**VIT-AP UNIVERSITY, ANDHRA PRADESH**

# Lab Sheet 6: MongoDB Basic commands

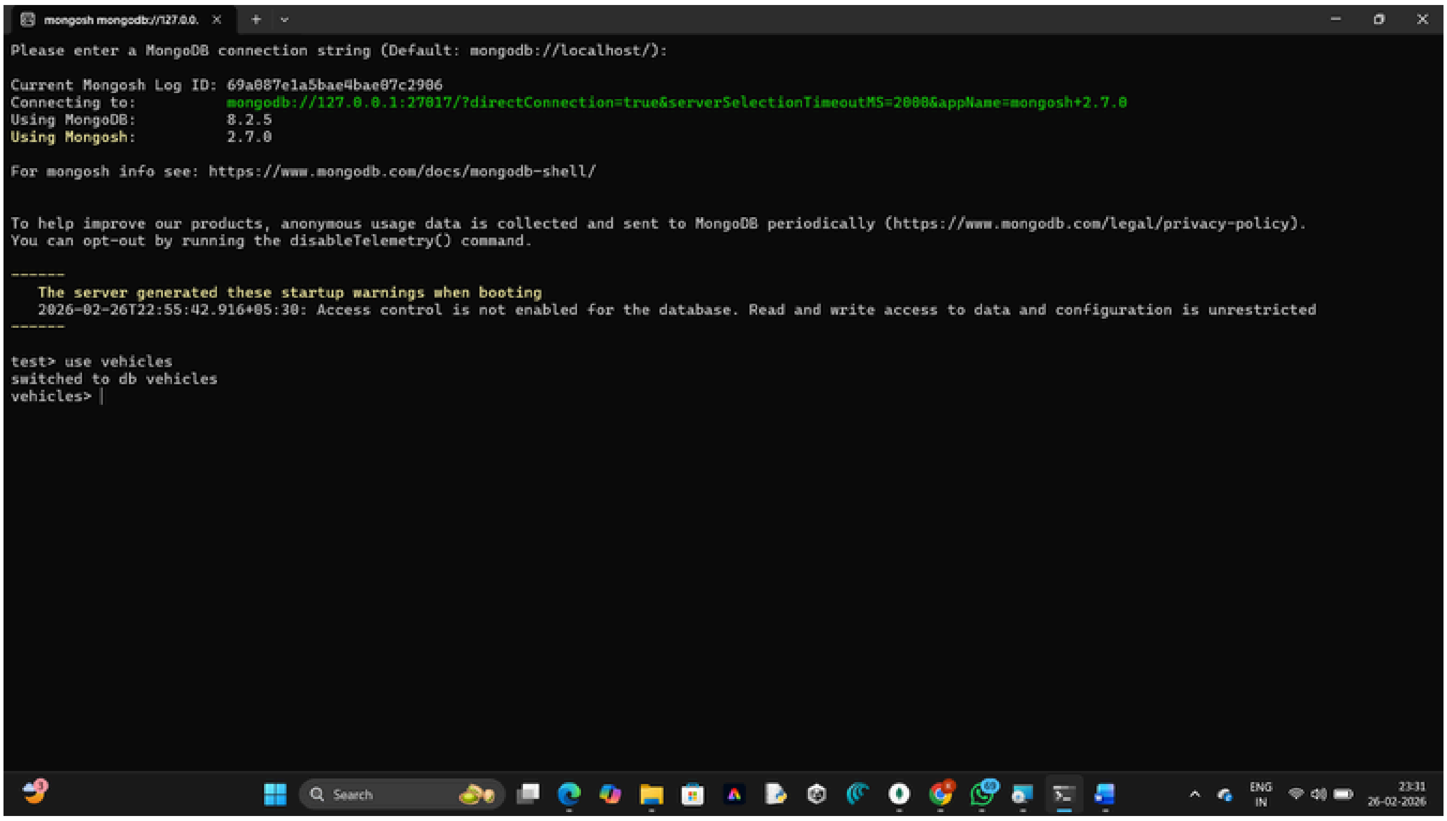
**Branch/ Class:** B.Tech/M.Tech  **Date:** 26-02-2026

**Faculty Name:** Prof. S.Gopikrishnan  **School:** SCOPE

**Student name:** Siri chandana  **Reg. no.:** 23BCE9048

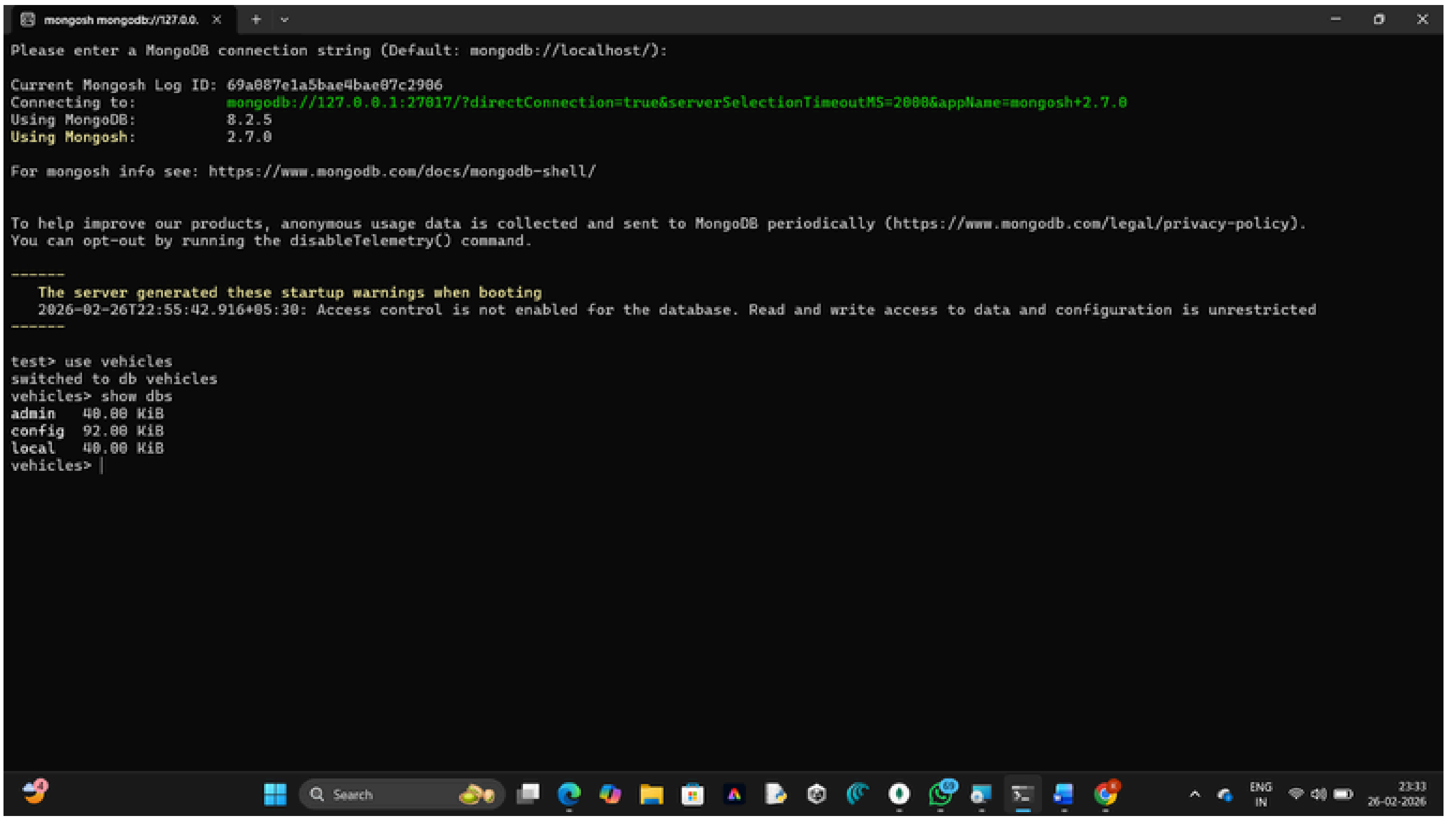
1. Use MongoDB to implement the following DB operations

1. Create a database called 'vehicles' and write a MongoDB query to select database as "vehicles".



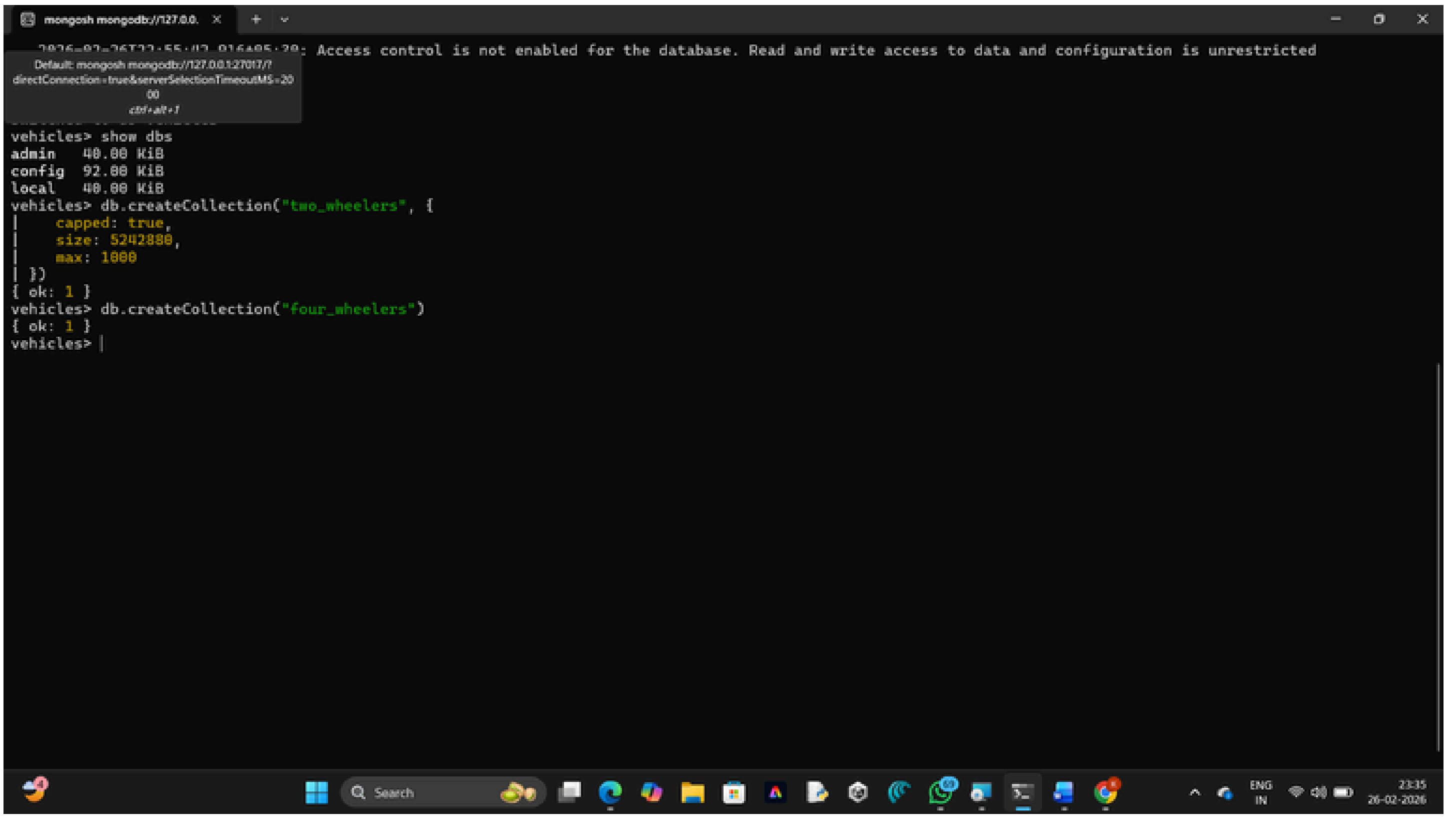
```
Please enter a MongoDB connection string (Default: mongodb://localhost/):

Current Mongosh Log ID: 69a087e1a5bae4bae07c2986
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.7.0
Using MongoDB:          8.2.5
Using Mongosh:          2.7.0

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2026-02-26T22:55:42.916+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

test> use vehicles
switched to db vehicles
vehicles> |
```

2.Write a MongoDB query to display all the databases.



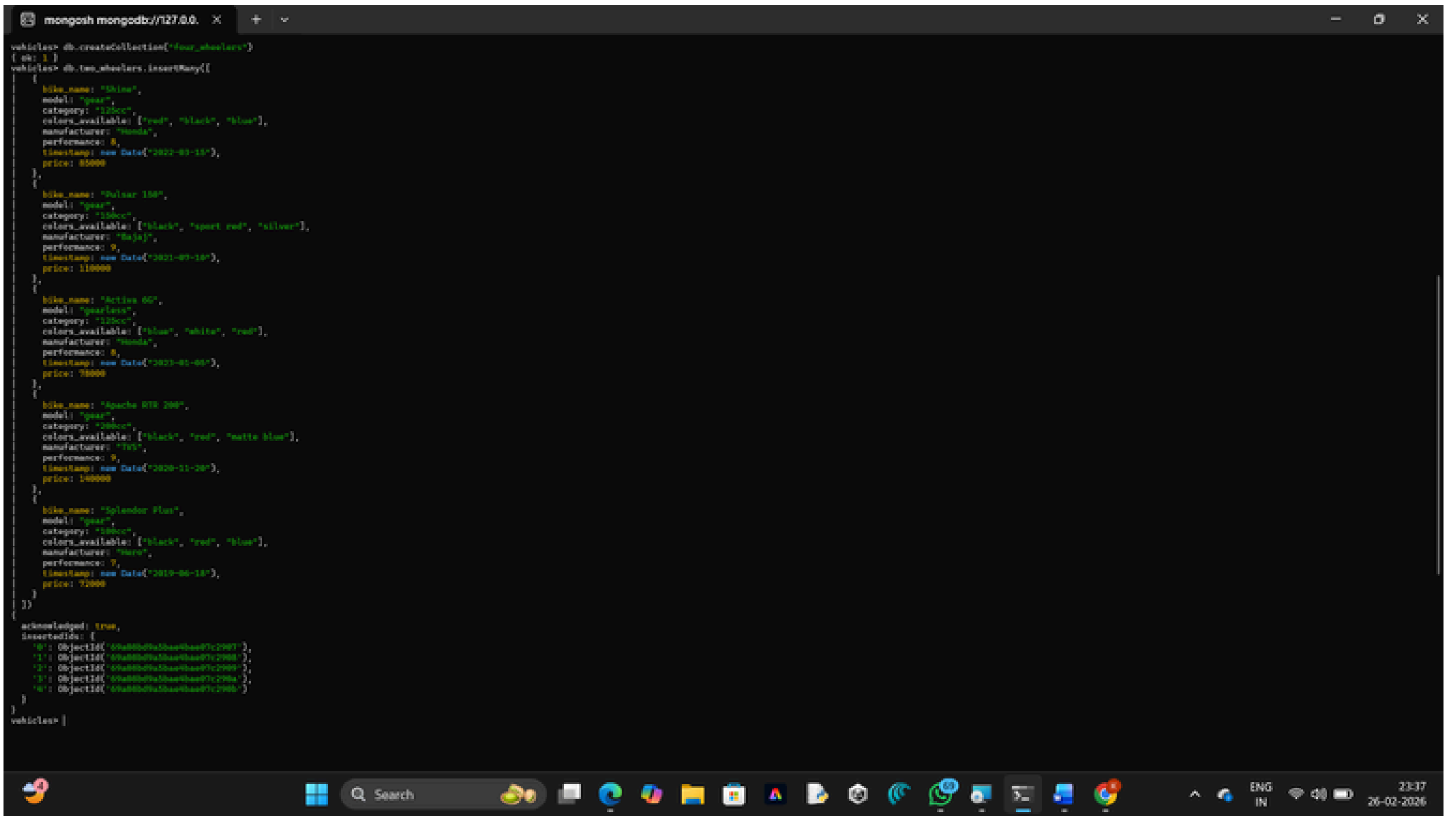
```
Please enter a MongoDB connection string (Default: mongodb://localhost/):

Current Mongosh Log ID: 69a887e1a5bae4bae07c2986
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.7.0
Using MongoDB:          8.2.5
Using Mongosh:          2.7.0

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2026-02-26T22:55:42.916+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

test> use vehicles
switched to db vehicles
vehicles> show dbs
admin   40.00 KiB
config  92.00 KiB
local   40.00 KiB
vehicles> |
```

3.Create a collection called ‘two_wheelers’. (use capping) and Create a collection called ‘four_wheelers’.



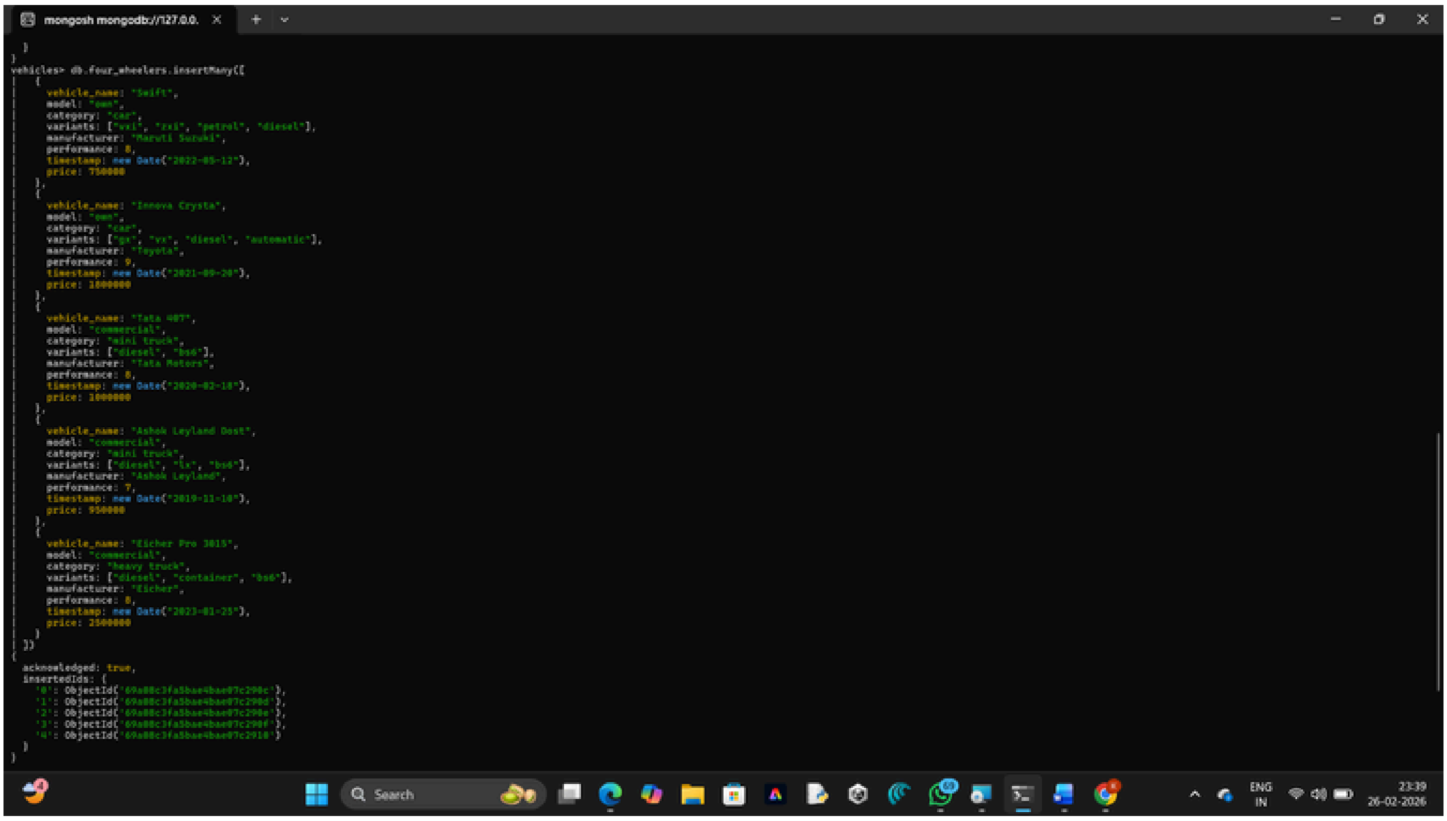
```
   2026-02-26T22:55:42.916+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
vehicles> show dbs
admin   40.00 KiB
config  92.00 KiB
local   40.00 KiB
vehicles> db.createCollection("two_wheelers", {
|     capped: true,
|     size: 5242880,
|     max: 1000
| })
{ ok: 1 }
vehicles> db.createCollection("four_wheelers")
{ ok: 1 }
vehicles> |
```

4. Add 5 two-wheeler details to the collection named ‘two_wheelers’. Each document consists of following fields as bike_name, model (gear or gearless), category (100cc, 125cc, 150cc, 200cc), colors_available (red, black, blue, sport red etc) as array, manufacturer, performance (out of 10), timestamp (date and year release) and price.



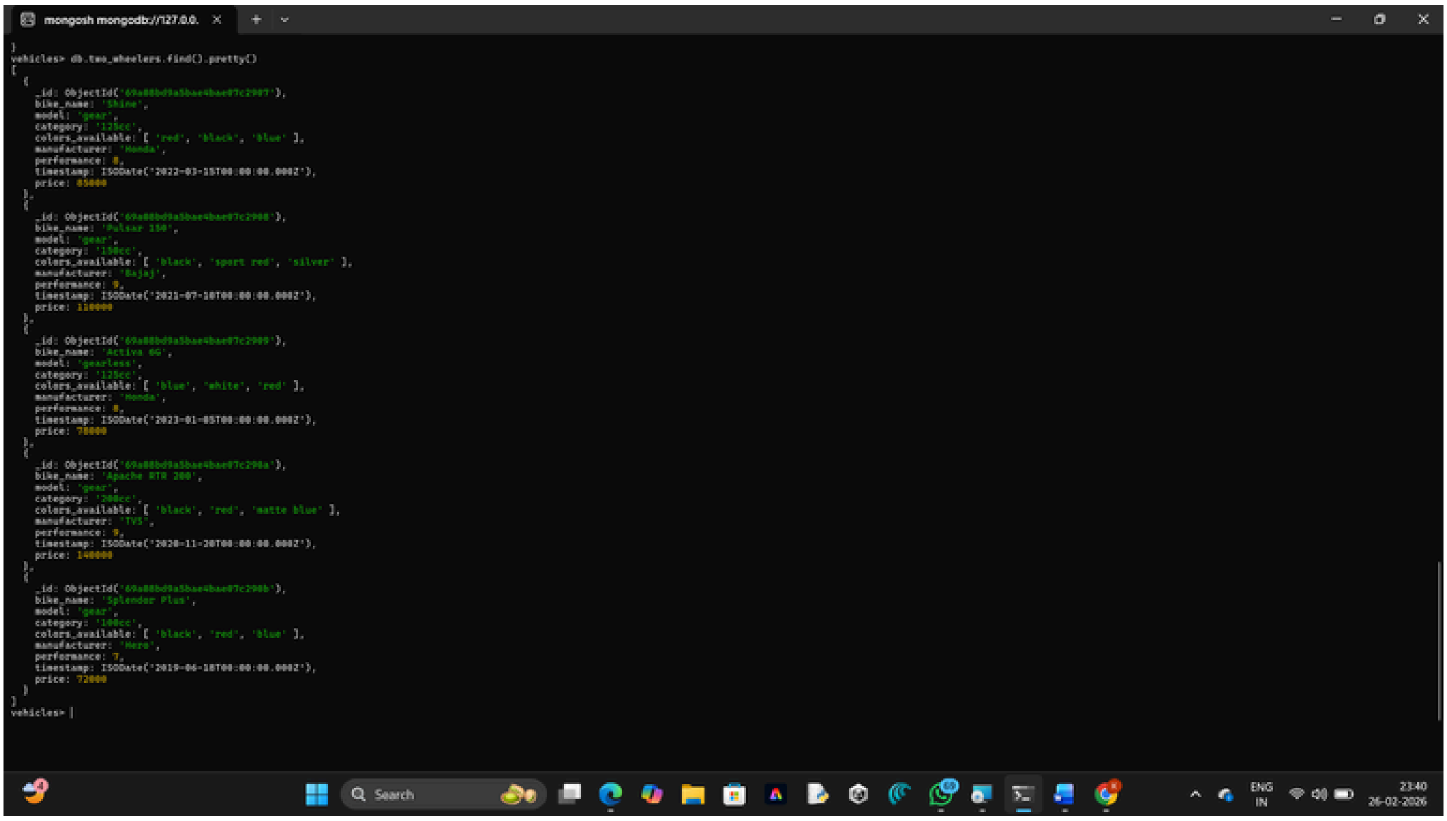
```
vehicles> db.createCollection("four_wheelers")
{ ok: 1 }
vehicles> db.two_wheelers.insertMany([
|     {
|       bike_name: "Shine",
|       model: "gear",
|       category: "125cc",
|       colors_available: ["red", "black", "blue"],
|       manufacturer: "Honda",
|       performance: 8,
|       timestamp: new Date("2022-03-15"),
|       price: 85000
|     },
|     {
|       bike_name: "Pulsar 150",
|       model: "gear",
|       category: "150cc",
|       colors_available: ["black", "sport red", "silver"],
|       manufacturer: "Bajaj",
|       performance: 9,
|       timestamp: new Date("2021-07-10"),
|       price: 110000
|     },
|     {
|       bike_name: "Activa 6G",
|       model: "gearless",
|       category: "125cc",
|       colors_available: ["blue", "white", "red"],
|       manufacturer: "Honda",
|       performance: 8,
|       timestamp: new Date("2023-01-05"),
|       price: 78000
|     },
|     {
|       bike_name: "Apache RTR 200",
|       model: "gear",
|       category: "200cc",
|       colors_available: ["black", "red", "matte blue"],
|       manufacturer: "TVS",
|       performance: 9,
|       timestamp: new Date("2020-11-20"),
|       price: 140000
|     },
|     {
|       bike_name: "Splendor Plus",
|       model: "gear",
|       category: "100cc",
|       colors_available: ["black", "red", "blue"],
|       manufacturer: "Hero",
|       performance: 7,
|       timestamp: new Date("2019-06-18"),
|       price: 72000
|     }
|   ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('69a88bd9a5bae4bae07c2907'),
    '1': ObjectId('69a88bd9a5bae4bae07c2908'),
    '2': ObjectId('69a88bd9a5bae4bae07c2909'),
    '3': ObjectId('69a88bd9a5bae4bae07c290a'),
    '4': ObjectId('69a88bd9a5bae4bae07c290b')
  }
}
vehicles> |
```

5. Add 5 four-wheeler details to the collection named ‘four_wheelers’. Each document consists of following fields as vehicle_name, model (commercial or own), category (car, lorry, bus, mini truck, heavy truck, containers), variants (vxi, zxi, petrol, diesel etc) as array, manufacturer, performance (out of 10), timestamp (date and year release) and price.

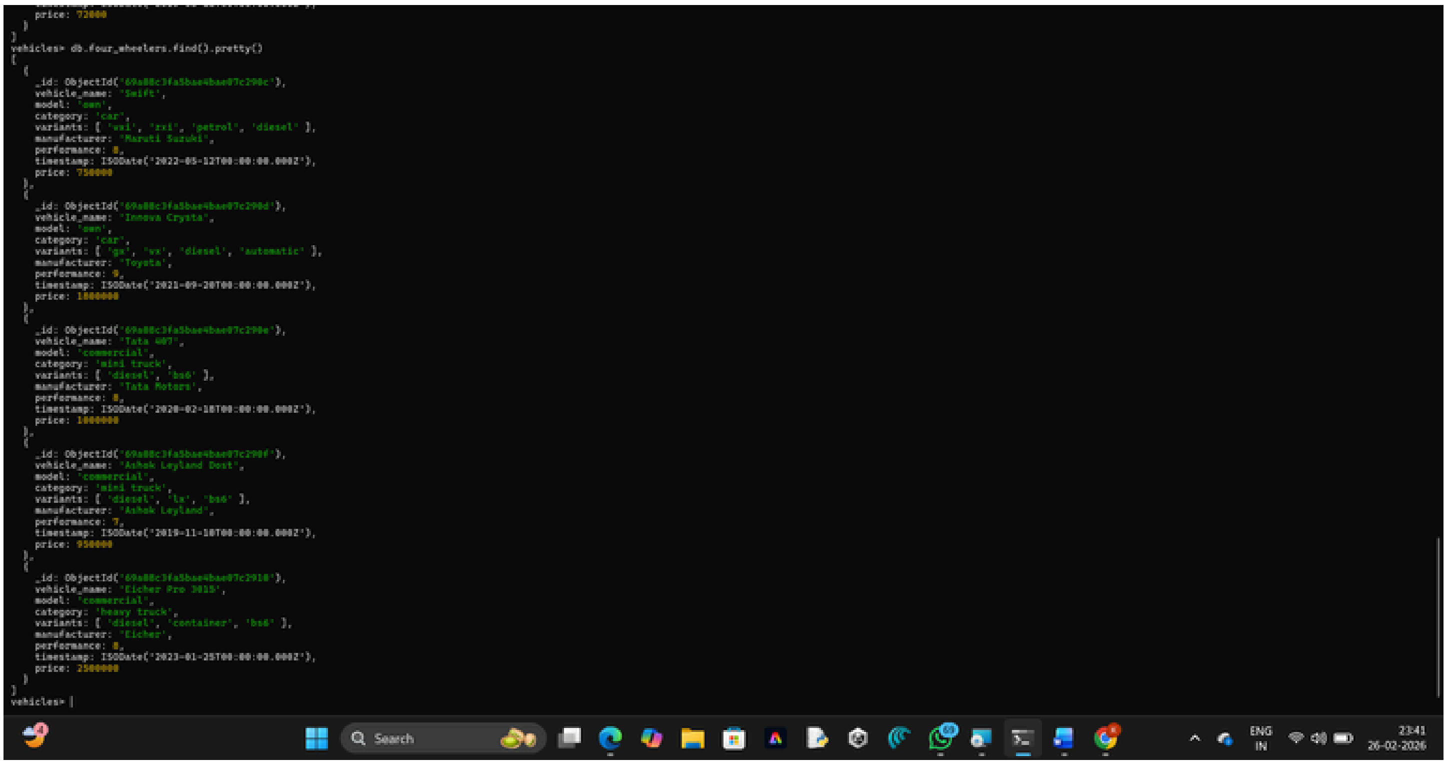


```
  }
}
vehicles> db.four_wheelers.insertMany([
|     {
|       vehicle_name: "Swift",
|       model: "own",
|       category: "car",
|       variants: ["vxi", "zxi", "petrol", "diesel"],
|       manufacturer: "Maruti Suzuki",
|       performance: 8,
|       timestamp: new Date("2022-05-12"),
|       price: 750000
|     },
|     {
|       vehicle_name: "Innova Crysta",
|       model: "own",
|       category: "car",
|       variants: ["gx", "vx", "diesel", "automatic"],
|       manufacturer: "Toyota",
|       performance: 9,
|       timestamp: new Date("2021-09-20"),
|       price: 1800000
|     },
|     {
|       vehicle_name: "Tata 407",
|       model: "commercial",
|       category: "mini truck",
|       variants: ["diesel", "bs6"],
|       manufacturer: "Tata Motors",
|       performance: 8,
|       timestamp: new Date("2020-02-18"),
|       price: 1000000
|     },
|     {
|       vehicle_name: "Ashok Leyland Dost",
|       model: "commercial",
|       category: "mini truck",
|       variants: ["diesel", "lx", "bs6"],
|       manufacturer: "Ashok Leyland",
|       performance: 7,
|       timestamp: new Date("2019-11-10"),
|       price: 950000
|     },
|     {
|       vehicle_name: "Eicher Pro 3015",
|       model: "commercial",
|       category: "heavy truck",
|       variants: ["diesel", "container", "bs6"],
|       manufacturer: "Eicher",
|       performance: 8,
|       timestamp: new Date("2023-01-25"),
|       price: 2500000
|     }
|   ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('69a88c3fa5bae4bae07c290c'),
    '1': ObjectId('69a88c3fa5bae4bae07c290d'),
    '2': ObjectId('69a88c3fa5bae4bae07c290e'),
    '3': ObjectId('69a88c3fa5bae4bae07c290f'),
    '4': ObjectId('69a88c3fa5bae4bae07c2910')
  }
}
```

6. Write a MongoDB query to display all documents available in two_wheelers and four_wheelers.

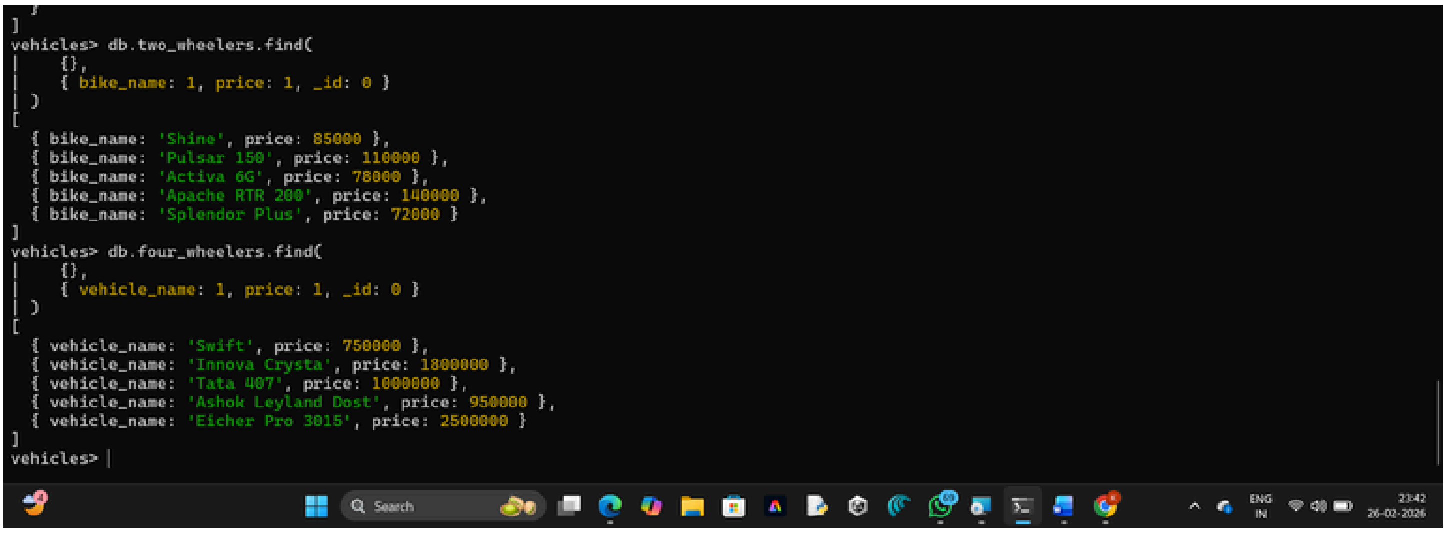


```
}
vehicles> db.two_wheelers.find().pretty()
[
  {
    _id: ObjectId('69a88bd9a5bae4bae07c2907'),
    bike_name: 'Shine',
    model: 'gear',
    category: '125cc',
    colors_available: [ 'red', 'black', 'blue' ],
    manufacturer: 'Honda',
    performance: 8,
    timestamp: ISODate('2022-03-15T00:00:00.000Z'),
    price: 85000
  },
  {
    _id: ObjectId('69a88bd9a5bae4bae07c2908'),
    bike_name: 'Pulsar 150',
    model: 'gear',
    category: '150cc',
    colors_available: [ 'black', 'sport red', 'silver' ],
    manufacturer: 'Bajaj',
    performance: 9,
    timestamp: ISODate('2021-07-10T00:00:00.000Z'),
    price: 110000
  },
  {
    _id: ObjectId('69a88bd9a5bae4bae07c2909'),
    bike_name: 'Activa 6G',
    model: 'gearless',
    category: '125cc',
    colors_available: [ 'blue', 'white', 'red' ],
    manufacturer: 'Honda',
    performance: 8,
    timestamp: ISODate('2023-01-05T00:00:00.000Z'),
    price: 78000
  },
  {
    _id: ObjectId('69a88bd9a5bae4bae07c290a'),
    bike_name: 'Apache RTR 200',
    model: 'gear',
    category: '200cc',
    colors_available: [ 'black', 'red', 'matte blue' ],
    manufacturer: 'TVS',
    performance: 9,
    timestamp: ISODate('2020-11-20T00:00:00.000Z'),
    price: 140000
  },
  {
    _id: ObjectId('69a88bd9a5bae4bae07c290b'),
    bike_name: 'Splendor Plus',
    model: 'gear',
    category: '100cc',
    colors_available: [ 'black', 'red', 'blue' ],
    manufacturer: 'Hero',
    performance: 7,
    timestamp: ISODate('2019-06-18T00:00:00.000Z'),
    price: 72000
  }
]
vehicles> |
```



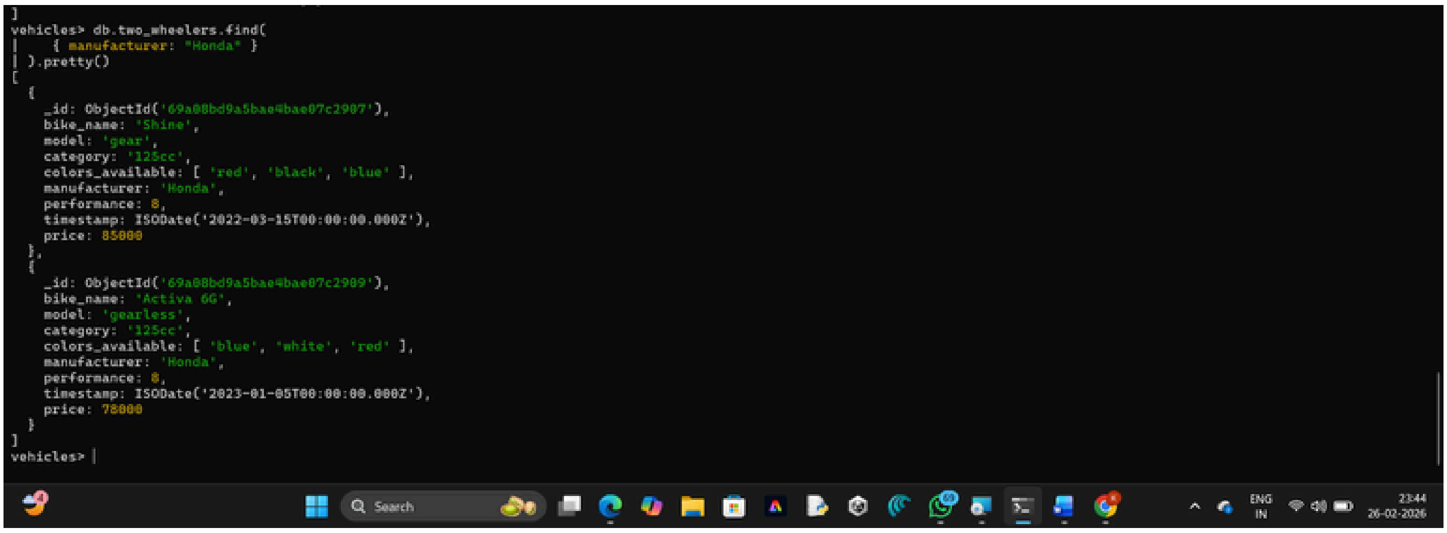
```
    price: 72000
  }
]
vehicles> db.four_wheelers.find().pretty()
[
  {
    _id: ObjectId('69a88c3fa5bae4bae07c290c'),
    vehicle_name: 'Swift',
    model: 'own',
    category: 'car',
    variants: [ 'vxi', 'zxi', 'petrol', 'diesel' ],
    manufacturer: 'Maruti Suzuki',
    performance: 8,
    timestamp: ISODate('2022-05-12T00:00:00.000Z'),
    price: 750000
  },
  {
    _id: ObjectId('69a88c3fa5bae4bae07c290d'),
    vehicle_name: 'Innova Crysta',
    model: 'own',
    category: 'car',
    variants: [ 'gx', 'vx', 'diesel', 'automatic' ],
    manufacturer: 'Toyota',
    performance: 9,
    timestamp: ISODate('2021-09-20T00:00:00.000Z'),
    price: 1800000
  },
  {
    _id: ObjectId('69a88c3fa5bae4bae07c290e'),
    vehicle_name: 'Tata 407',
    model: 'commercial',
    category: 'mini truck',
    variants: [ 'diesel', 'bs6' ],
    manufacturer: 'Tata Motors',
    performance: 8,
    timestamp: ISODate('2020-02-10T00:00:00.000Z'),
    price: 1000000
  },
  {
    _id: ObjectId('69a88c3fa5bae4bae07c290f'),
    vehicle_name: 'Ashok Leyland Dost',
    model: 'commercial',
    category: 'mini truck',
    variants: [ 'diesel', 'lx', 'bs6' ],
    manufacturer: 'Ashok Leyland',
    performance: 7,
    timestamp: ISODate('2019-11-10T00:00:00.000Z'),
    price: 950000
  },
  {
    _id: ObjectId('69a88c3fa5bae4bae07c2910'),
    vehicle_name: 'Eicher Pro 3015',
    model: 'commercial',
    category: 'heavy truck',
    variants: [ 'diesel', 'container', 'bs6' ],
    manufacturer: 'Eicher',
    performance: 8,
    timestamp: ISODate('2023-01-25T00:00:00.000Z'),
    price: 2500000
  }
]
vehicles> |
```

7. Write a MongoDB query to display only vehicle name and price in all the collection of the database



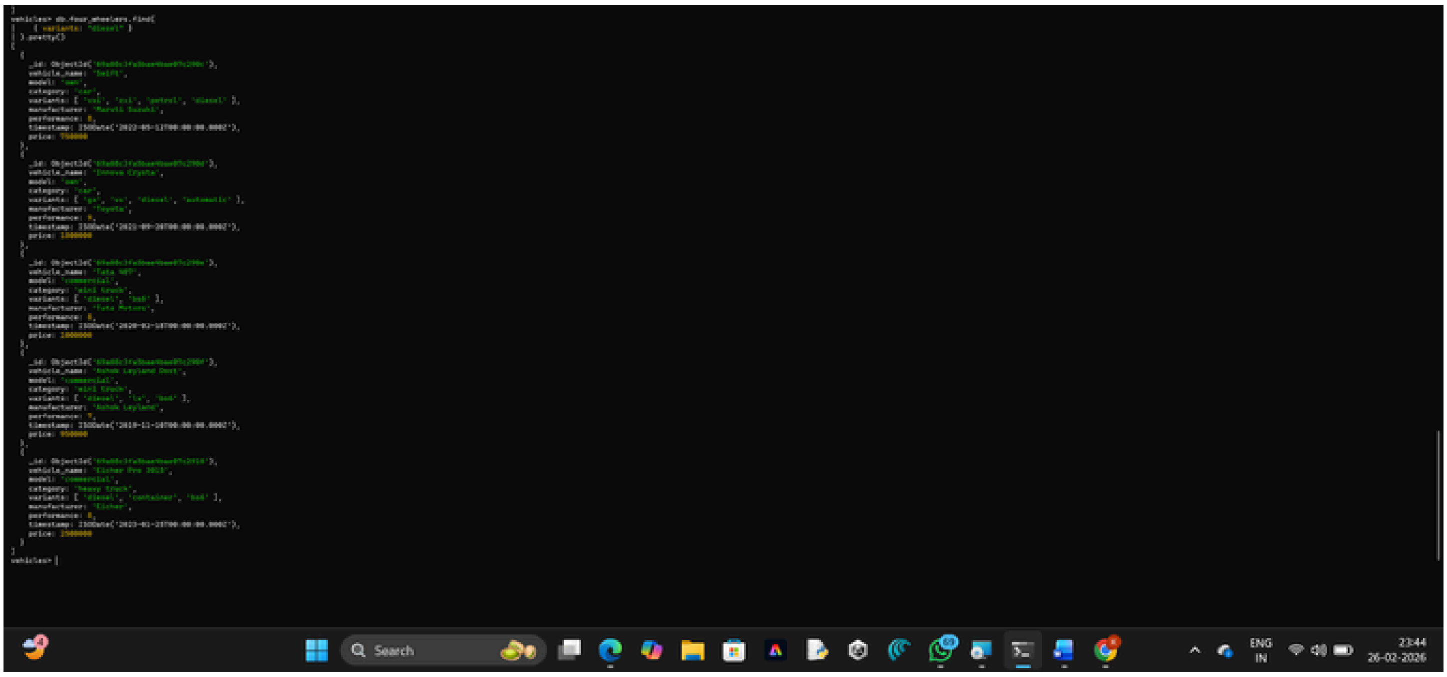
```
]
vehicles> db.two_wheelers.find(
|     {},
|     { bike_name: 1, price: 1, _id: 0 }
| )
[
  { bike_name: 'Shine', price: 85000 },
  { bike_name: 'Pulsar 150', price: 110000 },
  { bike_name: 'Activa 6G', price: 78000 },
  { bike_name: 'Apache RTR 200', price: 140000 },
  { bike_name: 'Splendor Plus', price: 72000 }
]
vehicles> db.four_wheelers.find(
|     {},
|     { vehicle_name: 1, price: 1, _id: 0 }
| )
[
  { vehicle_name: 'Swift', price: 750000 },
  { vehicle_name: 'Innova Crysta', price: 1800000 },
  { vehicle_name: 'Tata 407', price: 1000000 },
  { vehicle_name: 'Ashok Leyland Dost', price: 950000 },
  { vehicle_name: 'Eicher Pro 3015', price: 2500000 }
]
vehicles> |
```

8. Write a MongoDB query to display two_wheelers from a particular company



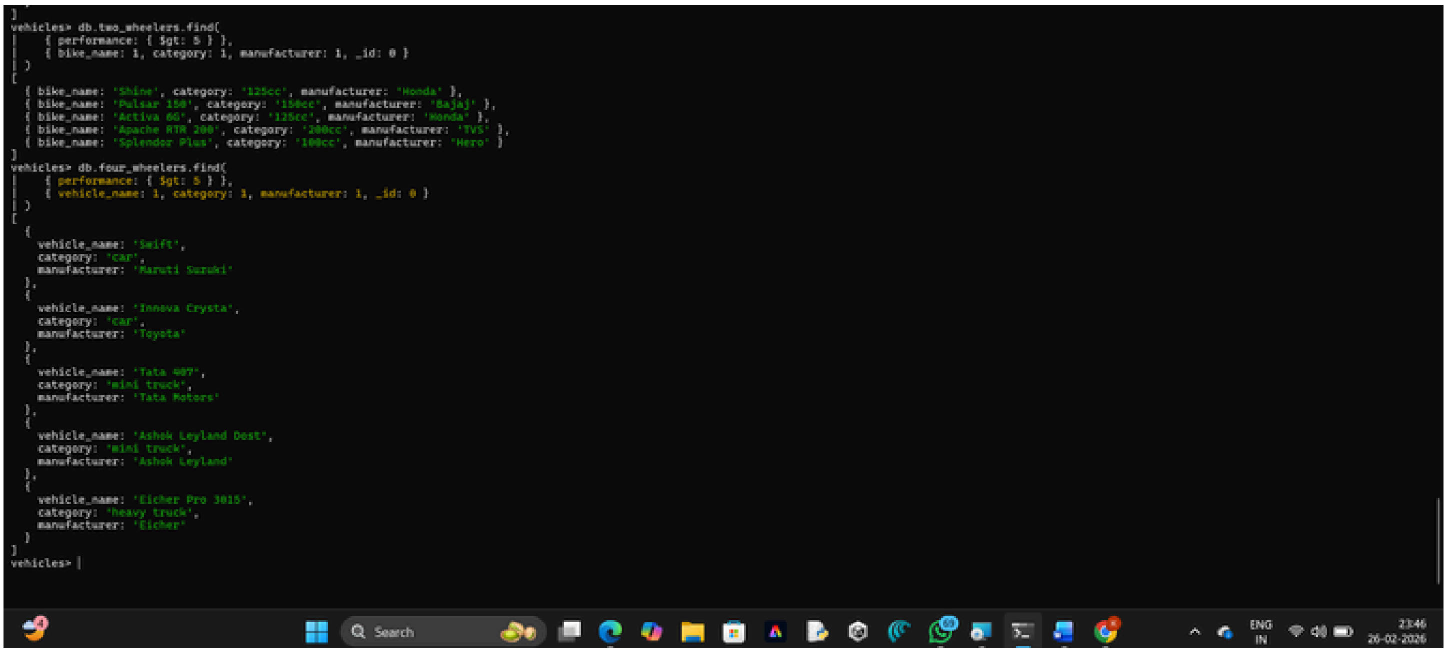
```
]
vehicles> db.two_wheelers.find(
|     { manufacturer: "Honda" }
| ).pretty()
[
  {
    _id: ObjectId('69a08bd9a5bae4bae07c2907'),
    bike_name: 'Shine',
    model: 'gear',
    category: '125cc',
    colors_available: [ 'red', 'black', 'blue' ],
    manufacturer: 'Honda',
    performance: 8,
    timestamp: ISODate('2022-03-15T00:00:00.000Z'),
    price: 85000
  },
  {
    _id: ObjectId('69a08bd9a5bae4bae07c2909'),
    bike_name: 'Activa 6G',
    model: 'gearless',
    category: '125cc',
    colors_available: [ 'blue', 'white', 'red' ],
    manufacturer: 'Honda',
    performance: 8,
    timestamp: ISODate('2023-01-05T00:00:00.000Z'),
    price: 78000
  }
]
vehicles> |
```

9. Write a MongoDB query to display four_wheelers available in diesel variants



```
]
vehicles> db.four_wheelers.find(
|     { variants: "diesel" }
| ).pretty()
[
  {
    _id: ObjectId('[illegible]'),
    vehicle_name: 'Swift',
    model: 'sedan',
    category: 'car',
    variants: [ 'vxi', 'zxi', 'petrol', 'diesel' ],
    manufacturer: 'Maruti Suzuki',
    performance: 8,
    timestamp: ISODate('2022-05-12T00:00:00.000Z'),
    price: 750000
  },
  {
    _id: ObjectId('[illegible]'),
    vehicle_name: 'Innova Crysta',
    model: 'suv',
    category: 'car',
    variants: [ 'gx', 'vx', 'diesel', 'automatic' ],
    manufacturer: 'Toyota',
    performance: 9,
    timestamp: ISODate('2021-09-20T00:00:00.000Z'),
    price: 1800000
  },
  {
    _id: ObjectId('[illegible]'),
    vehicle_name: 'Tata 407',
    model: 'commercial',
    category: 'mini truck',
    variants: [ 'diesel', 'bs6' ],
    manufacturer: 'Tata Motors',
    performance: 8,
    timestamp: ISODate('2020-02-18T00:00:00.000Z'),
    price: 1000000
  },
  {
    _id: ObjectId('[illegible]'),
    vehicle_name: 'Ashok Leyland Dost',
    model: 'commercial',
    category: 'mini truck',
    variants: [ 'diesel', 'ls', 'bs6' ],
    manufacturer: 'Ashok Leyland',
    performance: 7,
    timestamp: ISODate('2019-11-10T00:00:00.000Z'),
    price: 950000
  },
  {
    _id: ObjectId('[illegible]'),
    vehicle_name: 'Eicher Pro 3015',
    model: 'commercial',
    category: 'heavy truck',
    variants: [ 'diesel', 'container', 'bs6' ],
    manufacturer: 'Eicher',
    performance: 8,
    timestamp: ISODate('2023-01-20T00:00:00.000Z'),
    price: 2500000
  }
]
vehicles> |
```

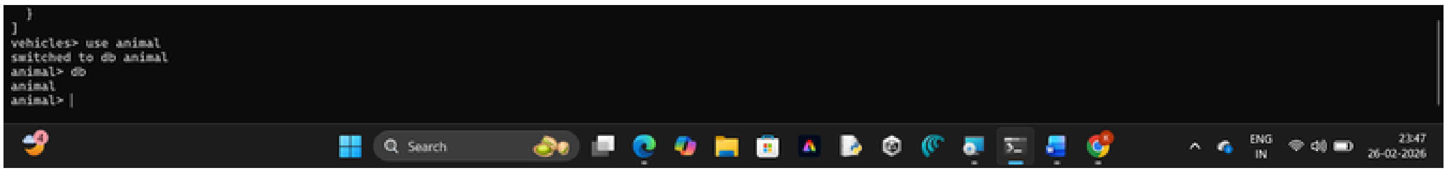
10. Write a MongoDB query to display vehicles name, category and manufacturer details whose rating is more than 5.



```
]
vehicles> db.two_wheelers.find(
|     { performance: { $gt: 5 } },
|     { bike_name: 1, category: 1, manufacturer: 1, _id: 0 }
| )
[
  { bike_name: 'Shine', category: '125cc', manufacturer: 'Honda' },
  { bike_name: 'Pulsar 150', category: '150cc', manufacturer: 'Bajaj' },
  { bike_name: 'Activa 6G', category: '125cc', manufacturer: 'Honda' },
  { bike_name: 'Apache RTR 200', category: '200cc', manufacturer: 'TVS' },
  { bike_name: 'Splendor Plus', category: '100cc', manufacturer: 'Hero' }
]
vehicles> db.four_wheelers.find(
|     { performance: { $gt: 5 } },
|     { vehicle_name: 1, category: 1, manufacturer: 1, _id: 0 }
| )
[
  {
    vehicle_name: 'Swift',
    category: 'car',
    manufacturer: 'Maruti Suzuki'
  },
  {
    vehicle_name: 'Innova Crysta',
    category: 'car',
    manufacturer: 'Toyota'
  },
  {
    vehicle_name: 'Tata 407',
    category: 'mini truck',
    manufacturer: 'Tata Motors'
  },
  {
    vehicle_name: 'Ashok Leyland Dost',
    category: 'mini truck',
    manufacturer: 'Ashok Leyland'
  },
  {
    vehicle_name: 'Eicher Pro 3015',
    category: 'heavy truck',
    manufacturer: 'Eicher'
  }
]
vehicles> |
```

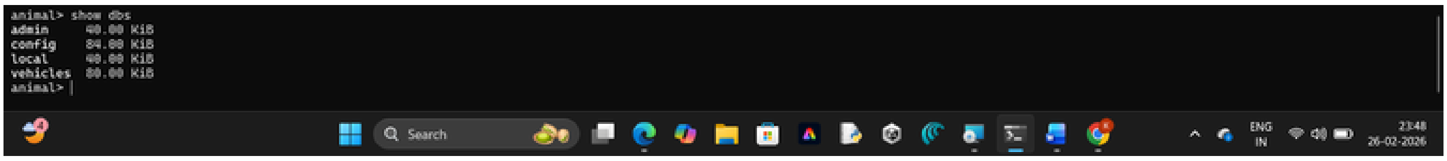
2. Use MongoDB to implement the following DB operations for a Zoo

1. Create a database called ‘animal’ and write a MongoDB query to select database as ‘animal’.



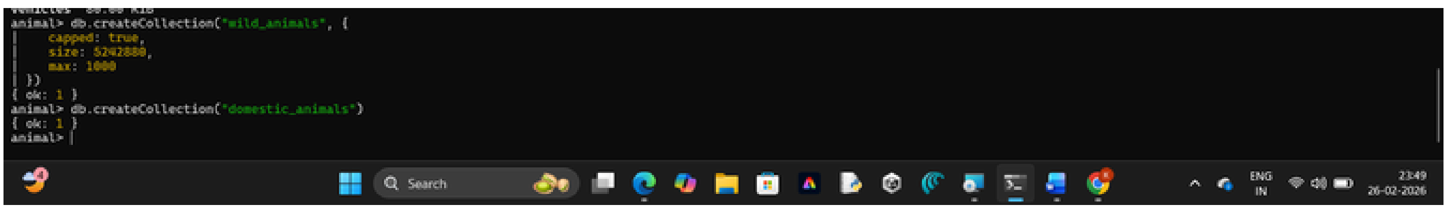
```
  }
]
vehicles> use animal
switched to db animal
animal> db
animal
animal> |
```

2. Write a MongoDB query to display all the databases.



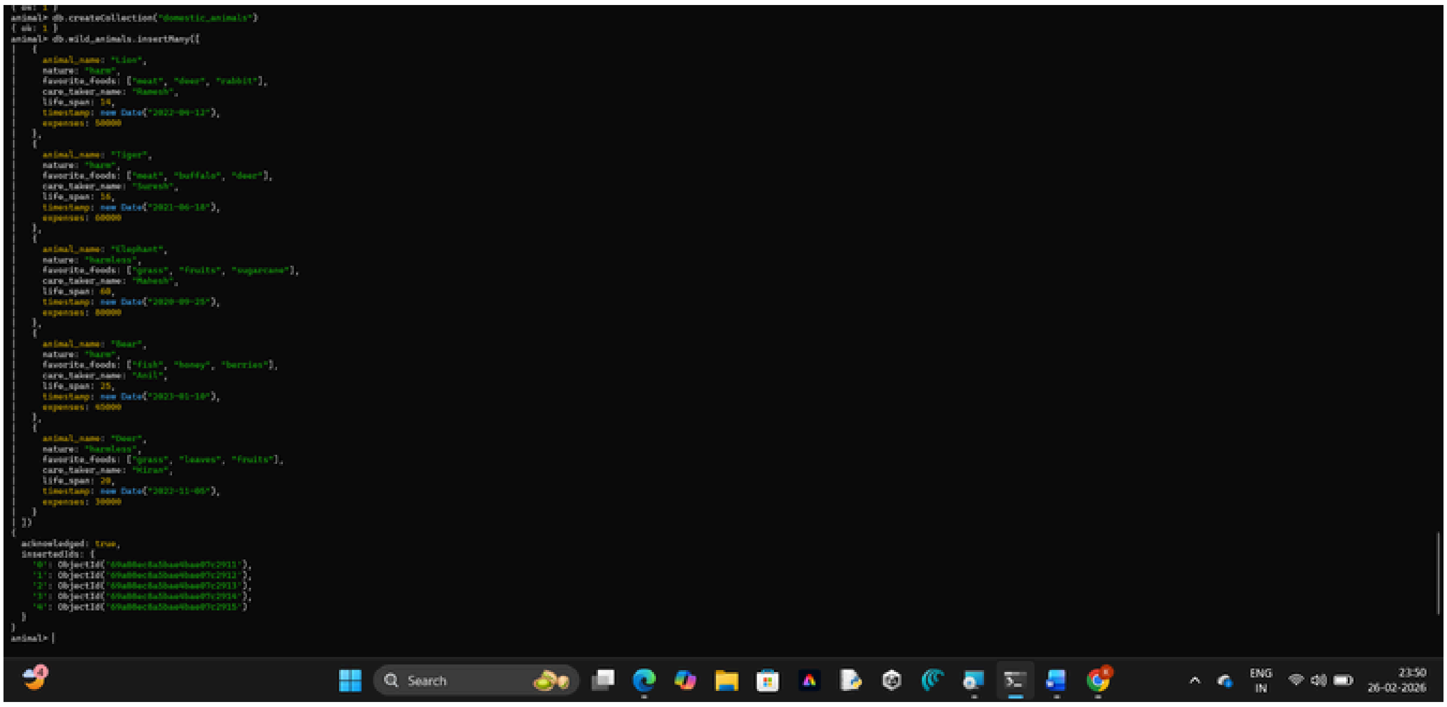
```
animal> show dbs
admin     40.00 KiB
config    84.00 KiB
local     40.00 KiB
vehicles  80.00 KiB
animal> |
```

3. Create a collection called ‘wild_animals’.(use capping) and Create a collection called ‘domestic_animals’.



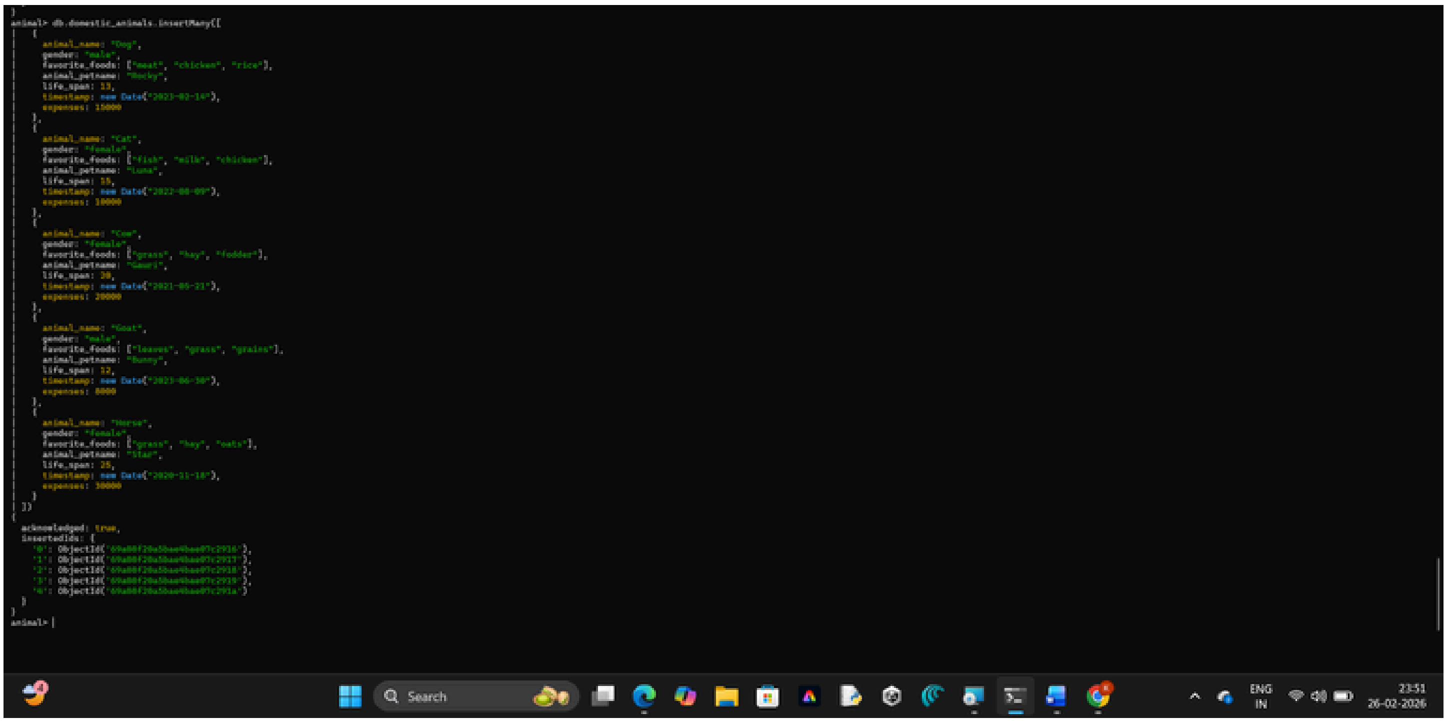
```
vehicles  80.00 KiB
animal> db.createCollection("wild_animals", {
|     capped: true,
|     size: 5242880,
|     max: 1000
| })
{ ok: 1 }
animal> db.createCollection("domestic_animals")
{ ok: 1 }
animal> |
```

4. Add 5 wild_animal details to the collection named ‘wild_animals’. Each document consists of following fields as animal_name, nature (harm or harmless), favorite_foods (meat, rabbits, deer etc) as array, care_taker_name, life span (in years), timestamp (when the animal registered at the Zoo) and expenses.



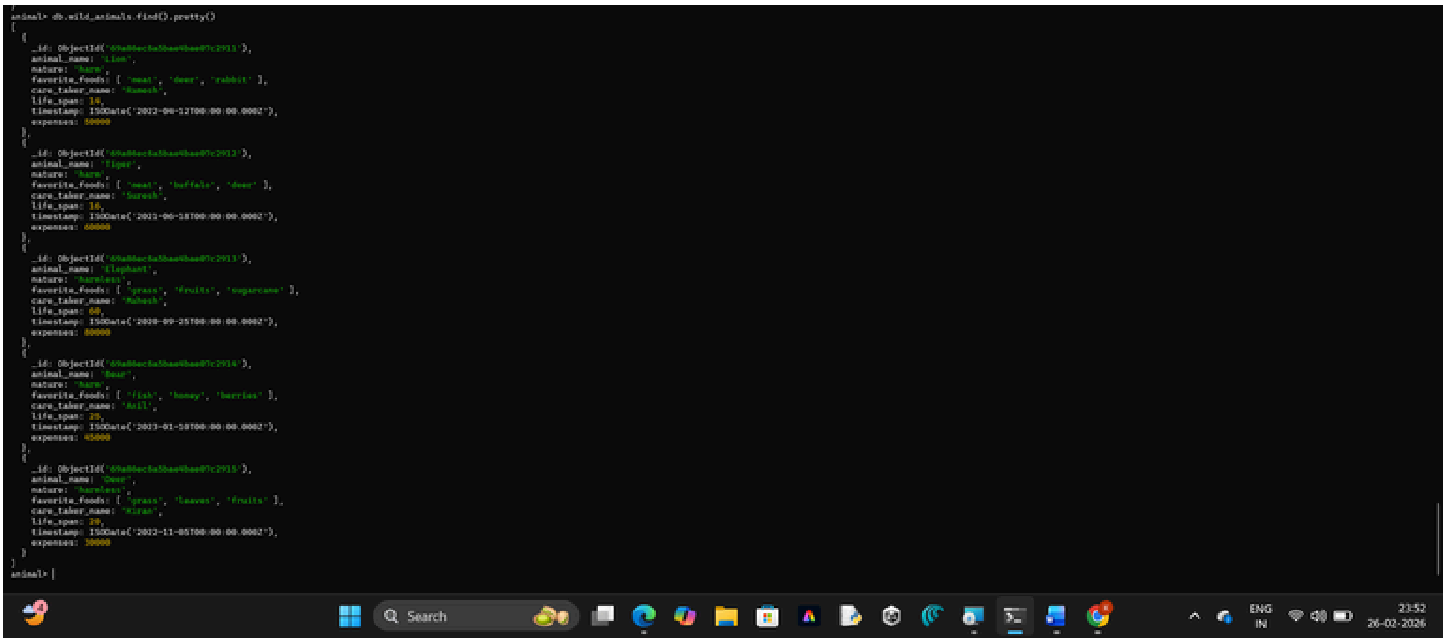
```
{ ok: 1 }
animal> db.createCollection("domestic_animals")
{ ok: 1 }
animal> db.wild_animals.insertMany([
|     {
|       animal_name: "Lion",
|       nature: "harm",
|       favorite_foods: ["meat", "deer", "rabbit"],
|       care_taker_name: "Ramesh",
|       life_span: 14,
|       timestamp: new Date("2022-04-12"),
|       expenses: 50000
|     },
|     {
|       animal_name: "Tiger",
|       nature: "harm",
|       favorite_foods: ["meat", "buffalo", "deer"],
|       care_taker_name: "Suresh",
|       life_span: 16,
|       timestamp: new Date("2021-06-18"),
|       expenses: 60000
|     },
|     {
|       animal_name: "Elephant",
|       nature: "harmless",
|       favorite_foods: ["grass", "fruits", "sugarcane"],
|       care_taker_name: "Mahesh",
|       life_span: 60,
|       timestamp: new Date("2020-09-25"),
|       expenses: 80000
|     },
|     {
|       animal_name: "Bear",
|       nature: "harm",
|       favorite_foods: ["fish", "honey", "berries"],
|       care_taker_name: "Anil",
|       life_span: 25,
|       timestamp: new Date("2023-01-10"),
|       expenses: 45000
|     },
|     {
|       animal_name: "Deer",
|       nature: "harmless",
|       favorite_foods: ["grass", "leaves", "fruits"],
|       care_taker_name: "Kiran",
|       life_span: 20,
|       timestamp: new Date("2022-11-05"),
|       expenses: 30000
|     }
| ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('69a08ec8a5bae4bae07c2911'),
    '1': ObjectId('69a08ec8a5bae4bae07c2912'),
    '2': ObjectId('69a08ec8a5bae4bae07c2913'),
    '3': ObjectId('69a08ec8a5bae4bae07c2914'),
    '4': ObjectId('69a08ec8a5bae4bae07c2915')
  }
}
animal> |
```

5. Add 5 domestic-animal details to the collection named 'domestic_animals'. Each document consists of following fields as animal_name, gender (male or female), favorite_foods (meat, rabbits, deer etc) as array, animal_petname, life span (in years), timestamp (when the animal registered at the Zoo) and expenses.

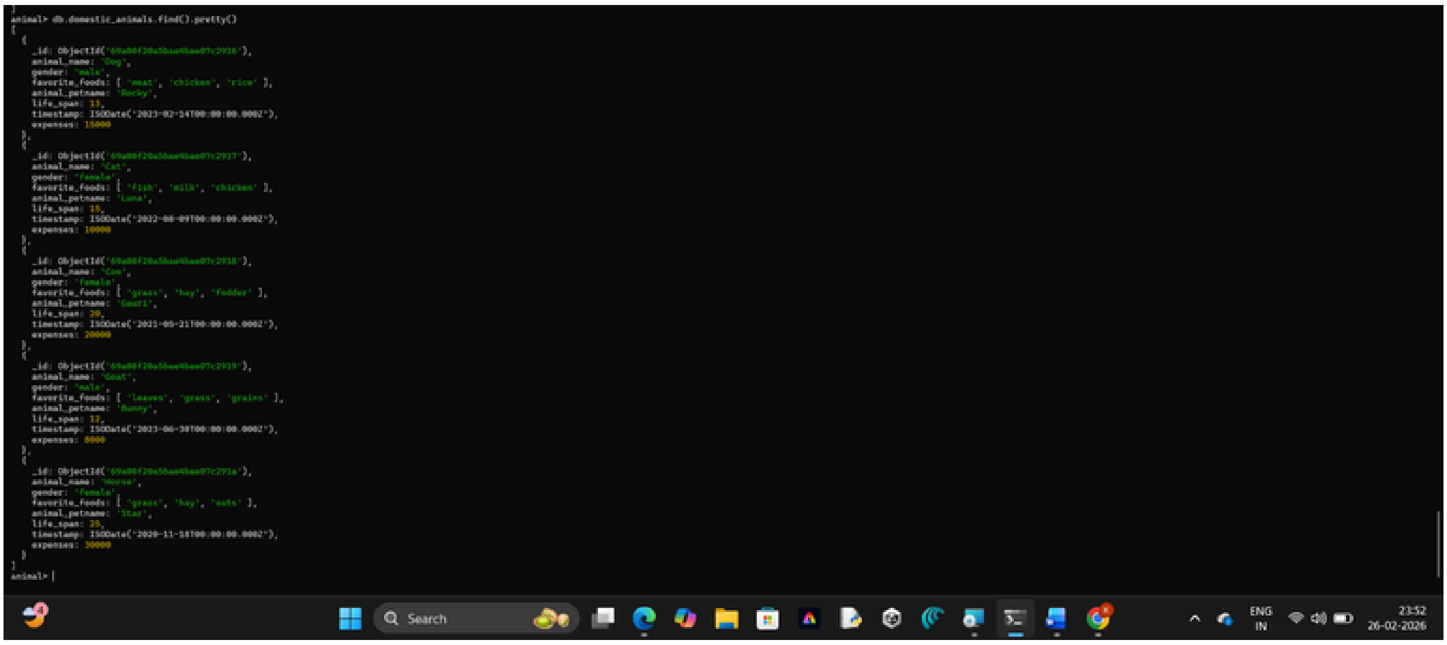


```
}
animal> db.domestic_animals.insertMany([
|     {
|       animal_name: "Dog",
|       gender: "male",
|       favorite_foods: ["meat", "chicken", "rice"],
|       animal_petname: "Rocky",
|       life_span: 13,
|       timestamp: new Date("2023-02-14"),
|       expenses: 15000
|     },
|     {
|       animal_name: "Cat",
|       gender: "female",
|       favorite_foods: ["fish", "milk", "chicken"],
|       animal_petname: "Luna",
|       life_span: 15,
|       timestamp: new Date("2022-08-09"),
|       expenses: 10000
|     },
|     {
|       animal_name: "Cow",
|       gender: "female",
|       favorite_foods: ["grass", "hay", "fodder"],
|       animal_petname: "Gauri",
|       life_span: 20,
|       timestamp: new Date("2021-05-21"),
|       expenses: 20000
|     },
|     {
|       animal_name: "Goat",
|       gender: "male",
|       favorite_foods: ["leaves", "grass", "grains"],
|       animal_petname: "Bunny",
|       life_span: 12,
|       timestamp: new Date("2023-06-30"),
|       expenses: 8000
|     },
|     {
|       animal_name: "Horse",
|       gender: "female",
|       favorite_foods: ["grass", "hay", "oats"],
|       animal_petname: "Star",
|       life_span: 25,
|       timestamp: new Date("2020-11-18"),
|       expenses: 30000
|     }
| ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('69a08f20a5bae4bae07c2916'),
    '1': ObjectId('69a08f20a5bae4bae07c2917'),
    '2': ObjectId('69a08f20a5bae4bae07c2918'),
    '3': ObjectId('69a08f20a5bae4bae07c2919'),
    '4': ObjectId('69a08f20a5bae4bae07c291a')
  }
}
animal> |
```

6. Write a MongoDB query to display all documents available in wild_animals and domastic_animals.

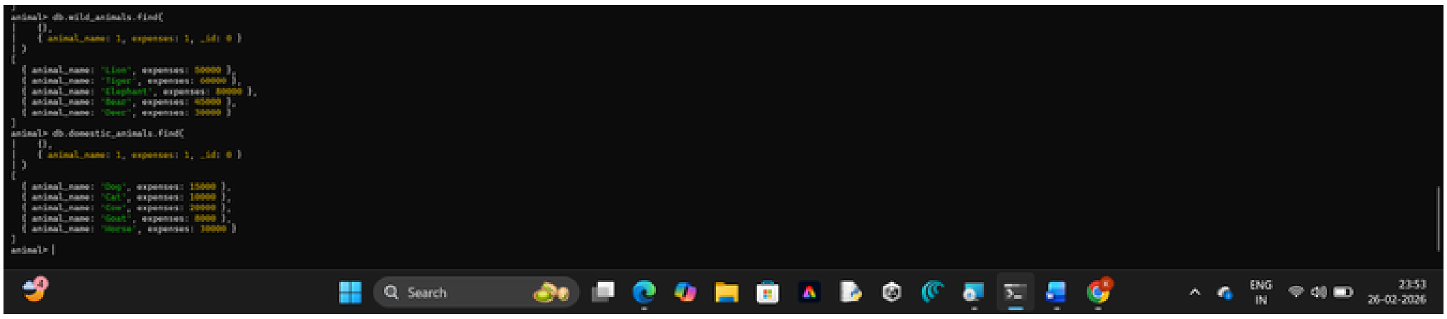


```
animal> db.wild_animals.find().pretty()
[
  {
    _id: ObjectId('69a08ec8a5bae4bae07c2911'),
    animal_name: 'Lion',
    nature: 'harm',
    favorite_foods: [ 'meat', 'deer', 'rabbit' ],
    care_taker_name: 'Ramesh',
    life_span: 14,
    timestamp: ISODate("2022-04-12T00:00:00.000Z"),
    expenses: 50000
  },
  {
    _id: ObjectId('69a08ec8a5bae4bae07c2912'),
    animal_name: 'Tiger',
    nature: 'harm',
    favorite_foods: [ 'meat', 'buffalo', 'deer' ],
    care_taker_name: 'Suresh',
    life_span: 16,
    timestamp: ISODate("2021-06-18T00:00:00.000Z"),
    expenses: 60000
  },
  {
    _id: ObjectId('69a08ec8a5bae4bae07c2913'),
    animal_name: 'Elephant',
    nature: 'harmless',
    favorite_foods: [ 'grass', 'fruits', 'sugarcane' ],
    care_taker_name: 'Mahesh',
    life_span: 60,
    timestamp: ISODate("2020-09-25T00:00:00.000Z"),
    expenses: 80000
  },
  {
    _id: ObjectId('69a08ec8a5bae4bae07c2914'),
    animal_name: 'Bear',
    nature: 'harm',
    favorite_foods: [ 'fish', 'honey', 'berries' ],
    care_taker_name: 'Anil',
    life_span: 25,
    timestamp: ISODate("2023-01-10T00:00:00.000Z"),
    expenses: 45000
  },
  {
    _id: ObjectId('69a08ec8a5bae4bae07c2915'),
    animal_name: 'Deer',
    nature: 'harmless',
    favorite_foods: [ 'grass', 'leaves', 'fruits' ],
    care_taker_name: 'Kiran',
    life_span: 20,
    timestamp: ISODate("2022-11-05T00:00:00.000Z"),
    expenses: 30000
  }
]
animal> |
```



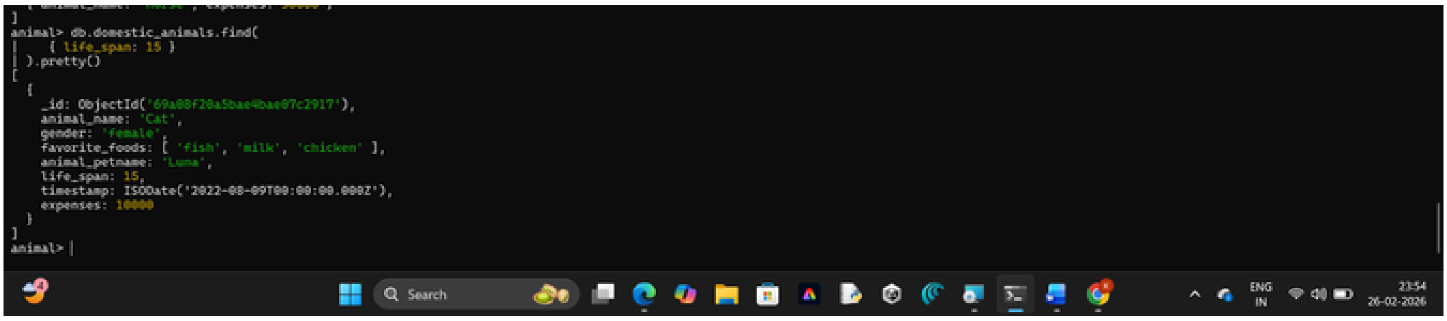
```
animal> db.domestic_animals.find().pretty()
[
  {
    _id: ObjectId('69a08f20a5bae4bae07c2916'),
    animal_name: 'Dog',
    gender: 'male',
    favorite_foods: [ 'meat', 'chicken', 'rice' ],
    animal_petname: 'Rocky',
    life_span: 13,
    timestamp: ISODate("2023-02-14T00:00:00.000Z"),
    expenses: 15000
  },
  {
    _id: ObjectId('69a08f20a5bae4bae07c2917'),
    animal_name: 'Cat',
    gender: 'female',
    favorite_foods: [ 'fish', 'milk', 'chicken' ],
    animal_petname: 'Luna',
    life_span: 15,
    timestamp: ISODate("2022-08-09T00:00:00.000Z"),
    expenses: 10000
  },
  {
    _id: ObjectId('69a08f20a5bae4bae07c2918'),
    animal_name: 'Cow',
    gender: 'female',
    favorite_foods: [ 'grass', 'hay', 'fodder' ],
    animal_petname: 'Gauri',
    life_span: 20,
    timestamp: ISODate("2021-05-21T00:00:00.000Z"),
    expenses: 20000
  },
  {
    _id: ObjectId('69a08f20a5bae4bae07c2919'),
    animal_name: 'Goat',
    gender: 'male',
    favorite_foods: [ 'leaves', 'grass', 'grains' ],
    animal_petname: 'Bunny',
    life_span: 12,
    timestamp: ISODate("2023-06-30T00:00:00.000Z"),
    expenses: 8000
  },
  {
    _id: ObjectId('69a08f20a5bae4bae07c291a'),
    animal_name: 'Horse',
    gender: 'female',
    favorite_foods: [ 'grass', 'hay', 'oats' ],
    animal_petname: 'Star',
    life_span: 25,
    timestamp: ISODate("2020-11-18T00:00:00.000Z"),
    expenses: 30000
  }
]
animal> |
```

7. Write a MongoDB query to display only animal name and expenses in all the collection of the database



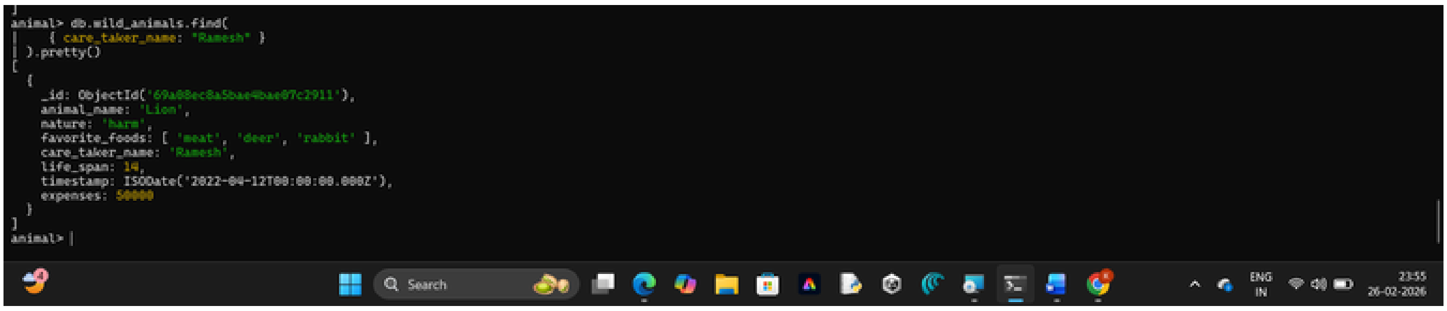
```
animal> db.wild_animals.find(
|     {},
|     { animal_name: 1, expenses: 1, _id: 0 }
| )
[
  { animal_name: 'Lion', expenses: 50000 },
  { animal_name: 'Tiger', expenses: 60000 },
  { animal_name: 'Elephant', expenses: 80000 },
  { animal_name: 'Bear', expenses: 45000 },
  { animal_name: 'Deer', expenses: 30000 }
]
animal> db.domestic_animals.find(
|     {},
|     { animal_name: 1, expenses: 1, _id: 0 }
| )
[
  { animal_name: 'Dog', expenses: 15000 },
  { animal_name: 'Cat', expenses: 10000 },
  { animal_name: 'Cow', expenses: 20000 },
  { animal_name: 'Goat', expenses: 8000 },
  { animal_name: 'Horse', expenses: 30000 }
]
animal> |
```

8. Write a MongoDB query to display domestic_animals whose life is a particular year



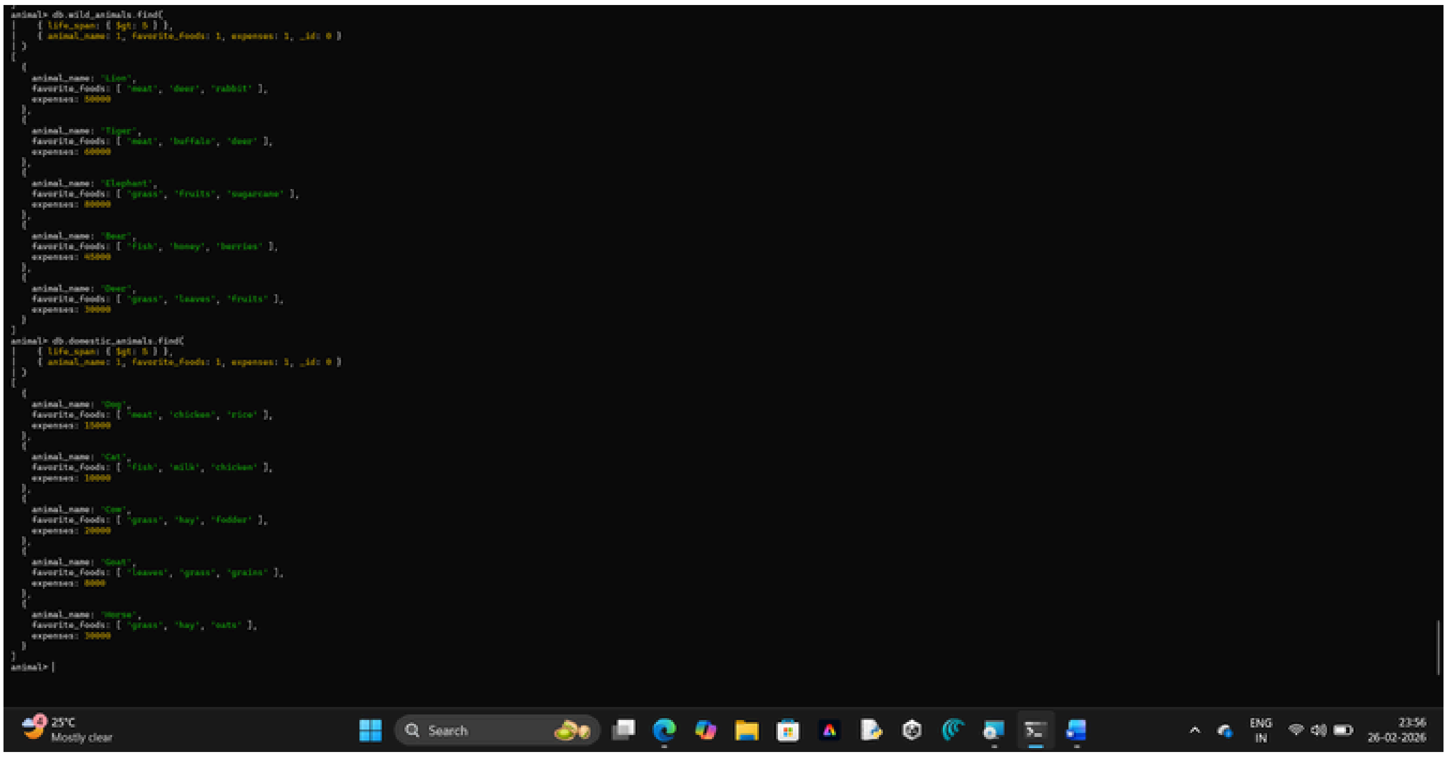
```
]
animal> db.domestic_animals.find(
|     { life_span: 15 }
| ).pretty()
[
  {
    _id: ObjectId('69a08f20a5bae4bae07c2917'),
    animal_name: 'Cat',
    gender: 'female',
    favorite_foods: [ 'fish', 'milk', 'chicken' ],
    animal_petname: 'Luna',
    life_span: 15,
    timestamp: ISODate('2022-08-09T00:00:00.000Z'),
    expenses: 10000
  }
]
animal> |
```

9. Write a MongoDB query to display wild_animals available under a particular care_taker



```
animal> db.wild_animals.find(
|     { care_taker_name: "Ramesh" }
| ).pretty()
[
  {
    _id: ObjectId('69a08ec8a5bae4bae07c2911'),
    animal_name: 'Lion',
    nature: 'harm',
    favorite_foods: [ 'meat', 'deer', 'rabbit' ],
    care_taker_name: 'Ramesh',
    life_span: 14,
    timestamp: ISODate('2022-04-12T00:00:00.000Z'),
    expenses: 50000
  }
]
animal> |
```

10. Write a MongoDB query to display animal name, favorite_foods and expenses details whose lifespan is more than 5 years.



```
animal> db.wild_animals.find(
|     { life_span: { $gt: 5 } },
|     { animal_name: 1, favorite_foods: 1, expenses: 1, _id: 0 }
| )
[
  {
    animal_name: 'Lion',
    favorite_foods: [ 'meat', 'deer', 'rabbit' ],
    expenses: 50000
  },
  {
    animal_name: 'Tiger',
    favorite_foods: [ 'meat', 'buffalo', 'deer' ],
    expenses: 60000
  },
  {
    animal_name: 'Elephant',
    favorite_foods: [ 'grass', 'fruits', 'sugarcane' ],
    expenses: 80000
  },
  {
    animal_name: 'Bear',
    favorite_foods: [ 'fish', 'honey', 'berries' ],
    expenses: 45000
  },
  {
    animal_name: 'Deer',
    favorite_foods: [ 'grass', 'leaves', 'fruits' ],
    expenses: 30000
  }
]
animal> db.domestic_animals.find(
|     { life_span: { $gt: 5 } },
|     { animal_name: 1, favorite_foods: 1, expenses: 1, _id: 0 }
| )
[
  {
    animal_name: 'Dog',
    favorite_foods: [ 'meat', 'chicken', 'rice' ],
    expenses: 15000
  },
  {
    animal_name: 'Cat',
    favorite_foods: [ 'fish', 'milk', 'chicken' ],
    expenses: 10000
  },
  {
    animal_name: 'Cow',
    favorite_foods: [ 'grass', 'hay', 'fodder' ],
    expenses: 20000
  },
  {
    animal_name: 'Goat',
    favorite_foods: [ 'leaves', 'grass', 'grains' ],
    expenses: 8000
  },
  {
    animal_name: 'Horse',
    favorite_foods: [ 'grass', 'hay', 'oats' ],
    expenses: 30000
  }
]
animal> |
```