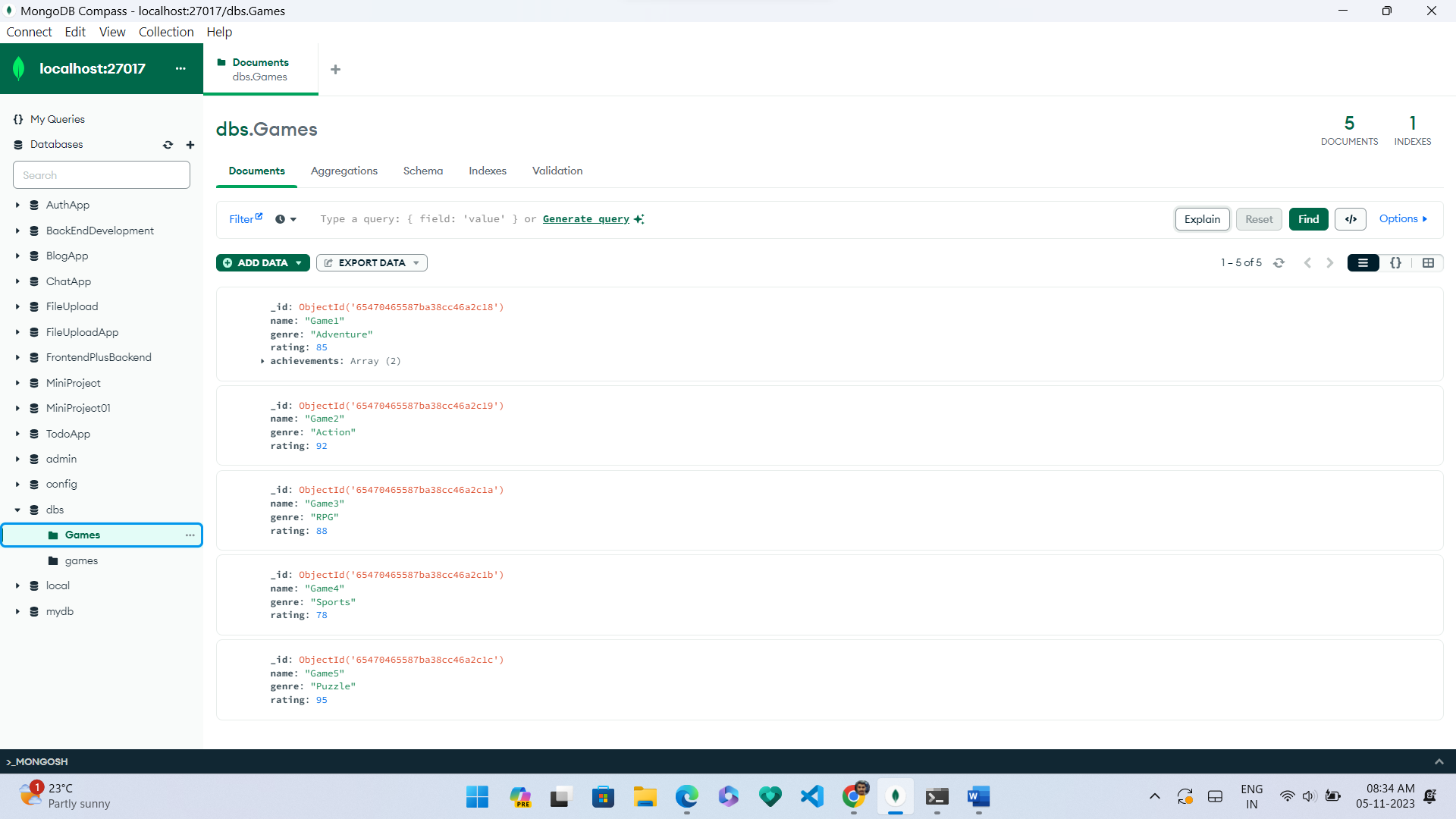






Screenshot of Database and Collection in the MongoDB Compass:





Problem Statement 02:

Using map-reduce method of MongoDB, write a reduce that calculates the total score from all games for each player and check the output.

Ans: Give the following queries to the mongosh:

1] db.createCollection("gameResults")

2] db.gameResults.insertMany([

{

"player": "PlayerA",

"game": "Game1",

"score": 100

},

{

"player": "PlayerB",

"game": "Game1",

"score": 120

},

{

"player": "PlayerA",

"game": "Game2",

"score": 80

},

{

"player": "PlayerB",

"game": "Game2",

"score": 110

},

{

"player": "PlayerA",

"game": "Game3",

"score": 90

},

{

"player": "PlayerB",

"game": "Game3",

"score": 130

}

])

3] var mapFunction = function() {

emit(this.player, this.score);

};

4] var reduceFunction = function(key, values) {

return Array.sum(values);

};

5] db.gameResults.mapReduce(

mapFunction,

reduceFunction,

{

out: "playerScores"

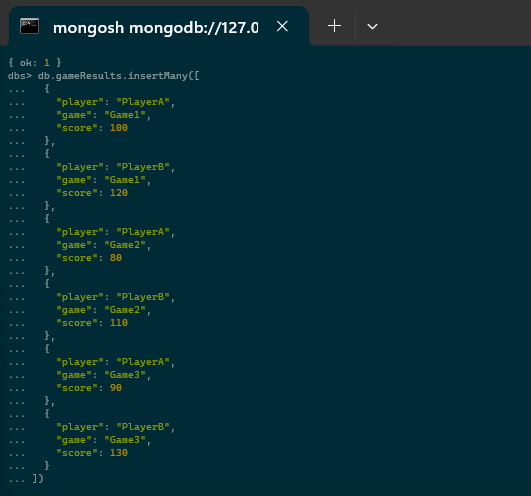
}

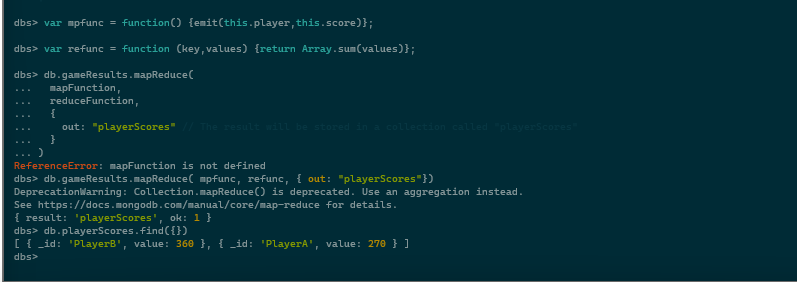
)

6] db.playerScores.find()

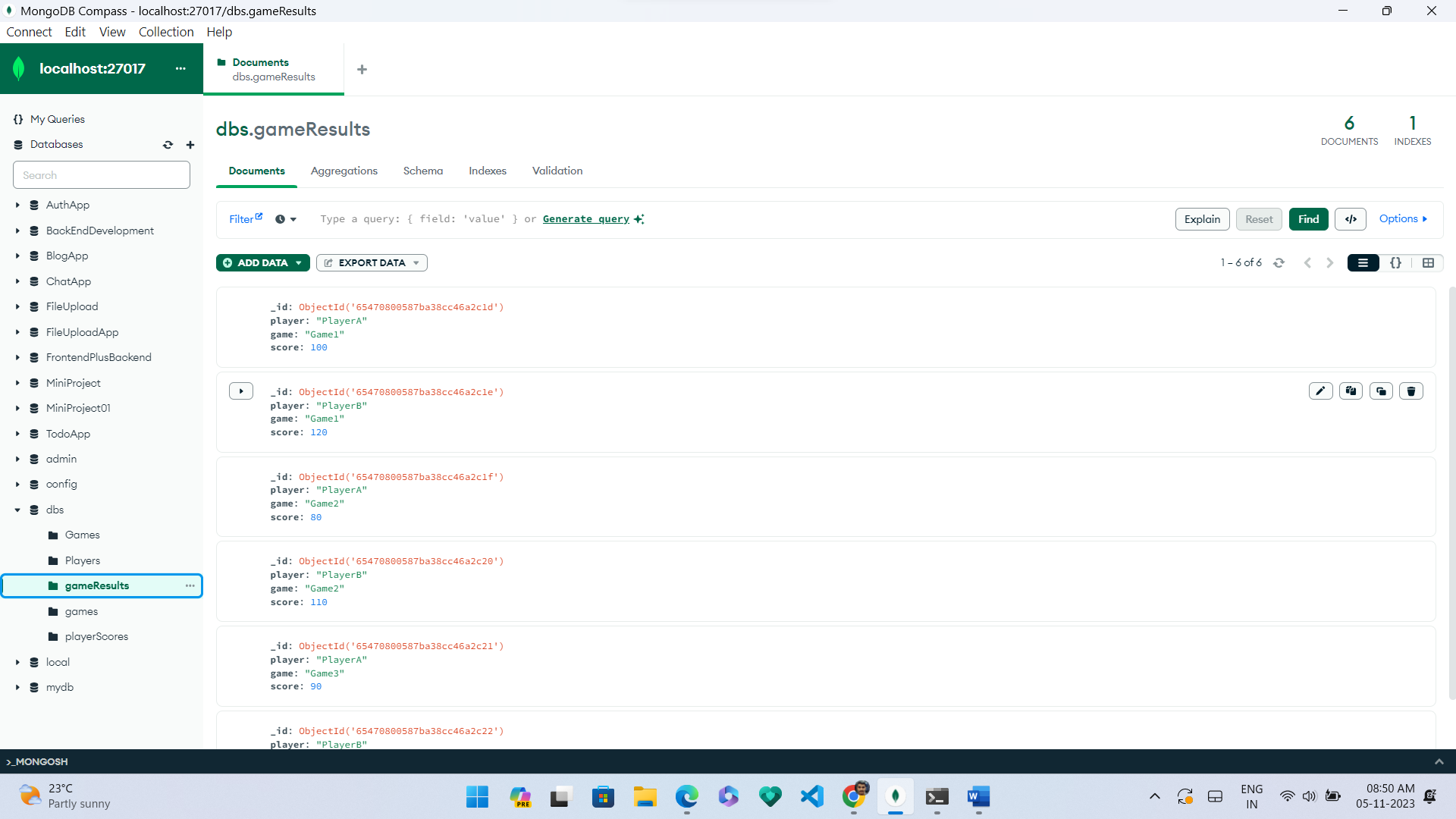
Screenshots:

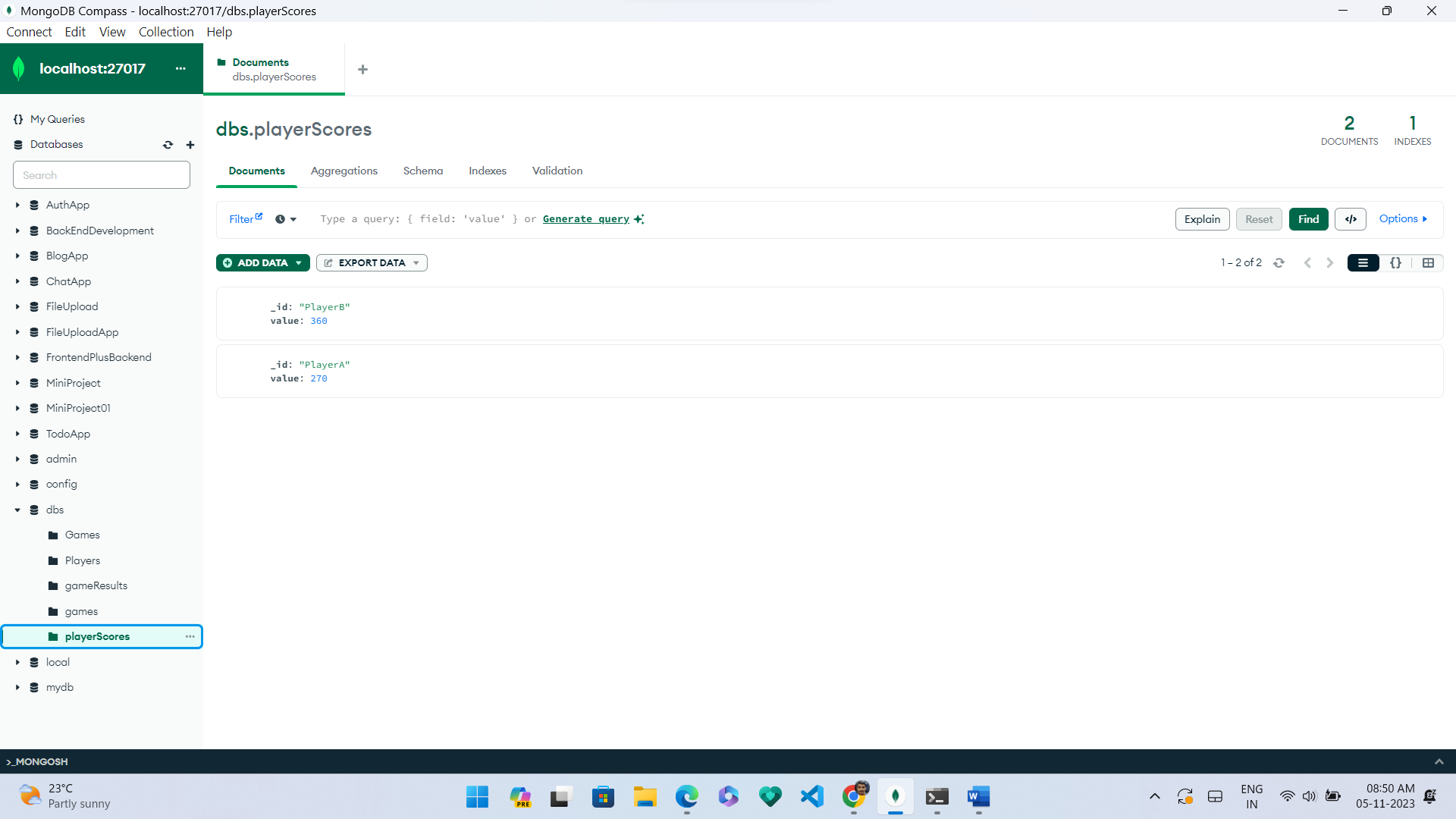






Screenshots from MongDB Compass:





Problem Statement 03:

Use the REST API to show all the game data stored in the db from the games

collection.

 Output all of the available returned data in an html table in the following format:

1]Game 2]Publisher 3]Release 4]Date 5]Rating 6]Average Score

Ans:

Here’s the code:

Backend:

const express **=** require('express')**;**

const mongoose **=** require('mongoose')**;**

const app **=** express()**;**

const port **=** process**.**env**.**PORT **||** 3000**;**

const cors **=** require('cors')**;**

*// Replace with your MongoDB connection string*

const dbURI **=** 'mongodb://127.0.0.1:27017/dbs'**;**

*// Connect to MongoDB*

mongoose**.**connect(dbURI, { useNewUrlParser**:** true, useUnifiedTopology**:** true })

**.**then(() **=>** {

    console**.**log('Connected to MongoDB')**;**

  })

**.**catch(**err** **=>** {

    console**.**error('Error connecting to MongoDB', err)**;**

  })**;**

*// Define a MongoDB schema for the "games" collection*

const gameSchema **=** **new** mongoose**.**Schema({

  Game**:** *String*,

  Publisher**:** *String*,

  'Release Date'**:** *String*,

  Rating**:** *Number*,

  'Average Score'**:** *Number*,

})**;**

*// Create a model based on the schema*

const Game **=** mongoose**.**model('Game', gameSchema)**;**

*// Set up middleware to parse JSON requests*

app**.**use(express**.**json())**;**

app**.**use(cors())**;**

*// Define a route to retrieve game data*

app**.**get('/games', async (**req**, **res**) **=>** {

**try** {

    const games **=** **await** Game**.**find()**;**

    res**.**json(games)**;**

*// console.log(games)*

**return** games**;**

  } **catch** (err) {

    console**.**error(err)**;**

    res**.**status(500)**.**send('Internal Server Error')**;**

  }

})**;**

*// Start the Express server*

app**.**listen(port, () **=>** {

  console**.**log(`Server is running on port ${port}`)**;**

})**;**

Frontend:

<!DOCTYPE *html*>

<html>

<head>

  <title>Game Data</title>

</head>

<body>

  <h1>Game Data</h1>

  <table border**=**"1">

    <thead>

      <tr>

        <th>Game</th>

        <th>Publisher</th>

        <th>Release Date</th>

        <th>Rating</th>

        <th>Average Score</th>

      </tr>

    </thead>

    <tbody ***id*=**"gameTableBody"></tbody>

  </table>

  <script>

*// Replace with the actual URL of your Express API*

    const apiUrl **=** 'http://127.0.0.1:3000/games'**;**

*// Function to fetch game data from the API*

    async function fetchGameList() {

**try** {

        const response **=** **await** fetch(apiUrl)**;**

        const games **=** **await** response**.**json()**;**

        console**.**log(games)**;**

        const tableBody **=** document**.**getElementById('gameTableBody')**;**

        games**.**forEach(**game** **=>** {

          const row **=** document**.**createElement('tr')**;**

          row**.***innerHTML* **=** `

            <td>${game**.***Game*}</td>

            <td>${game**.***Publisher*}</td>

            <td>${game['Release Date']}</td>

            <td>${game**.***Rating*}</td>

            <td>${game['Average Score']}</td>

          `**;**

          tableBody**.**appendChild(row)**;**

        })**;**

      } **catch** (error) {

        console**.**error(error)**;**

      }

    }

*// Call the function to fetch and display game data*

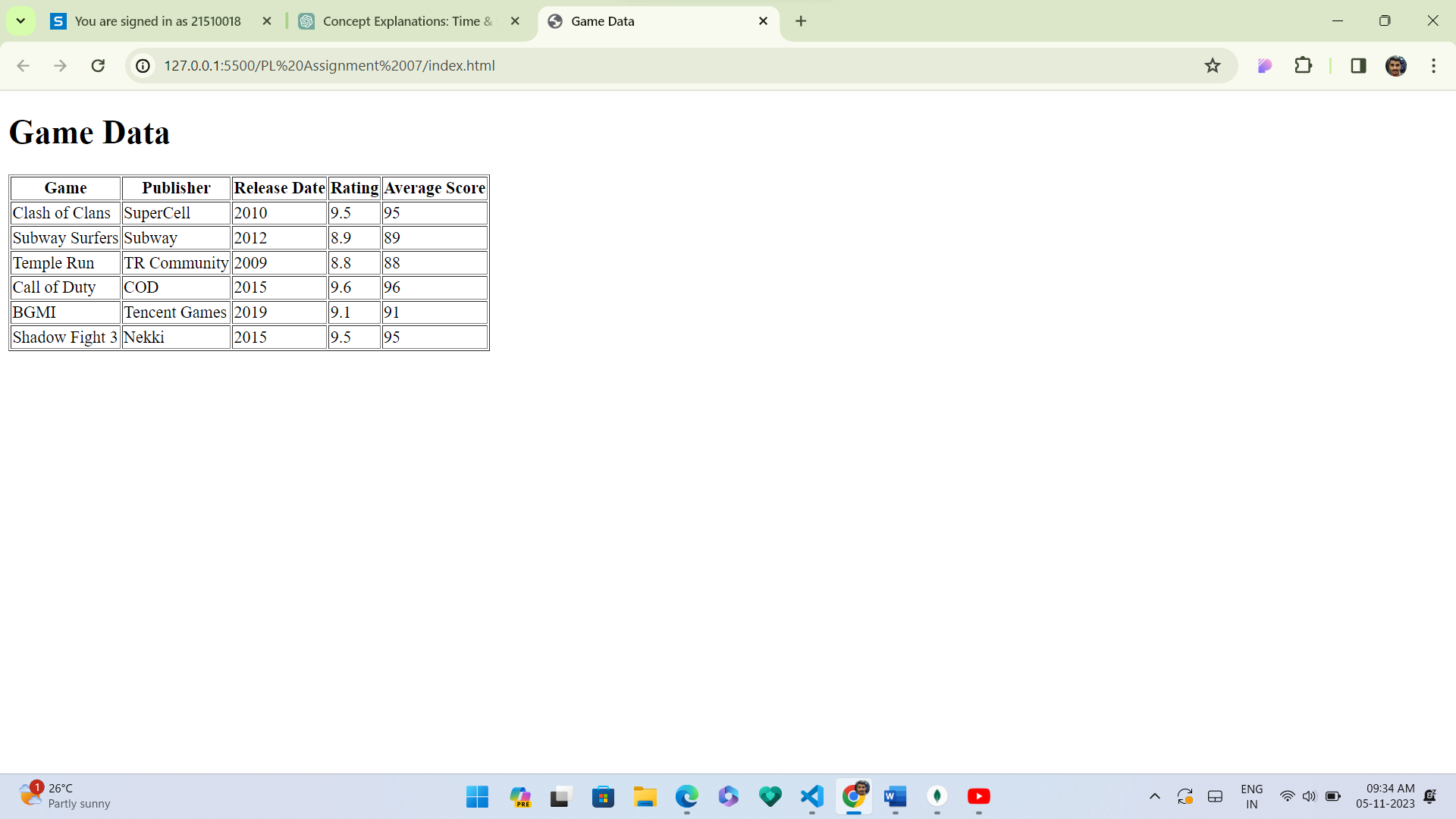
    fetchGameList()**;**

  </script>

</body>

</html>

ScreenShots:



End of Assignment.