FACULTY OF COMPUTING AND INFORMATICS

TIS3351 Advanced Database

TRIMESTER 2 2022/23

Assignment 2

Lecture Section : TC1L
Tutorial Section : TT2L

for:
Mr. Muhd Faizal Ahmad

from:

Student ID	Name	Email Address	Phone No.
1191101213	Raven Lim Zhe Xuan	1191101213@student.mmu.edu.my	01155873318
1191100776	Ngaindran A/L Kanaseelanayagam	1191100776@student.mmu.edu.my	01123569102
1191100556	Liew Jiann Shen	1191100556@student.mmu.edu.my	0174922881
1191100350	Fong Zheng Wei	1191100350@student.mmu.edu.my	0123229588

Table of Contents

Introduction

This report focuses on the utilisation of NoSQL databases, specifically MongoDB, and how it can be better implemented within the scope of a Movie Ticketing System. The report is divided into two parts: Part A emphasises data creation, while Part B discusses the relevance of Big Data within the context of the system.

First and foremost, touching upon the data modelling segment of the report, where we discuss the format of the data that will be inserted into each respective collection. Said instruction is followed by the data dictionary to analyse the structure of each individual collection and respective commands needed to create the database on the MongoDB platform, using the command line upon connection to the MongoDB server. Also found in this assignment includes several commands that could be used to spice up the database management on the NoSQL database.

We would like to express our gratitude to our friends for their unwavering support throughout the completion of this assignment. We would also like to extend our thanks to our esteemed lecturers, Mr. Faizal and Mr. Khye Neoh, for their invaluable guidance and assistance throughout the process.

By following the outline provided in this report, we aim to showcase the benefits and practical implementation of NoSQL databases, specifically MongoDB, within the context of a Movie Ticketing System.

The attached MongoDB database execution script submitted with the submission is in a javascript format. This can be run in the MongoSH terminal by copying intended contents to it.

Part A

1. Data Modeling

Touches upon data structures and the organisation of data within a database system. The main goal is to ensure that the data is accurately represented, organised, and stored in a way that supports efficient data retrieval and manipulation.

We have decided to categorise our tables (otherwise known as collections since we are using MongoDB) into several parts of the database. Each part plays an important role as it;s own object to ensure a more cohesive structure between them.

Figure 1.1: Final Data Modelling Diagram of the Movie Ticketing System

2. Data Dictionary

This data dictionary is meant to provide clearer documentation with detailed information about the structure, organisation, and characteristics of the data within this database system. It serves as a reference guide for understanding the meaning, usage and relationships of the data elements in this database.

Some of the information that will play a crucial part in the database creation process includes Data Types ,Constraints, Relationships, Data Usage and Data Access.

Collections Name	Object Name	Data Type	Unique
Customer	_id	oid	True
	name	String	
	email	String	
	contact	String	
	membership	Boolean	
Cinema	_id	oid	True
	name	String	
	address	Object	
	address.street	String	
	address.city	String	
	address.state	String	
	halls	Array[String]	
Movie	_id	oid	True
	name	String	
	relase_date	Date	
	running_minutes	Integer	
	genre	String	

	language	String	
	classification	String	
	categories	Array(Object)	
	categories[].name	String	
	categories[].price	Double	
Promotion	_id	oid	True
	code	String	
	description	String	
	discount	Double	
	promo_start	Date	
	promo_end	Date	
Screening	_id	oid	True
	movie	oid	
	category	String	
	cinema	oid	
	hall	String	
	time	Date	
Ticket	_id	oid	True
	customer	oid	
	screening	oid	
	promotion	oid	
	seats	Array(String)	
	nett_price	Double	

Figure 2.1: Data Dictionary of the Movie Ticketing System

3. Database Creation

As we venture from the Data Dictionary, we begin the Data Creation process. This begins with a connection to the MongoDB database. For this setup, we have set up a MongoDB database locally on our machine, alongside using MongoDBCompass to manage the database. Despite the GUI elements of MongoDBCompass, we have decided to use the command-line based arguments instead.

```
use('movie_assignment2');



// Q3 Database Creation
db.createCollection('customer',
  {
    validator: {
      $jsonSchema: {
        bsonType: 'object',
        required: ['_id', 'name', 'email', 'contact'],
        properties: {
          name: {
            bsonType: 'string',
            description: 'Enter a name'
          },
          email: {
            bsonType: 'string',
            description: 'Enter an email'
          },
          contact: {
            bsonType: 'string',
            description: 'Enter a phone number'
          }
        }
      }
    }
  });
db.createCollection('cinema',
  {
    validator: {
```

```
      $jsonSchema: {
        bsonType: 'object',
        required: ['_id', 'name', 'address'],
        properties: {
          name: {
            bsonType: 'string',
            description: 'Requires a name'
          },
          halls: {
            bsonType: 'array',
            description: 'Must be an array of numbers to indicate hall
number',
            items: {
              bsonType: 'string'
            }
          }
        }
      }
    }
  });
db.createCollection('movie',
  {
    validator: {
      $jsonSchema: {
        bsonType: 'object',
        required: ['_id', 'name', 'language', 'classification'],
        properties: {
          name: {
            bsonType: 'string',
            description: 'Do insert a name'
          },
          release_date: {
            bsonType: 'date',
            description: 'Insert a valid date'
          },
          running_minutes: {
            bsonType: 'int',
            description: 'Give the movie them running minutes'
          },
          genre: {
            bsonType: 'string',
```

```
            description: 'Do insert a genre'
          },
          language: {
            bsonType: 'string',
            description: 'Do insert a language'
          },
          classification: {
            bsonType: 'string',
            description: 'Please insert a classification'
          }
        }
      }
    }
  });
db.createCollection('promotion',
  {
    validator: {
      $jsonSchema: {
        bsonType: 'object',
        required: ['_id', 'code'],
        properties: {
          code: {
            bsonType: 'string',
            description: 'Requires a name'
          }
        }
      }
    }
  });
db.createCollection('screening',
  {
    validator: {
      $jsonSchema: {
        bsonType: 'object',
        required: ['_id', 'movie', 'cinema'],
      }
    }
  });
db.createCollection('ticket',
  {
    validator: {
```

```
      $jsonSchema: {
        bsonType: 'object',
        required: ['_id', 'customer', 'screening', 'seats', 'nett_price'],
        properties: {
          seats: {
            bsonType: 'array',
            description: 'Must be an array of numbers to indicate seat
number',
            items: {
              bsonType: 'string'
            }
          }
        }
      }
    }
  });
```

Figure 3.1: Source code of the database creation

Starting with the use command, to set the database name. Should there not be any database name with the following created prior, it will create a database with the following name and access it directly. This is followed by the “createCollection” command with some of them to include validators to help verify the input that goes into the database. Validators allow to ensure that the value of a specific data type is being inserted into the database.

4. Sample Data

Generation of at least 10 entries into the database have been inserted into this now-created database. The way this is done is via the generation of random data and organising it into a specific format, to be inserted into the database array.

ObjectID links are then properly connected while creating the data. The object ID is first generated into a bunch of arrays, and they can be referenced in other collections by ArrayIDs that contains the objects

```
// Q4 Sample data
// ObjectID generation
customerIDs = [ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId(),
ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId()];
cinemaIDs = [ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId(),
ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId()];
movieIDs = [ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId(),
ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId()];
promotionIDs = [ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId(),
ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId()];
screeningIDs = [ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId(),
ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId()];
ticketIDs = [ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId(),
ObjectId(), ObjectId(), ObjectId(), ObjectId(), ObjectId()];
// Customer
db.customer.insertMany(
  [
    {
      "_id": customerIDs[0],
      "name": "Fong Zheng Wei",
      "email": "fongzw@gmail.com",
      "contact": "0123229588",
      "membership": true
    },
```

```
{
  "_id": customerIDs[1],
  "name": "Liew Jiann Shen",
  "email": "ljs0056@gmail.com",
  "contact": "0165846148",
  "membership": true
},
{
  "_id": customerIDs[2],
  "name": "Nagaindran",
  "email": "naga@gmail.com",
  "contact": "017565956",
  "membership": false
},
{
  "_id": customerIDs[3],
  "name": "Raven Lim",
  "email": "raven1010@gmail.com",
  "contact": "0125668646",
  "membership": false
},
{
  "_id": customerIDs[4],
  "name": "Khye Neoh",
  "email": "ky@gmail.com",
  "contact": "012656568",
  "membership": false
},
{
  "_id": customerIDs[5],
  "name": "Tee Wei How",
  "email": "twh0001@gmail.com",
  "contact": "0164554458",
  "membership": false
},
{
  "_id": customerIDs[6],
  "name": "William Hii",
  "email": "william@gmail.com",
  "contact": "0178565684",
  "membership": true
```

```
    },
    {
      "_id": customerIDs[7],
      "name": "David Hooi",
      "email": "david@gmail.com",
      "contact": "017565956",
      "membership": false
    },
    {
      "_id": customerIDs[8],
      "name": "Muthyramy",
      "email": "muthu@gmail.com",
      "contact": "0136985623",
      "membership": true
    },
    {
      "_id": customerIDs[9],
      "name": "Mohd Fariz",
      "email": "farez@gmail.com",
      "contact": "0158765484",
      "membership": true
    }
  ]
);
// Cinema
db.cinema.insertMany(
  [
    {
      "_id": cinemaIDs[0],
      "name": "SSC Cyberjaya",
      "address": {
        "street": "Jalan Teknokrat 1",
        "city": "Cyberjaya",
        "state": "Selangor"
      },
      "halls": ["1", "2", "3", "4", "5", "6", "7", "8", "9", "10"]
    },
    {
      "_id": cinemaIDs[1],
      "name": "SSC Puchong",
      "address": {
```

```
      "street": "Jalan Besar 5",
      "city": "Puchong",
      "state": "Selangor"
    },
    "halls": ["1", "2", "3", "4", "5", "6", "7", "8"]
  },
  {
    "_id": cinemaIDs[2],
    "name": "Malacca City",
    "address": {
      "street": "Jalan Hang Tuah 2",
      "city": "Melaka City",
      "state": "Melaka"
    },
    "halls": ["1", "2", "3", "4", "5", "6"]
  },
  {
    "_id": cinemaIDs[3],
    "name": "SSC Georgetown",
    "address": {
      "street": "Jalan Swenson 3",
      "city": "Georgetown",
      "state": "Penang"
    },
    "halls": ["1", "2", "3", "4", "5", "6"]
  },
  {
    "_id": cinemaIDs[4],
    "name": "SSC Johor",
    "address": {
      "street": "Jalan Ismail 4",
      "city": "Johor Bahru",
      "state": "Johor"
    },
    "halls": ["1", "2", "3", "4", "5", "6", "7", "8"]
  },
  {
    "_id": cinemaIDs[5],
    "name": "SSC Kuala Lumpur",
    "address": {
      "street": "Jalan Lumpur 6/9",
```

```
      "city": "Kuala Lumpur",
      "state": "Kuala Lumpur"
    },
    "halls": ["1", "2", "3", "4", "5", "6", "7", "8", "9", "10"]
  },
  {
    "_id": cinemaIDs[6],
    "name": "SSC Kuantan",
    "address": {
      "street": "Jalan Timur 1/1",
      "city": "Kuantan",
      "state": "Pahang"
    },
    "halls": ["1", "2", "3", "4", "5", "6", "7", "8", "9", "10"]
  },
  {
    "_id": cinemaIDs[7],
    "name": "SSC Ipoh",
    "address": {
      "street": "Jalan Sultan Idris 1",
      "city": "Ipoh",
      "state": "Perak"
    },
    "halls": ["1", "2", "3", "4", "5", "6"]
  },
  {
    "_id": cinemaIDs[8],
    "name": "SSC Seremban",
    "address": {
      "street": "Jalan Timur 1/1",
      "city": "Kuantan",
      "state": "Pahang"
    },
    "halls": ["1", "2", "3", "4"]
  },
  {
    "_id": cinemaIDs[9],
    "name": "SSC Ipoh",
    "address": {
      "street": "Jalan Melati 7",
      "city": "Seremban",
```

```
        "state": "Negeri Sembilan"
      },
      "halls": ["1", "2", "3", "4", "5", "6", "7", "8"]
    }
  ]
);
// Movie
db.movie.insertMany(
  [
    {
      "_id": movieIDs[0],
      "name": "The Super Mario Bros Movie",
      "release_date": new Date("2023-04-20"),
      "running_minutes": 93,
      "genre": "Animation",
      "language": "English",
      "classification": "P12",
      "categories": [
        {
          "name": "2D",
          "price": 16.50
        }
      ]
    },
    {
      "_id": movieIDs[1],
      "name": "Guardian of the Galaxy",
      "release_date": new Date("2023-05-05"),
      "running_minutes": 150,
      "genre": "Action",
      "language": "English",
      "classification": "P12",
      "categories": [
        {
          "name": "2D",
          "price": 16.50
        },
        {
          "name": "IMAX",
          "price": 23.50
        }
```

```
    ]
  },
  {
    "_id": movieIDs[2],
    "name": "Ponniyin Selvan Part 2",
    "release_date": new Date("2023-04-28"),
    "running_minutes": 165,
    "genre": "Historical",
    "language": "Tamil",
    "classification": "P12",
    "categories": [
      {
        "name": "2D",
        "price": 16.50
      },
      {
        "name": "IMAX",
        "price": 23.50
      }
    ]
  },
  {
    "_id": movieIDs[3],
    "name": "VII XII",
    "release_date": new Date("2023-05-04"),
    "running_minutes": 86,
    "genre": "Action",
    "language": "Malay",
    "classification": "13",
    "categories": [
      {
        "name": "2D",
        "price": 16.50
      },
      {
        "name": "IMAX",
        "price": 23.50
      }
    ]
  },
  {
```

```
    "_id": movieIDs[4],
    "name": "Flashover",
    "release_date": new Date("2023-04-28"),
    "running_minutes": 75,
    "genre": "Action",
    "language": "Mandarin",
    "classification": "P12",
    "categories": [
      {
        "name": "2D",
        "price": 16.50
      },
      {
        "name": "IMAX",
        "price": 23.50
      }
    ]
  },
  {
    "_id": movieIDs[5],
    "name": "The Ghost Station",
    "release_date": new Date("2023-05-04"),
    "running_minutes": 71,
    "genre": "Horror",
    "language": "Korean",
    "classification": "13",
    "categories": [
      {
        "name": "2D",
        "price": 16.50
      },
      {
        "name": "IMAX",
        "price": 23.50
      }
    ]
  },
  {
    "_id": movieIDs[6],
    "name": "Beautiful Disaster",
    "release_date": new Date("2023-05-04"),
```

```
    "running_minutes": 95,
    "genre": "Romance",
    "language": "English",
    "classification": "18",
    "categories": [
      {
        "name": "2D",
        "price": 16.50
      },
      {
        "name": "IMAX",
        "price": 23.50
      }
    ]
  },
  {
    "_id": movieIDs[7],
    "name": "John Wick 4",
    "release_date": new Date("2023-03-23"),
    "running_minutes": 170,
    "genre": "Action",
    "language": "English",
    "classification": "18",
    "categories": [
      {
        "name": "2D",
        "price": 16.50
      },
      {
        "name": "IMAX",
        "price": 23.50
      }
    ]
  },
  {
    "_id": movieIDs[8],
    "name": "Suzume",
    "release_date": new Date("2023-03-09"),
    "running_minutes": 120,
    "genre": "Animation",
    "language": "Japanese",
```

```
      "classification": "P12",
      "categories": [
        {
          "name": "2D",
          "price": 16.50
        }
      ]
    },
    {
      "_id": movieIDs[9],
      "name": "The First Slam Dunk",
      "release_date": new Date("2023-02-23"),
      "running_minutes": 125,
      "genre": "Animation",
      "language": "Japanese",
      "classification": "13",
      "categories": [
        {
          "name": "2D",
          "price": 16.50
        }
      ]
    }
  ]
);
// Promotion
db.promotion.insertMany(
  [
    {
      "_id": promotionIDs[0],
      "code": "",
      "description": "No promotion",
      "discount": 0.00,
      "promo_start": new Date("2000-01-01"),
      "promo_end": new Date("2999-12-31")
    },
    {
      "_id": promotionIDs[1],
      "code": "NEWSSC",
      "description": "Newcomer discount",
      "discount": 10.00,
```

```
    "promo_start": new Date("2023-05-11"),
    "promo_end": new Date("2023-06-11")
  },
  {
    "_id": promotionIDs[2],
    "code": "MAYBANK5",
    "description": "Maybank Promo",
    "discount": 5.00,
    "promo_start": new Date("2023-05-20"),
    "promo_end": new Date("2023-05-31")
  },
  {
    "_id": promotionIDs[3],
    "code": "MAY",
    "description": "Super Saver May",
    "discount": 5.50,
    "promo_start": new Date("2023-05-01"),
    "promo_end": new Date("2023-05-31")
  },
  {
    "_id": promotionIDs[4],
    "code": "ALLIANZ",
    "description": "AllianzBank Promo",
    "discount": 8.00,
    "promo_start": new Date("2023-05-25"),
    "promo_end": new Date("2023-06-05")
  },
  {
    "_id": promotionIDs[5],
    "code": "STUDENT",
    "description": "Student Price Promo",
    "discount": 7.50,
    "promo_start": new Date("2023-05-01"),
    "promo_end": new Date("2023-07-31")
  },
  {
    "_id": promotionIDs[6],
    "code": "JUNE6",
    "description": "June Fiesta",
    "discount": 6.60,
    "promo_start": new Date("2023-06-01"),
```

```
      "promo_end": new Date("2023-06-30")
    },
    {
      "_id": promotionIDs[7],
      "code": "SSCANNIVERSARY",
      "description": "SSC Anniversary",
      "discount": 10.00,
      "promo_start": new Date("2023-06-20"),
      "promo_end": new Date("2023-06-25")
    },
    {
      "_id": promotionIDs[8],
      "code": "ALLYEAR",
      "description": "All year discount",
      "discount": 5.00,
      "promo_start": new Date("2023-01-01"),
      "promo_end": new Date("2023-12-31")
    },
    {
      "_id": promotionIDs[9],
      "code": "HOLIDAY",
      "description": "Holiday Discount",
      "discount": 11.50,
      "promo_start": new Date("2023-12-01"),
      "promo_end": new Date("2023-12-15")
    }
  ]
);
// Screening
db.screening.insertMany(
  [
    {
      "_id": screeningIDs[0],
      "movie": movieIDs[9],
      "category": "2D",
      "cinema": cinemaIDs[3],
      "hall": "9",
      "time": new Date("2023-05-10T10:30:00")
    },
    {
      "_id": screeningIDs[1],
```

```
    "movie": movieIDs[0],
    "category": "2D",
    "cinema": cinemaIDs[4],
    "hall": "6",
    "time": new Date("2023-05-10T11:45:00")
  },
  {
    "_id": screeningIDs[2],
    "movie": movieIDs[8],
    "category": "2D",
    "cinema": cinemaIDs[2],
    "hall": "3",
    "time": new Date("2023-05-10T13:15:00")
  },
  {
    "_id": screeningIDs[3],
    "movie": movieIDs[1],
    "category": "2D",
    "cinema": cinemaIDs[4],
    "hall": "5",
    "time": new Date("2023-05-10T15:30:00")
  },
  {
    "_id": screeningIDs[4],
    "movie": movieIDs[7],
    "category": "IMAX",
    "cinema": cinemaIDs[1],
    "hall": "1",
    "time": new Date("2023-05-10T18:10:00")
  },
  {
    "_id": screeningIDs[5],
    "movie": movieIDs[2],
    "category": "2D",
    "cinema": cinemaIDs[2],
    "hall": "1",
    "time": new Date("2023-05-11T11:45:00")
  },
  {
    "_id": screeningIDs[6],
    "movie": movieIDs[6],
```

```
      "category": "IMAX",
      "cinema": cinemaIDs[3],
      "hall": "4",
      "time": new Date("2023-05-11T14:20:00")
    },
    {
      "_id": screeningIDs[7],
      "movie": movieIDs[3],
      "category": "2D",
      "cinema": cinemaIDs[1],
      "hall": "8",
      "time": new Date("2023-05-11T16:30:00")
    },
    {
      "_id": screeningIDs[8],
      "movie": movieIDs[5],
      "category": "IMAX",
      "cinema": cinemaIDs[4],
      "hall": "7",
      "time": new Date("2023-05-11T19:35:00")
    },
    {
      "_id": screeningIDs[9],
      "movie": movieIDs[4],
      "category": "2D",
      "cinema": cinemaIDs[3],
      "hall": "9",
      "time": new Date("2023-05-11T22:00:00")
    }
  ]
);
// Ticket
db.ticket.insertMany(
  [
    {
      "_id": ticketIDs[0],
      "customer": customerIDs[0],
      "screening": screeningIDs[6],
      "promotion": promotionIDs[0],
      "seats": ["3D", "3E"],
      "nett_price": 47.00
```

```
  },
  {
    "_id": ticketIDs[1],
    "customer": customerIDs[6],
    "screening": screeningIDs[3],
    "promotion": promotionIDs[1],
    "seats": ["2F"],
    "nett_price": 6.50
  },
  {
    "_id": ticketIDs[2],
    "customer": customerIDs[2],
    "screening": screeningIDs[4],
    "promotion": promotionIDs[7],
    "seats": ["5G", "5H", "5I"],
    "nett_price": 60.50
  },
  {
    "_id": ticketIDs[3],
    "customer": customerIDs[1],
    "screening": screeningIDs[5],
    "promotion": promotionIDs[0],
    "seats": ["2A"],
    "nett_price": 16.50
  },
  {
    "_id": ticketIDs[4],
    "customer": customerIDs[8],
    "screening": screeningIDs[1],
    "promotion": promotionIDs[0],
    "seats": ["6C", "6D"],
    "nett_price": 33.00
  },
  {
    "_id": ticketIDs[5],
    "customer": customerIDs[3],
    "screening": screeningIDs[6],
    "promotion": promotionIDs[2],
    "seats": ["4F", "4G", "4H", "4I", "4J", "4K"],
    "nett_price": 136.00
  },
```

```
    {
      "_id": ticketIDs[6],
      "customer": customerIDs[6],
      "screening": screeningIDs[9],
      "promotion": promotionIDs[0],
      "seats": ["1H"],
      "nett_price": 16.50
    },
    {
      "_id": ticketIDs[7],
      "customer": customerIDs[4],
      "screening": screeningIDs[7],
      "promotion": promotionIDs[4],
      "seats": ["3A", "3B"],
      "nett_price": 25.00
    },
    {
      "_id": ticketIDs[8],
      "customer": customerIDs[9],
      "screening": screeningIDs[4],
      "promotion": promotionIDs[0],
      "seats": ["7G", "7H", "7I", "7J"],
      "nett_price": 94.00
    },
    {
      "_id": ticketIDs[9],
      "customer": customerIDs[8],
      "screening": screeningIDs[3],
      "promotion": promotionIDs[3],
      "seats": ["5F"],
      "nett_price": 11.00
    }
  ]
);
```

Figure 4.1: Data Insertion code for the Movie Ticketing System Database

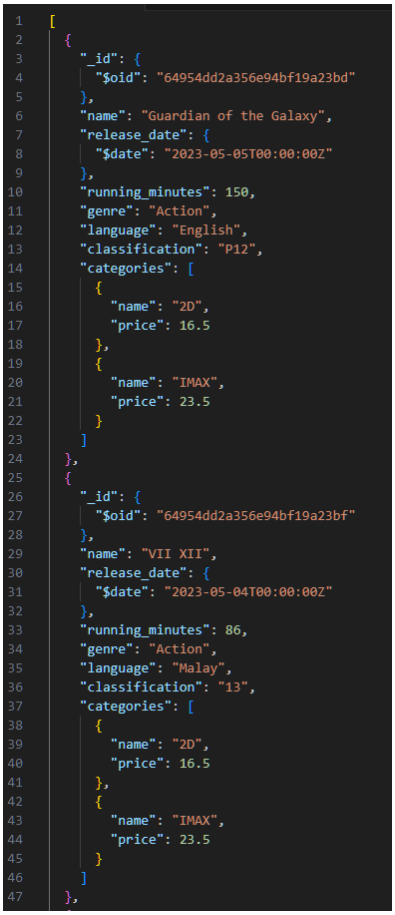
5. NoSQL

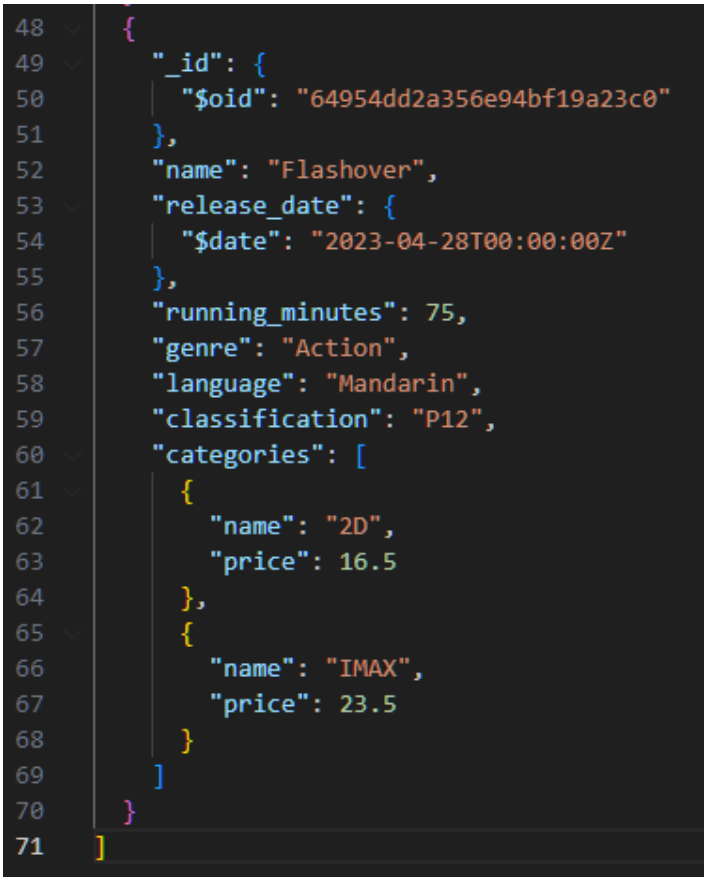
i) Logical Operation

The command created touches upon movies with genre of “Action” or “Animation”, which does not have a language of “Japanese”, that is released after 25/04/2023

```
// i) Logical Operation
db.movie.find({
  $and: [
      {$or: [ { genre: "Action" }, { genre: "Animation" } ]},
      {language: { $not: { $eq: "Japanese"}}},
      {release_date: { $gt: ISODate("2023-04-25T00:00:00.000Z")}}
  ]
});
```

End Result:





ii) Comparison Operation

Finds promotions that has a discount greater than or equal to 10, and is between 01/05/2023 and 31/06/2023

```
// ii) Comparison Operation
db.promotion.find({
  $and: [
      {discount: {$gte: 10}},
      {promo_start: {$gt: ISODate('2023-05-01T00:00:00Z')}},
      {promo_end: {$lte: ISODate('2023-06-31T00:00:00Z')}},
  ]
});
```

End Result:


```
[
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23c7"
    },
    "code": "NEWSSC",
    "description": "Newcomer discount",
    "discount": 10,
    "promo_start": {
      "$date": "2023-05-11T00:00:00Z"
    },
    "promo_end": {
      "$date": "2023-06-11T00:00:00Z"
    }
  },
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23cd"
    },
    "code": "SSCANNIVERSARY",
    "description": "SSC Anniversary",
    "discount": 10,
    "promo_start": {
      "$date": "2023-06-20T00:00:00Z"
    },
    "promo_end": {
      "$date": "2023-06-25T00:00:00Z"
    }
  }
]
```

iii) Aggregate Operation

For each ticket, connected to the customer using the id, sum up how much each customer spent (that has at least bought a ticket) and their total average spending on tickets.

```
// iii) Aggregate Operation
db.ticket.aggregate(
  [
      {
          $group: {
              _id: "$customer",
              totalTicketSales: { $sum: "$nett_price" },
              averageTicketSales: { $avg: "$nett_price" },
          }
      }
  ]
);
```

End Result:

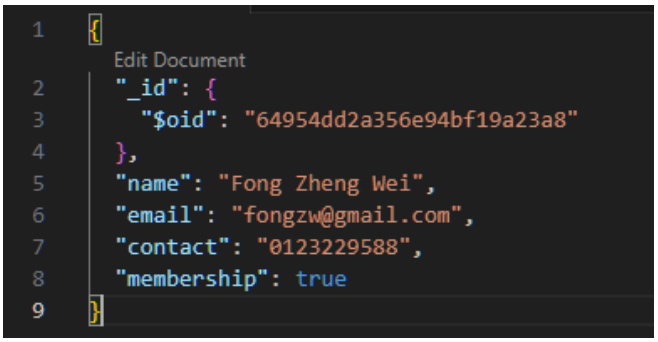

```
[
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23ae"
    },
    "totalTicketSales": 23,
    "averageTicketSales": 11.5
  },
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23b0"
    },
    "totalTicketSales": 44,
    "averageTicketSales": 22
  },
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23b1"
    },
    "totalTicketSales": 94,
    "averageTicketSales": 94
  },
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23ab"
    },
    "totalTicketSales": 136,
    "averageTicketSales": 136
  },
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23ac"
    },
    "totalTicketSales": 25,
    "averageTicketSales": 25
  },
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23a8"
    },
    "totalTicketSales": 47,
    "averageTicketSales": 47
  },
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23aa"
    },
    "totalTicketSales": 60.5,
    "averageTicketSales": 60.5
  },
  {
    "_id": {
      "$oid": "64954dd2a356e94bf19a23a9"
    },
    "totalTicketSales": 16.5,
    "averageTicketSales": 16.5
  }
]
```

6. Update

The utilisation of the following command touches upon the customer with an email of “fongzw@gmail.com” and contact of “0123229588” to new values.

```
db.customer.updateOne(
  {
    email: "fongzw@gmail.com",
    contact: "0123229588"
  },
  {
      $set: {
          name: "Zheng Wei Fong",
          email: "zhengweifong@gmail.com",
          contact: "015484565865",
          membership: true
      }
  }
);
```

Before Result:



After Result:

7. Delete

Deleting every single movie that was released before 31/03/2023. Useful for making a way to show off new movies that may grab more attention of ticket buyers.

```
db.movie.deleteMany(

  {release_date: {$lte: ISODate("2023-03-31T00:00:00.000Z")}}

);
```

Before Result (10 Entries):


```
[
  {
    "$oid": "6495519647faeaa84ee2ce4c"
  },
  {
    "$oid": "6495519647faeaa84ee2ce4d"
  },
  {
    "$oid": "6495519647faeaa84ee2ce4e"
  },
  {
    "$oid": "6495519647faeaa84ee2ce4f"
  },
  {
    "$oid": "6495519647faeaa84ee2ce50"
  },
  {
    "$oid": "6495519647faeaa84ee2ce51"
  },
  {
    "$oid": "6495519647faeaa84ee2ce52"
  },
  {
    "$oid": "6495519647faeaa84ee2ce53"
  },
  {
    "$oid": "6495519647faeaa84ee2ce54"
  },
  {
    "$oid": "6495519647faeaa84ee2ce55"
  }
]
```


After Result(7 entries):


```
[
  {
    "$oid": "6495519647faeaa84ee2ce4c"
  },
  {
    "$oid": "6495519647faeaa84ee2ce4d"
  },
  {
    "$oid": "6495519647faeaa84ee2ce4e"
  },
  {
    "$oid": "6495519647faeaa84ee2ce4f"
  },
  {
    "$oid": "6495519647faeaa84ee2ce50"
  },
  {
    "$oid": "6495519647faeaa84ee2ce51"
  },
  {
    "$oid": "6495519647faeaa84ee2ce52"
  }
]
```

8. Two commands not used in lecture

a) createUser & role

Following command sets the user info for the specific database, alongside read and write permissions to a specific database that is currently being worked on

```
use('admin');
db.createUser(
  {
    user: "ZhengWeiSenpai",
    pwd: "myverysecurepassword",
    roles: [
      { role: "readWrite", db: "movie_assignment2" }
    ]
  }
);
```

b) $lookup

Getting all the cinema info alongside the movie and screening that are possibility available within the cinema using the $lookup option

```
db.cinema.aggregate([
  {
    $lookup: {
      from: "screening",
      localField: "_id",
      foreignField: "cinema",
      as: "onScreen"
    }
  },
  {
    $project: { "onScreen.movie.name": 1, "onScreen.category": 1 }
  }
]);
```

Part B

Short Essay

Big data is data that contains greater variety, arriving in increasing volumes and with more velocity (What Is Big Data?, n.d.). It includes retail, healthcare, financial, etc over the past few years, and social media marketing is no exception. Social media platforms' abundance of user-generated data has given marketers massive opportunities to advertise. This essay will dive into how big data affects social media marketing in various businesses.

First and foremost, the impact of big data touches upon data optimization. Data optimization is the process by which organisations extract, analyse, and store data for maximum efficiency (What is Data Optimization?, n.d.). Social media systems allow for collection and ingestion of data within itself. This includes social media feeds, likes, user interaction, and logs. The collected data enables the marketers to keep up to date with market information with ease by analysing behaviours and trends, and real-time insight extraction.

Another impact on social media marketing is the ability to target customer segmentation. Data analysis on customer behaviour may help marketers to define target audience easily. It’s the best way to identify the most and least profitable customers, and focus marketing efforts on loyal customers who will most likely be interested (Jay Cameron, 2014). The data can include the customer demographics, buying behaviour, geographical areas, and interests. Marketers can use this information to gain insights of their target audiences, which can improve marketing effectiveness and customer satisfaction.

Customer care is also impacted by big data. Marketers create recommendations for customers, by leveraging purchasing history to offer promotion to customers and build up loyalty among customers, through analysis of customer purchase patterns and customer data (Marinina, 2019). This level of personalization enhances the customers’ experience and strengthens the relationship between customers and the marketers’ brand. After-sales service also benefits from this approach.

The final impact is marketing advertisement. Big data enables marketers to perform monitoring and analysing social media content that is relevant for social media marketers and sentiment analysis to gain insights into consumer opinions, brand reputation, and market trends (Garduno, 2022). By monitoring social media conversations, marketers can understand how their brand is perceived, identify emerging trends and quickly address any issues raised by customers.

To sum up, big data has significantly impacted social media marketing. Marketers can get audience insights, engage with their target audience, and provide personalised experiences by utilising the enormous amount of data on social media platforms.

(400 words)

References

What is Big Data? (n.d.). Retrieved from Oracle: https://www.oracle.com/my/big-data/what-is-big-data/

What is Data Optimization? (n.d.). Retrieved from Acceldata: https://www.acceldata.io/article/what-is-data-optimization

Jay Cameron, M. (2014, December 8). *Big Data and Customer Segmentation*. Retrieved from LinkedIn: https://www.linkedin.com/pulse/20141207214309-98881704-big-data-and-customer-segmentation/

Marinina, M. (2019, October 30). *Big Data in Ecommerce Personalization, Explained*. Retrieved from Medium: https://towardsdatascience.com/big-data-in-ecommerce-personalization-explained-580efa0dec50

Garduno, C. (2022, March 15). *How Big Data Is Helping Advertisers Solve Problems*. Retrieved from Forbes: https://www.forbes.com/sites/forbesagencycouncil/2022/03/15/how-big-data-is-helping-advertisers-solve-problems/