

Lab Sheet 6: MongoDB Basic commands

Branch/ Class: B.Tech/M.Tech

Date: 26-2-2026

Faculty Name: Prof. S.Gopikrishnan

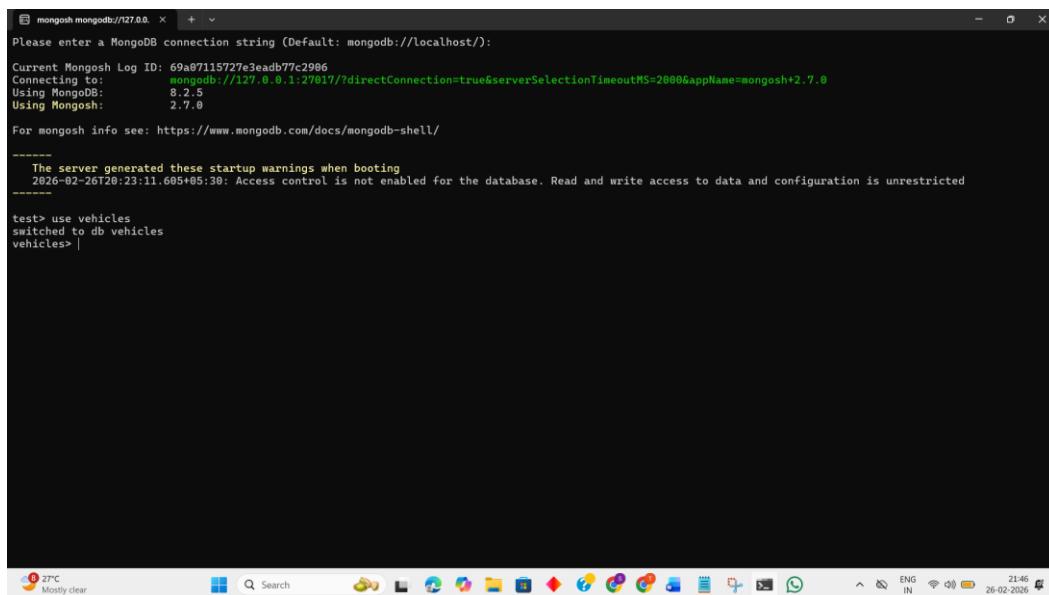
School: SCOPE

Student name: ch.kavya sri

Reg. no.: 23BCE9461

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”.



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.7.0

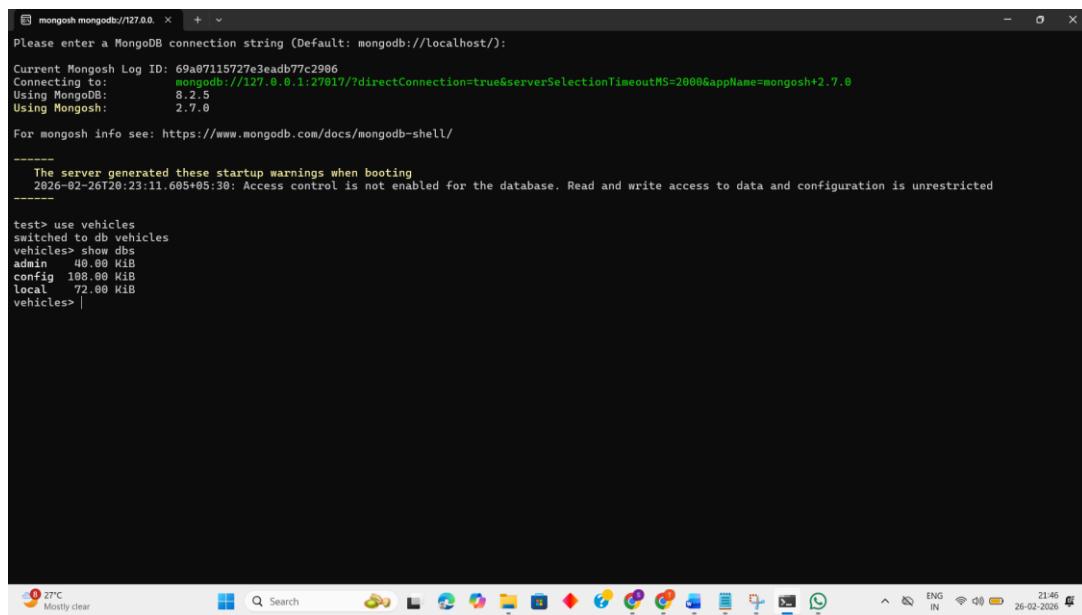
Please enter a MongoDB connection string (Default: mongodb://localhost/):
Current Mongosh Log ID: 69a07115727e3eadb77c2986
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.7.0
Using MongoDB:      8.2.5
Using Mongosh:      2.7.0

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

-----
The server generated these startup warnings when booting
2026-02-26T20:23:11.605+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.



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.7.0

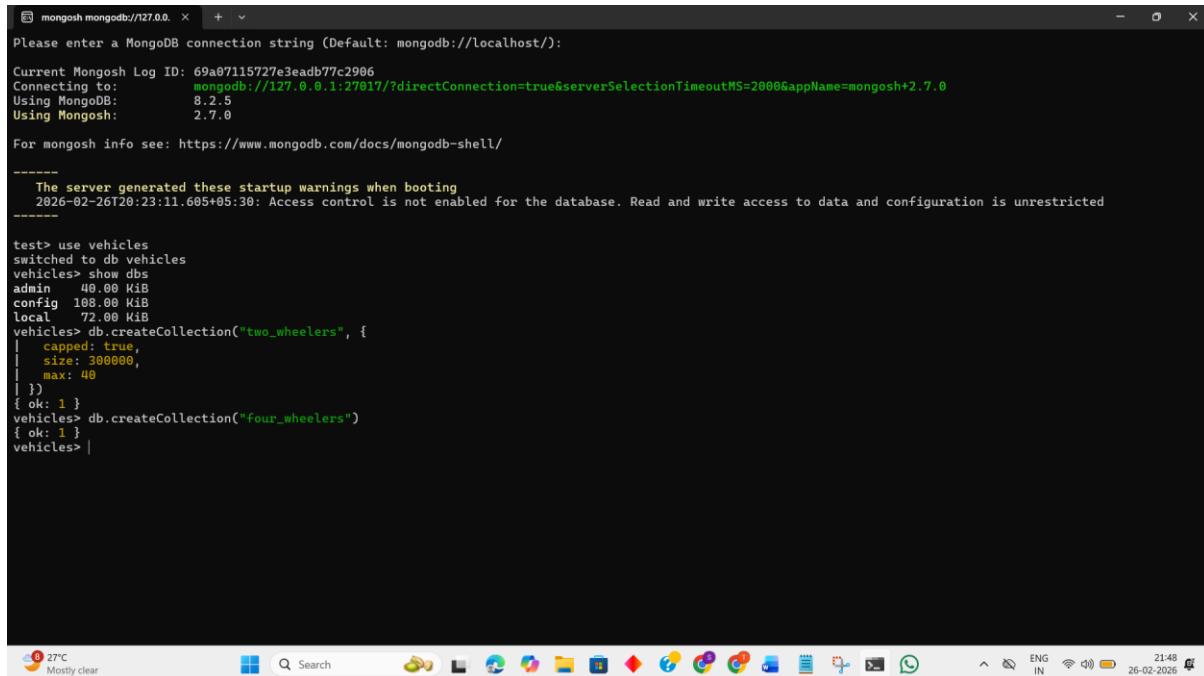
Please enter a MongoDB connection string (Default: mongodb://localhost/):
Current Mongosh Log ID: 69a07115727e3eadb77c2986
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.7.0
Using MongoDB:      8.2.5
Using Mongosh:      2.7.0

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

-----
The server generated these startup warnings when booting
2026-02-26T20:23:11.605+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  108.00 KiB
local   72.00 KiB
vehicles> |
```

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

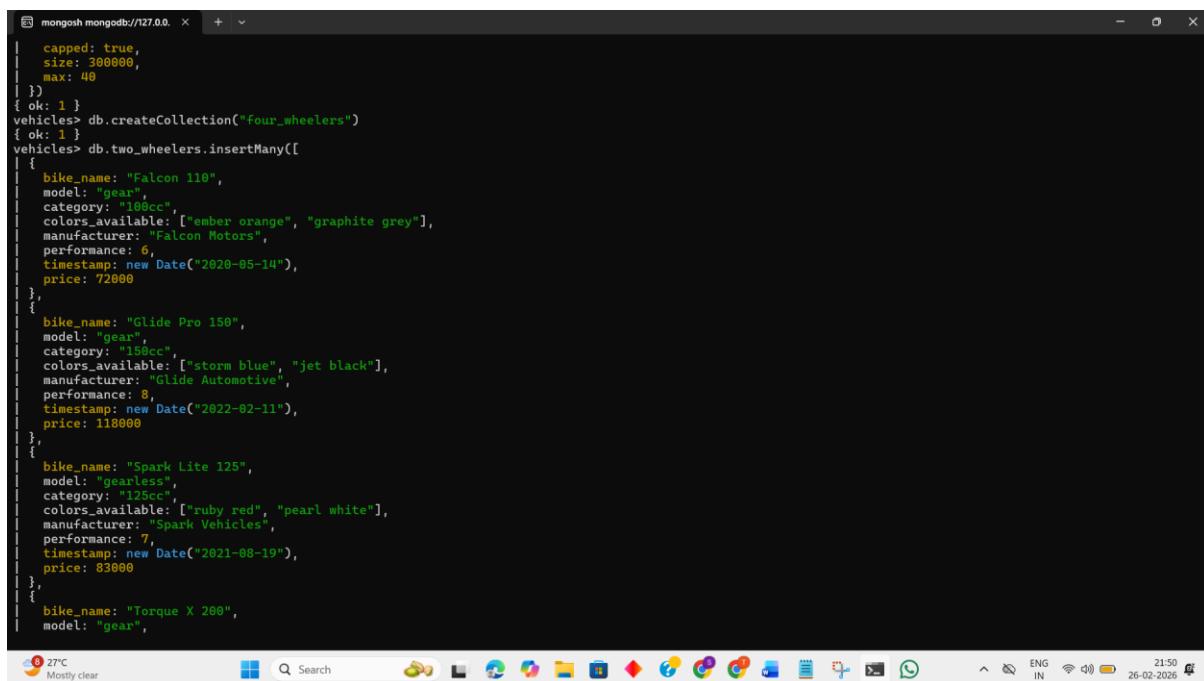


```

mongosh mongodb://127.0.0.1:27017
Please enter a MongoDB connection string (Default: mongodb://localhost):
Current Mongosh Log ID: 69a07115727e3eadb77c2986
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/
-----
The server generated these startup warnings when booting
2026-02-26T20:23:11.605+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   48.00 KiB
config  108.00 KiB
local   72.00 KiB
vehicles> db.createCollection("two_wHEELERS", {
|   capped: true,
|   size: 300000,
|   max: 40
| })
{ 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.



```

capped: true,
size: 300000,
max: 40
)
{ ok: 1 }
vehicles> db.createCollection("four_wHEELERS")
{ ok: 1 }
vehicles> db.two_wHEELERS.insertMany([
{
  bike_name: "Falcon 110",
  model: "gear",
  category: "100cc",
  colors_available: ["ember orange", "graphite grey"],
  manufacturer: "Falcon Motors",
  performance: 6,
  timestamp: new Date("2020-05-14"),
  price: 72000
},
{
  bike_name: "Glide Pro 150",
  model: "gear",
  category: "150cc",
  colors_available: ["storm blue", "jet black"],
  manufacturer: "Glide Automotive",
  performance: 8,
  timestamp: new Date("2022-02-11"),
  price: 118000
},
{
  bike_name: "Spark Lite 125",
  model: "gearless",
  category: "125cc",
  colors_available: ["ruby red", "pearl white"],
  manufacturer: "Spark Vehicles",
  performance: 7,
  timestamp: new Date("2021-08-19"),
  price: 83000
},
{
  bike_name: "Torque X 200",
  model: "gear",

```

```

mongosh mongodb://127.0.0.1:27017
{
  bike_name: "Spark Lite 125",
  model: "gearless",
  category: "125cc",
  colors_available: ["ruby red", "pearl white"],
  manufacturer: "Spark Vehicles",
  performance: 7,
  timestamp: new Date("2021-08-19"),
  price: 83000
},
{
  bike_name: "Torque X 200",
  model: "gear",
  category: "200cc",
  colors_available: ["matte green", "carbon black"],
  manufacturer: "Torque Wheels",
  performance: 9,
  timestamp: new Date("2023-01-09"),
  price: 165000
},
{
  bike_name: "Urban E-Ride",
  model: "gearless",
  category: "150cc",
  colors_available: ["electric blue", "silver frost"],
  manufacturer: "Urban Mobility",
  performance: 8,
  timestamp: new Date("2023-06-30"),
  price: 145000
}
])
acknowledged: true,
insertedIds: [
  '69a07294727e3eadb77c2907',
  '69a07294727e3eadb77c2908',
  '69a07294727e3eadb77c2909',
  '69a07294727e3eadb77c290a',
  '69a07294727e3eadb77c290b'
]
}
vehicles> |

```

27°C Mostly clear Search ENG IN 21:50 26-02-2026

- 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.

```

mongosh mongodb://127.0.0.1:27017
}
vehicles> db.four_wheelers.insertMany([
{
  vehicle_name: "Atlas Compact",
  model: "own",
  category: "car",
  variants: ["petrol", "manual"],
  manufacturer: "Atlas Motors",
  performance: 7,
  timestamp: new Date("2022-03-18"),
  price: 620000
},
{
  vehicle_name: "RoadKing Carrier",
  model: "commercial",
  category: "lorry",
  variants: ["diesel"],
  manufacturer: "RoadKing Ltd",
  performance: 8,
  timestamp: new Date("2021-12-22"),
  price: 2100000
},
{
  vehicle_name: "Metro Shuttle",
  model: "commercial",
  category: "bus",
  variants: ["diesel", "city edition"],
  manufacturer: "Metro Transport",
  performance: 9,
  timestamp: new Date("2020-09-10"),
  price: 3400000
},
{
  vehicle_name: "CargoMax 300",
  model: "commercial",
  category: "containers",
  variants: ["diesel", "heavy duty"],
  manufacturer: "CargoMax Industries",
  performance: 8,
  timestamp: new Date("2019-04-16"),
  price: 4800000
}
]

```

27°C Mostly clear Search ENG IN 21:51 26-02-2026

```
mongosh mongodb://127.0.0.1:27017
vehicle_name: "Metro Shuttle",
model: "commercial",
category: "bus",
variants: ["diesel", "city edition"],
manufacturer: "Metro Transport",
performance: 9,
timestamp: new Date("2020-09-10"),
price: 3400000
},
{
  vehicle_name: "CargoMax 300",
  model: "commercial",
  category: "containers",
  variants: ["diesel", "heavy duty"],
  manufacturer: "CargoMax Industries",
  performance: 8,
  timestamp: new Date("2019-04-16"),
  price: 4800000
},
{
  vehicle_name: "Nova Prime",
  model: "own",
  category: "car",
  variants: ["diesel", "automatic"],
  manufacturer: "Nova Automotive",
  performance: 6,
  timestamp: new Date("2023-07-12"),
  price: 890000
}
])
acknowledged: true,
insertedIds: [
  '_0': ObjectId('69a072f4727e3eadb77c290c'),
  '_1': ObjectId('69a072f4727e3eadb77c290d'),
  '_2': ObjectId('69a072f4727e3eadb77c290e'),
  '_3': ObjectId('69a072f4727e3eadb77c290f'),
  '_4': ObjectId('69a072f4727e3eadb77c2910')
]
}
vehicles> |
```

27°C Mostly clear Search 🌐 📱 📺 🎮 📸 📥 🚗 📈 📊 📉 📕 📖 📧 📨 📩 📪 📫 📬 📭 📮 📯 📰 📲 📴 📵 📷 📸 📹 📻 📽 📿 ENG IN 21:51 26-02-2026

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

```
mongosh mongodb://127.0.0.1:27017
}
vehicles> db.two_wHEELERS.find()
| db.four_wHEELERS.find()
[
  {
    _id: ObjectId('69a072f4727e3eadb77c290c'),
    vehicle_name: 'Atlas Compact',
    model: 'own',
    category: 'car',
    variants: ['petrol', 'manual'],
    manufacturer: 'Atlas Motors',
    performance: 7,
    timestamp: ISODate('2022-03-18T00:00:00.000Z'),
    price: 620000
  },
  {
    _id: ObjectId('69a072f4727e3eadb77c290d'),
    vehicle_name: 'RoadKing Carrier',
    model: 'commercial',
    category: 'lorry',
    variants: ['diesel'],
    manufacturer: 'RoadKing Ltd',
    performance: 8,
    timestamp: ISODate('2021-12-22T00:00:00.000Z'),
    price: 2100000
  },
  {
    _id: ObjectId('69a072f4727e3eadb77c290e'),
    vehicle_name: 'Metro Shuttle',
    model: 'commercial',
    category: 'bus',
    variants: ['diesel', 'city edition'],
    manufacturer: 'Metro Transport',
    performance: 9,
    timestamp: ISODate('2020-09-10T00:00:00.000Z'),
    price: 3400000
  },
  {
    _id: ObjectId('69a072f4727e3eadb77c290f'),
    vehicle_name: 'CargoMax 300',
    model: 'commercial',
```

27°C Mostly clear Search 🌐 📱 📺 🎮 📸 📥 📈 📊 📉 📕 📖 📧 📨 📩 📪 📫 📬 📭 📮 📯 📰 📲 📴 📵 📷 📸 📹 📻 📽 📿 ENG IN 21:53 26-02-2026

```
mongosh mongodb://127.0.0.1:27017
{
  variants: [ 'diesel' ],
  manufacturer: 'RoadKing Ltd',
  performance: 8,
  timestamp: ISODate('2021-12-22T00:00:00.000Z'),
  price: 2100000
},
{
  _id: ObjectId('69a072f4727e3eadb77c290e'),
  vehicle_name: 'Metro Shuttle',
  model: 'commercial',
  category: 'bus',
  variants: [ 'diesel', 'city edition' ],
  manufacturer: 'Metro Transport',
  performance: 9,
  timestamp: ISODate('2020-09-10T00:00:00.000Z'),
  price: 3400000
},
{
  _id: ObjectId('69a072f4727e3eadb77c290f'),
  vehicle_name: 'CargoMax 300',
  model: 'commercial',
  category: 'containers',
  variants: [ 'diesel', 'heavy duty' ],
  manufacturer: 'CargoMax Industries',
  performance: 8,
  timestamp: ISODate('2019-04-16T00:00:00.000Z'),
  price: 4800000
},
{
  _id: ObjectId('69a072f4727e3eadb77c2910'),
  vehicle_name: 'Nova Prime',
  model: 'own',
  category: 'car',
  variants: [ 'diesel', 'automatic' ],
  manufacturer: 'Nova Automotive',
  performance: 6,
  timestamp: ISODate('2023-07-12T00:00:00.000Z'),
  price: 890000
}
]
vehicles> |
```

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

```
mongosh mongodb://127.0.0.1:27017
},
{
  _id: ObjectId('69a072f4727e3eadb77c290e'),
  vehicle_name: 'CargoMax 300',
  model: 'commercial',
  category: 'containers',
  variants: [ 'diesel', 'heavy duty' ],
  manufacturer: 'CargoMax Industries',
  performance: 8,
  timestamp: ISODate('2019-04-16T00:00:00.000Z'),
  price: 4800000
},
{
  _id: ObjectId('69a072f4727e3eadb77c2910'),
  vehicle_name: 'Nova Prime',
  model: 'own',
  category: 'car',
  variants: [ 'diesel', 'automatic' ],
  manufacturer: 'Nova Automotive',
  performance: 6,
  timestamp: ISODate('2023-07-12T00:00:00.000Z'),
  price: 890000
}
]
vehicles> db.two_wheelers.find({}, { bike_name: 1, price: 1, _id: 0 })
[
  { bike_name: 'Falcon 110', price: 72000 },
  { bike_name: 'Glide Pro 150', price: 118000 },
  { bike_name: 'Spark Lite 125', price: 83000 },
  { bike_name: 'Torque X 200', price: 165000 },
  { bike_name: 'Urban E-Ride', price: 145000 }
]
vehicles> db.four_wheelers.find({}, { vehicle_name: 1, price: 1, _id: 0 })
[
  { vehicle_name: 'Atlas Compact', price: 620000 },
  { vehicle_name: 'RoadKing Carrier', price: 2100000 },
  { vehicle_name: 'Metro Shuttle', price: 3400000 },
  { vehicle_name: 'CargoMax 300', price: 4800000 },
  { vehicle_name: 'Nova Prime', price: 890000 }
]
vehicles> |
```

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

```
mongosh mongodb://127.0.0.1:27017
vehicle_name: 'Nova Prime',
model: 'own',
category: 'car',
variants: [ 'diesel', 'automatic' ],
manufacturer: 'Nova Automotive',
performance: 6,
timestamp: ISODate('2023-07-12T00:00:00.000Z'),
price: 890000
}
]
vehicles> db.two_wheelers.find({}, { bike_name: 1, price: 1, _id: 0 })
[
{ bike_name: 'Falcon 110', price: 72000 },
{ bike_name: 'Glide Pro 150', price: 118000 },
{ bike_name: 'Spark Lite 125', price: 83000 },
{ bike_name: 'Torque X 200', price: 165000 },
{ bike_name: 'Urban E-Ride', price: 145000 }
]
vehicles> db.four_wheelers.find({}, { vehicle_name: 1, price: 1, _id: 0 })
[
{ vehicle_name: 'Atlas Compact', price: 620000 },
{ vehicle_name: 'RoadKing Carrier', price: 2100000 },
{ vehicle_name: 'Metro Shuttle', price: 3400000 },
{ vehicle_name: 'CargoMax 300', price: 4800000 },
{ vehicle_name: 'Nova Prime', price: 890000 }
]
vehicles> db.two_wheelers.find({ manufacturer: "Torque Wheels" })
[
{
_id: ObjectId('69a072f4727e3eadb77c290a'),
bike_name: 'Torque X 200',
model: 'gear',
category: '200cc',
colors_available: [ 'matte green', 'carbon black' ],
manufacturer: 'Torque Wheels',
performance: 9,
timestamp: ISODate('2023-01-09T00:00:00.000Z'),
price: 165000
}
]
vehicles> |
```

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

```
mongosh mongodb://127.0.0.1:27017
colors_available: [ 'matte green', 'carbon black' ],
manufacturer: 'Torque Wheels',
performance: 9,
timestamp: ISODate('2023-01-09T00:00:00.000Z'),
price: 165000
}
]
vehicles> db.four_wheelers.find({ variants: "diesel" })
[
{
_id: ObjectId('69a072f4727e3eadb77c290d'),
vehicle_name: 'RoadKing Carrier',
model: 'commercial',
category: 'lorry',
variants: [ 'diesel' ],
manufacturer: 'RoadKing Ltd',
performance: 8,
timestamp: ISODate('2021-12-22T00:00:00.000Z'),
price: 2100000
},
{
_id: ObjectId('69a072f4727e3eadb77c290e'),
vehicle_name: 'Metro Shuttle',
model: 'commercial',
category: 'bus',
variants: [ 'diesel', 'city edition' ],
manufacturer: 'Metro Transport',
performance: 9,
timestamp: ISODate('2020-09-10T00:00:00.000Z'),
price: 3400000
},
{
_id: ObjectId('69a072f4727e3eadb77c290f'),
vehicle_name: 'CargoMax 300',
model: 'commercial',
category: 'containers',
variants: [ 'diesel', 'heavy duty' ],
manufacturer: 'CargoMax Industries',
performance: 8,
timestamp: ISODate('2019-04-16T00:00:00.000Z'),
price: 4800000
}
```

```
| mongosh mongodb://127.0.0.1:27017 - | + - 
| variants: [ 'diesel' ],
| manufacturer: 'RoadKing Ltd',
| performance: 8,
| timestamp: ISODate('2021-12-22T00:00:00.000Z'),
| price: 2100000
|,
| {
|   _id: ObjectId('69a072f4727e3eadb77c290e'),
|   vehicle_name: 'Metro Shuttle',
|   model: 'commercial',
|   category: 'bus',
|   variants: [ 'diesel', 'city edition' ],
|   manufacturer: 'Metro Transport',
|   performance: 9,
|   timestamp: ISODate('2020-09-10T00:00:00.000Z'),
|   price: 3000000
| },
| {
|   _id: ObjectId('69a072f4727e3eadb77c290f'),
|   vehicle_name: 'CargoMax 300',
|   model: 'commercial',
|   category: 'containers',
|   variants: [ 'diesel', 'heavy duty' ],
|   manufacturer: 'CargoMax Industries',
|   performance: 8,
|   timestamp: ISODate('2019-04-16T00:00:00.000Z'),
|   price: 4000000
| },
| {
|   _id: ObjectId('69a072f4727e3eadb77c2910'),
|   vehicle_name: 'Nova Prime',
|   model: 'own',
|   category: 'car',
|   variants: [ 'diesel', 'automatic' ],
|   manufacturer: 'Nova Automotive',
|   performance: 6,
|   timestamp: ISODate('2023-07-12T00:00:00.000Z'),
|   price: 890000
| }
]
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: 'Falcon 110',
|   category: '100cc',
|   manufacturer: 'Falcon Motors'
| },
| {
|   bike_name: 'Glide Pro 150',
|   category: '150cc',
|   manufacturer: 'Glide Automotive'
| },
| {
|   bike_name: 'Spark Lite 125',
|   category: '125cc',
|   manufacturer: 'Spark Vehicles'
| },
| {
|   bike_name: 'Torque X 200',
|   category: '200cc',
|   manufacturer: 'Torque Wheels'
| },
| {
|   bike_name: 'Urban E-Ride',
|   category: '150cc',
|   manufacturer: 'Urban Mobility'
| }
]
vehicles> db.four_wheelers.find(
| { performance: { $gt: 5 } },
| { vehicle_name: 1, category: 1, manufacturer: 1, _id: 0 }
| )
[
| {
|   vehicle_name: 'Atlas Compact',
|   category: 'car',
| }
```

```
mongosh mongodb://127.0.0.1:27017
{
  category: '200cc',
  manufacturer: 'Torque Wheels'
},
{
  bike_name: 'Urban E-Ride',
  category: '150cc',
  manufacturer: 'Urban Mobility'
}
]
vehicles> db.four_wheelers.find(
| { performance: { $gt: 5 } },
| { vehicle_name: 1, category: 1, manufacturer: 1, _id: 0 }
|
[
{
  vehicle_name: 'Atlas Compact',
  category: 'car',
  manufacturer: 'Atlas Motors'
},
{
  vehicle_name: 'RoadKing Carrier',
  category: 'lorry',
  manufacturer: 'RoadKing Ltd'
},
{
  vehicle_name: 'Metro Shuttle',
  category: 'bus',
  manufacturer: 'Metro Transport'
},
{
  vehicle_name: 'CargoMax 300',
  category: 'containers',
  manufacturer: 'CargoMax Industries'
},
{
  vehicle_name: 'Nova Prime',
  category: 'car',
  manufacturer: 'Nova Automotive'
}
]
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'.

```
},
{
  bike_name: 'Urban E-Ride',
  category: '150cc',
  manufacturer: 'Urban Mobility'
}
]
vehicles> db.four_wheelers.find(
| { performance: { $gt: 5 } },
| { vehicle_name: 1, category: 1, manufacturer: 1, _id: 0 }
|
[
{
  vehicle_name: 'Atlas Compact',
  category: 'car',
  manufacturer: 'Atlas Motors'
},
{
  vehicle_name: 'RoadKing Carrier',
  category: 'lorry',
  manufacturer: 'RoadKing Ltd'
},
{
  vehicle_name: 'Metro Shuttle',
  category: 'bus',
  manufacturer: 'Metro Transport'
},
{
  vehicle_name: 'CargoMax 300',
  category: 'containers',
  manufacturer: 'CargoMax Industries'
},
{
  vehicle_name: 'Nova Prime',
  category: 'car',
  manufacturer: 'Nova Automotive'
}
]
vehicles> use animal
switched to db animal
animal> |
```



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

```
[mongosh mongoDB://127.0.0.1:27017] + ···
]
vehicles> db.four_wheelers.find(
| { performance: { $gt: 5 } },
| { vehicle_name: 1, category: 1, manufacturer: 1, _id: 0 }
| )
[
  {
    vehicle_name: 'Atlas Compact',
    category: 'car',
    manufacturer: 'Atlas Motors'
  },
  {
    vehicle_name: 'RoadKing Carrier',
    category: 'lorry',
    manufacturer: 'RoadKing Ltd'
  },
  {
    vehicle_name: 'Metro Shuttle',
    category: 'bus',
    manufacturer: 'Metro Transport'
  },
  {
    vehicle_name: 'CargoMax 300',
    category: 'containers',
    manufacturer: 'CargoMax Industries'
  },
  {
    vehicle_name: 'Nova Prime',
    category: 'car',
    manufacturer: 'Nova Automotive'
  }
]
vehicles> use animal
switched to db animal
animal> show dbs
admin      40.00 KiB
config     96.00 KiB
local      72.00 KiB
vehicles   80.00 KiB
animal> |
```



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

```
[mongosh mongoDB://127.0.0.1:27017] + ···
[
  {
    vehicle_name: 'Atlas Compact',
    category: 'car',
    manufacturer: 'Atlas Motors'
  },
  {
    vehicle_name: 'RoadKing Carrier',
    category: 'lorry',
    manufacturer: 'RoadKing Ltd'
  },
  {
    vehicle_name: 'Metro Shuttle',
    category: 'bus',
    manufacturer: 'Metro Transport'
  },
  {
    vehicle_name: 'CargoMax 300',
    category: 'containers',
    manufacturer: 'CargoMax Industries'
  },
  {
    vehicle_name: 'Nova Prime',
    category: 'car',
    manufacturer: 'Nova Automotive'
  }
]
vehicles> use animal
switched to db animal
animal> show dbs
admin      40.00 KiB
config     96.00 KiB
local      72.00 KiB
vehicles   80.00 KiB
animal> db.createCollection("wild_animals", {
|   capped: true,
|   size: 300000,
|   max: 50
| })
{ ok: 1 }
animal> |
```



```
mongosh mongodb://127.0.0.1:27017
{
  vehicle_name: 'Atlas Compact',
  category: 'car',
  manufacturer: 'Atlas Motors'
},
{
  vehicle_name: 'RoadKing Carrier',
  category: 'lorry',
  manufacturer: 'RoadKing Ltd'
},
{
  vehicle_name: 'Metro Shuttle',
  category: 'bus',
  manufacturer: 'Metro Transport'
},
{
  vehicle_name: 'CargoMax 300',
  category: 'containers',
  manufacturer: 'CargoMax Industries'
},
{
  vehicle_name: 'Nova Prime',
  category: 'car',
  manufacturer: 'Nova Automotive'
}
]
vehicles> use animal
switched to db animal
animal> show dbs
admin   40.00 KiB
config  96.00 KiB
local   72.00 KiB
vehicles 80.00 KiB
animal> db.createCollection("wild_animals", {
|   capped: true,
|   size: 300000,
|   max: 50
| })
{ 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.

```
mongosh mongodb://127.0.0.1:27017
{
  vehicle_name: 'Nova Prime',
  category: 'car',
  manufacturer: 'Nova Automotive'
}
]
vehicles> use animal
switched to db animal
animal> show dbs
admin   40.00 KiB
config  96.00 KiB
local   72.00 KiB
vehicles 80.00 KiB
animal> db.createCollection("wild_animals", {
|   capped: true,
|   size: 300000,
|   max: 50
| })
{ ok: 1 }
animal> db.createCollection("domestic_animals")
{ ok: 1 }
animal> db.wild_animals.insertMany([
|   {
|     animal_name: "Red Panda",
|     nature: "harmless",
|     favorite_foods: ["bamboo", "fruits"],
|     care_taker_name: "Anil Kumar",
|     life_span: 14,
|     timestamp: new Date("2020-05-15"),
|     expenses: 22000
|   },
|   {
|     animal_name: "Black Panther",
|     nature: "harm",
|     favorite_foods: ["meat", "deer"],
|     care_taker_name: "Ritu Singh",
|     life_span: 18,
|     timestamp: new Date("2019-03-10"),
|     expenses: 60000
|   },
|   {
|     animal_name: "Lion",
|     nature: "harm",
|     favorite_foods: ["meat", "dear"],
|     care_taker_name: "Shivam Singh",
|     life_span: 20,
|     timestamp: new Date("2018-01-01"),
|     expenses: 45000
|   },
|   {
|     animal_name: "Tiger",
|     nature: "harm",
|     favorite_foods: ["meat", "dear"],
|     care_taker_name: "Karan Singh",
|     life_span: 22,
|     timestamp: new Date("2017-07-01"),
|     expenses: 50000
|   },
|   {
|     animal_name: "Leopard",
|     nature: "harm",
|     favorite_foods: ["meat", "dear"],
|     care_taker_name: "Vishal Singh",
|     life_span: 19,
|     timestamp: new Date("2016-12-01"),
|     expenses: 40000
|   }
| ])
```

```

    expenses: 60000
},
{
  animal_name: "Giraffe",
  nature: "harmless",
  favorite_foods: ["leaves"],
  care_taker_name: "Anil Kumar",
  life_span: 25,
  timestamp: new Date("2021-07-25"),
  expenses: 35000
},
{
  animal_name: "Crocodile",
  nature: "harmful",
  favorite_foods: ["fish", "meat"],
  care_taker_name: "Megha Rao",
  life_span: 70,
  timestamp: new Date("2018-11-12"),
  expenses: 40000
},
{
  animal_name: "Zebra",
  nature: "harmless",
  favorite_foods: ["grass"],
  care_taker_name: "Ritu Singh",
  life_span: 20,
  timestamp: new Date("2022-01-18"),
  expenses: 18000
}
])
{
  acknowledged: true,
  insertedIds: [
    '0': ObjectId('69a075a5727e3eadb77c2911'),
    '1': ObjectId('69a075a5727e3eadb77c2912'),
    '2': ObjectId('69a075a5727e3eadb77c2913'),
    '3': ObjectId('69a075a5727e3eadb77c2914'),
    '4': ObjectId('69a075a5727e3eadb77c2915')
  ]
}
animal> |

```

27°C Mostly clear 22:02 ENG IN 26-02-2026

- 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.

```

| ])
{
  acknowledged: true,
  insertedIds: [
    '0': ObjectId('69a075a5727e3eadb77c2911'),
    '1': ObjectId('69a075a5727e3eadb77c2912'),
    '2': ObjectId('69a075a5727e3eadb77c2913'),
    '3': ObjectId('69a075a5727e3eadb77c2914'),
    '4': ObjectId('69a075a5727e3eadb77c2915')
  ]
}
animal> db.domestic_animals.insertMany([
| {
  animal_name: "Dog",
  gender: "male",
  favorite_foods: ["meat", "dog food"],
  animal_petname: "Rocky",
  life_span: 13,
  timestamp: new Date("2021-09-20"),
  expenses: 9000
},
| {
  animal_name: "Cat",
  gender: "female",
  favorite_foods: ["fish", "milk"],
  animal_petname: "Luna",
  life_span: 15,
  timestamp: new Date("2020-02-14"),
  expenses: 7000
},
| {
  animal_name: "Cow",
  gender: "female",
  favorite_foods: ["grass", "fodder"],
  animal_petname: "Kamadhenu",
  life_span: 22,
  timestamp: new Date("2017-08-10"),
  expenses: 12000
},
| {
  animal_name: "Goat",

```

27°C Mostly clear 22:03 ENG IN 26-02-2026

```
mongosh mongodb://127.0.0.1:27017/test
expenses: 7000
},
{
  animal_name: "Cow",
  gender: "female",
  favorite_foods: ["grass", "fodder"],
  animal_petsname: "Kamadhenu",
  life_span: 22,
  timestamp: new Date("2017-08-10"),
  expenses: 12000
},
{
  animal_name: "Goat",
  gender: "male",
  favorite_foods: ["leaves", "grass"],
  animal_petsname: "Bunty",
  life_span: 10,
  timestamp: new Date("2019-06-05"),
  expenses: 6000
},
{
  animal_name: "Horse",
  gender: "male",
  favorite_foods: ["hay", "grains"],
  animal_petsname: "Blaze",
  life_span: 28,
  timestamp: new Date("2016-04-30"),
  expenses: 25000
})
})
{
  acknowledged: true,
  insertedIds: [
    '0': ObjectId('69a075d7727e3eadb77c2916'),
    '1': ObjectId('69a075d7727e3eadb77c2917'),
    '2': ObjectId('69a075d7727e3eadb77c2918'),
    '3': ObjectId('69a075d7727e3eadb77c2919'),
    '4': ObjectId('69a075d7727e3adb77c291a')
  ]
}
animal> |
```

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

```
mongosh mongodb://127.0.0.1:27017/test
}
animal> db.wild_animals.find()
[
  {
    _id: ObjectId('69a075a5727e3eadb77c2911'),
    animal_name: 'Red Panda',
    nature: 'harmless',
    favorite_foods: ['bamboo', 'fruits'],
    care_taker_name: 'Anil Kumar',
    life_span: 14,
    timestamp: ISODate('2028-05-15T00:00:00.000Z'),
    expenses: 22000
  },
  {
    _id: ObjectId('69a075a5727e3eadb77c2912'),
    animal_name: 'Black Panther',
    nature: 'harm',
    favorite_foods: ['meat', 'deer'],
    care_taker_name: 'Ritu Singh',
    life_span: 18,
    timestamp: ISODate('2019-03-10T00:00:00.000Z'),
    expenses: 60000
  },
  {
    _id: ObjectId('69a075a5727e3eadb77c2913'),
    animal_name: 'Giraffe',
    nature: 'harmless',
    favorite_foods: ['leaves'],
    care_taker_name: 'Anil Kumar',
    life_span: 25,
    timestamp: ISODate('2021-07-25T00:00:00.000Z'),
    expenses: 35000
  },
  {
    _id: ObjectId('69a075a5727e3eadb77c2914'),
    animal_name: 'Crocodile',
    nature: 'harm',
    favorite_foods: ['fish', 'meat'],
    care_taker_name: 'Megha Rao',
    life_span: 70,
    timestamp: ISODate('2018-11-12T00:00:00.000Z'),
```

```
mongosh mongodb://127.0.0.1:27017
expenses: 35000
},
{
  _id: ObjectId('69a075a5727e3eadb77c2914'),
  animal_name: 'Crocodile',
  nature: 'harm',
  favorite_foods: [ 'fish', 'meat' ],
  care_taker_name: 'Megha Rao',
  life_span: 70,
  timestamp: ISODate('2018-11-12T00:00:00.000Z'),
  expenses: 40000
},
{
  _id: ObjectId('69a075a5727e3eadb77c2915'),
  animal_name: 'Zebra',
  nature: 'harmless',
  favorite_foods: [ 'grass' ],
  care_taker_name: 'Ritu Singh',
  life_span: 20,
  timestamp: ISODate('2022-01-18T00:00:00.000Z'),
  expenses: 18000
}
]
animal> db.domestic_animals.find()
[
  {
    _id: ObjectId('69a075d7727e3eadb77c2916'),
    animal_name: 'Dog',
    gender: 'male',
    favorite_foods: [ 'meat', 'dog food' ],
    animal_petsname: 'Rocky',
    life_span: 13,
    timestamp: ISODate('2021-09-20T00:00:00.000Z'),
    expenses: 9000
  },
  {
    _id: ObjectId('69a075d7727e3eadb77c2917'),
    animal_name: 'Cat',
    gender: 'female',
    favorite_foods: [ 'fish', 'milk' ],
    animal_petsname: 'Luna',
    life_span: 15,
    timestamp: ISODate('2020-02-14T00:00:00.000Z'),
    expenses: 7000
  },
  {
    _id: ObjectId('69a075d7727e3eadb77c2918'),
    animal_name: 'Cow',
    gender: 'female',
    favorite_foods: [ 'grass', 'fodder' ],
    animal_petsname: 'Kamadhenu',
    life_span: 22,
    timestamp: ISODate('2017-08-10T00:00:00.000Z'),
    expenses: 12000
  },
  {
    _id: ObjectId('69a075d7727e3eadb77c2919'),
    animal_name: 'Goat',
    gender: 'male',
    favorite_foods: [ 'leaves', 'grass' ],
    animal_petsname: 'Bunty',
    life_span: 10,
    timestamp: ISODate('2019-06-05T00:00:00.000Z'),
    expenses: 6000
  },
  {
    _id: ObjectId('69a075d7727e3eadb77c291a'),
    animal_name: 'Horse',
    gender: 'male',
    favorite_foods: [ 'hay', 'grains' ],
    animal_petsname: 'Blaze',
    life_span: 28,
    timestamp: ISODate('2016-04-30T00:00:00.000Z'),
    expenses: 25000
  }
]
animal> |
```

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

```

mongosh mongodb://127.0.0.1:27017
{
  timestamp: ISODate('2017-08-10T00:00:00.000Z'),
  expenses: 12000
},
{
  _id: ObjectId('69a075d7727e3eadb77c2919'),
  animal_name: 'Goat',
  gender: 'male',
  favorite_foods: [ 'leaves', 'grass' ],
  animal_petsname: 'Bunty',
  life_span: 10,
  timestamp: ISODate('2019-06-05T00:00:00.000Z'),
  expenses: 6000
},
{
  _id: ObjectId('69a075d7727e3eadb77c291a'),
  animal_name: 'Horse',
  gender: 'male',
  favorite_foods: [ 'hay', 'grains' ],
  animal_petsname: 'Blaze',
  life_span: 28,
  timestamp: ISODate('2016-04-30T00:00:00.000Z'),
  expenses: 25000
}
]
animal> db.wild_animals.find({}, { animal_name: 1, expenses: 1, _id: 0 })
[
  { animal_name: 'Red Panda', expenses: 22000 },
  { animal_name: 'Black Panther', expenses: 60000 },
  { animal_name: 'Giraffe', expenses: 35000 },
  { animal_name: 'Crocodile', expenses: 40000 },
  { animal_name: 'Zebra', expenses: 18000 }
]
animal> db.domestic_animals.find({}, { animal_name: 1, expenses: 1, _id: 0 })
[
  { animal_name: 'Dog', expenses: 9000 },
  { animal_name: 'Cat', expenses: 7000 },
  { animal_name: 'Cow', expenses: 12000 },
  { animal_name: 'Goat', expenses: 6000 },
  { animal_name: 'Horse', expenses: 25000 }
]
animal>

```

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

```

mongosh mongodb://127.0.0.1:27017
{
  _id: ObjectId('69a075d7727e3eadb77c291a'),
  animal_name: 'Horse',
  gender: 'male',
  favorite_foods: [ 'hay', 'grains' ],
  animal_petsname: 'Blaze',
  life_span: 28,
  timestamp: ISODate('2016-04-30T00:00:00.000Z'),
  expenses: 25000
}
]
animal> db.wild_animals.find({}, { animal_name: 1, expenses: 1, _id: 0 })
[
  { animal_name: 'Red Panda', expenses: 22000 },
  { animal_name: 'Black Panther', expenses: 60000 },
  { animal_name: 'Giraffe', expenses: 35000 },
  { animal_name: 'Crocodile', expenses: 40000 },
  { animal_name: 'Zebra', expenses: 18000 }
]
animal> db.domestic_animals.find({}, { animal_name: 1, expenses: 1, _id: 0 })
[
  { animal_name: 'Dog', expenses: 9000 },
  { animal_name: 'Cat', expenses: 7000 },
  { animal_name: 'Cow', expenses: 12000 },
  { animal_name: 'Goat', expenses: 6000 },
  { animal_name: 'Horse', expenses: 25000 }
]
animal> db.domestic_animals.find({ life_span: 15 })
[
  {
    _id: ObjectId('69a075d7727e3eadb77c2917'),
    animal_name: 'Cat',
    gender: 'female',
    favorite_foods: [ 'fish', 'milk' ],
    animal_petsname: 'Luna',
    life_span: 15,
    timestamp: ISODate('2020-02-14T00:00:00.000Z'),
    expenses: 7000
  }
]
animal>

```

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

```

mongosh mongodb://127.0.0.1:27017
{
  animal_name: 'Cow',
  expenses: 12000
},
{
  animal_name: 'Goat',
  expenses: 6000
},
{
  animal_name: 'Horse',
  expenses: 25000
}
]
animal> db.domestic_animals.find({ life_span: 15 })
[
  {
    _id: ObjectId('69a075d7727e3eadb77c2917'),
    animal_name: 'Cat',
    gender: 'female',
    favorite_foods: ['fish', 'milk'],
    animal_petsname: 'Luna',
    life_span: 15,
    timestamp: ISODate('2020-02-14T00:00:00.000Z'),
    expenses: 7000
  }
]
animal> db.wild_animals.find({ care_taker_name: "Anil Kumar" })
[
  {
    _id: ObjectId('69a075a5727e3eadb77c2911'),
    animal_name: 'Red Panda',
    nature: 'harmless',
    favorite_foods: ['bamboo', 'fruits'],
    care_taker_name: 'Anil Kumar',
    life_span: 14,
    timestamp: ISODate('2020-05-15T00:00:00.000Z'),
    expenses: 22000
  },
  {
    _id: ObjectId('69a075a5727e3eadb77c2913'),
    animal_name: 'Giraffe',
    nature: 'harmless',
    favorite_foods: ['leaves'],
    care_taker_name: 'Anil Kumar',
    life_span: 25,
    timestamp: ISODate('2021-07-25T00:00:00.000Z'),
    expenses: 35000
  }
]
animal>

```

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

```

nature: 'harmless',
favorite_foods: ['leaves'],
care_taker_name: 'Anil Kumar',
life_span: 25,
timestamp: ISODate('2021-07-25T00:00:00.000Z'),
expenses: 35000
}
]
animal> db.wild_animals.find(
| { life_span: { $gt: 5 } },
| { animal_name: 1, favorite_foods: 1, expenses: 1, _id: 0 }
|
[
  {
    animal_name: 'Red Panda',
    favorite_foods: ['bamboo', 'fruits'],
    expenses: 22000
  },
  {
    animal_name: 'Black Panther',
    favorite_foods: ['meat', 'deer'],
    expenses: 60000
  },
  {
    animal_name: 'Giraffe',
    favorite_foods: ['leaves'],
    expenses: 35000
  },
  {
    animal_name: 'Crocodile',
    favorite_foods: ['fish', 'meat'],
    expenses: 40000
  },
  {
    animal_name: 'Zebra',
    favorite_foods: ['grass'],
    expenses: 18000
  }
]
animal> db.domestic_animals.find(
| { life_span: { $gt: 5 } },
| { animal_name: 1, favorite_foods: 1, expenses: 1, _id: 0 }
|
[

```

```
| mongosh mongodb://127.0.0.1:27017 | + - x
| expenses: 35000
|,
| {
|   animal_name: 'Crocodile',
|   favorite_foods: [ 'fish', 'meat' ],
|   expenses: 40000
| },
| {
|   animal_name: 'Zebra', favorite_foods: [ 'grass' ], expenses: 18000
| }
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', 'dog food' ],
  expenses: 9000
},
{
  animal_name: 'Cat',
  favorite_foods: [ 'fish', 'milk' ],
  expenses: 7000
},
{
  animal_name: 'Cow',
  favorite_foods: [ 'grass', 'fodder' ],
  expenses: 12000
},
{
  animal_name: 'Goat',
  favorite_foods: [ 'leaves', 'grass' ],
  expenses: 6000
},
{
  animal_name: 'Horse',
  favorite_foods: [ 'hay', 'grains' ],
  expenses: 25000
}
]
animal> |
```

26°C
Mostly clear

Search



ENG IN 22:07 26-02-2026