

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: Varun Kumar Tenali

Reg. no.: 23BCE9454

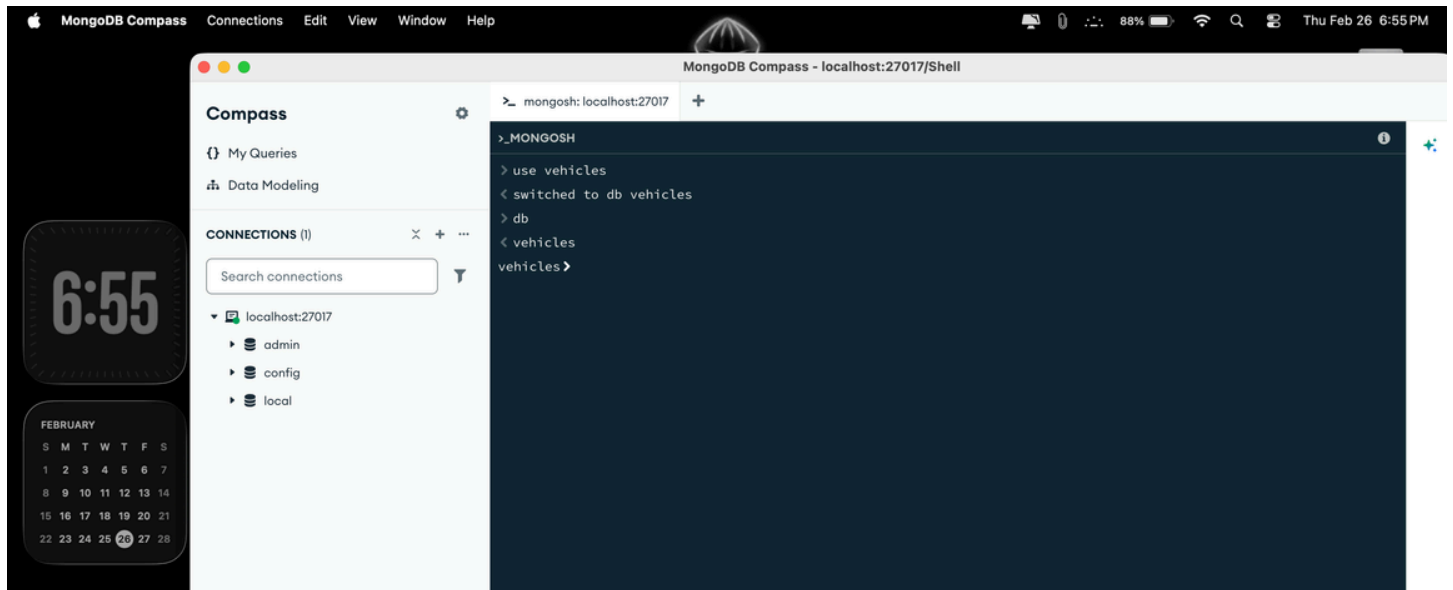


VIT-AP

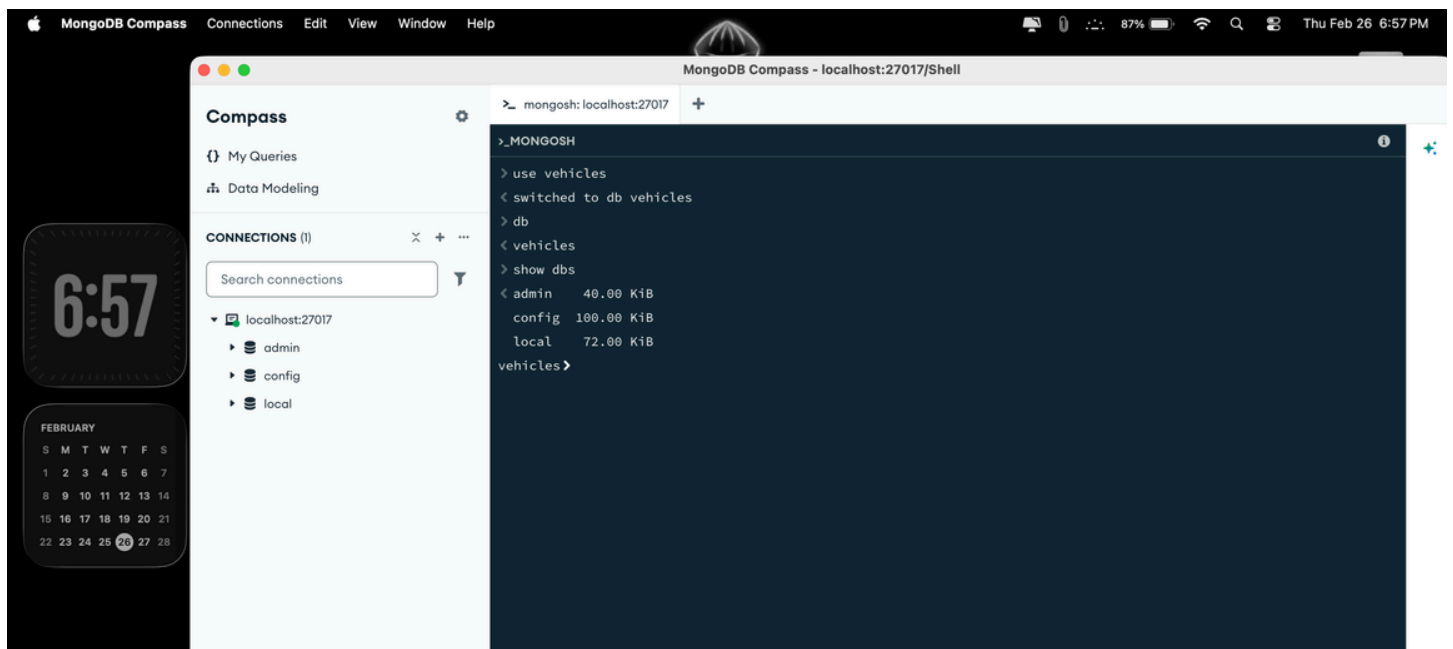
UNIVERSITY

1. Use MongoDB to implement the following DB operations :

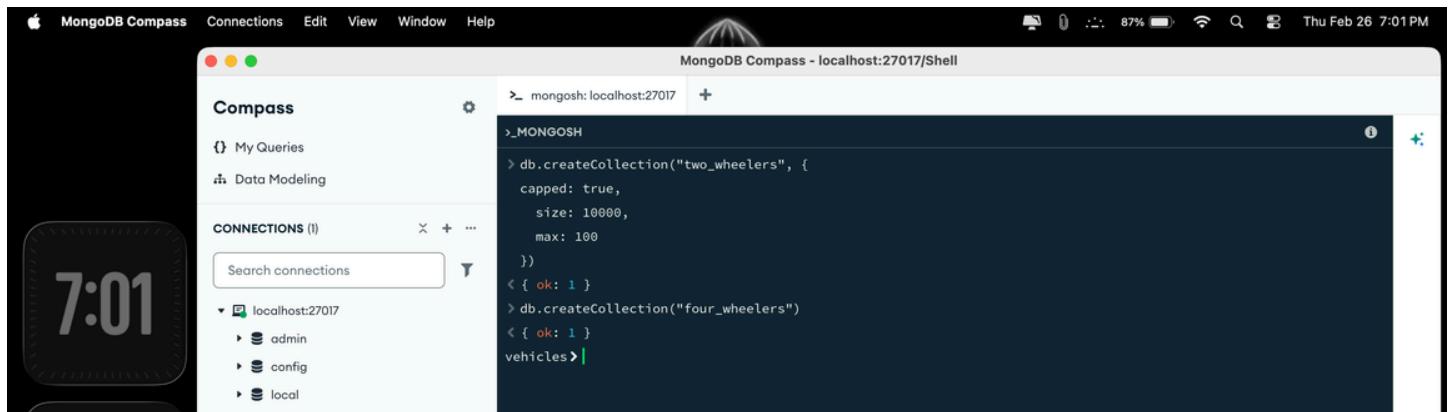
1) Create a database called 'vehicles' and write a MongoDB query to select database as "vehicles".



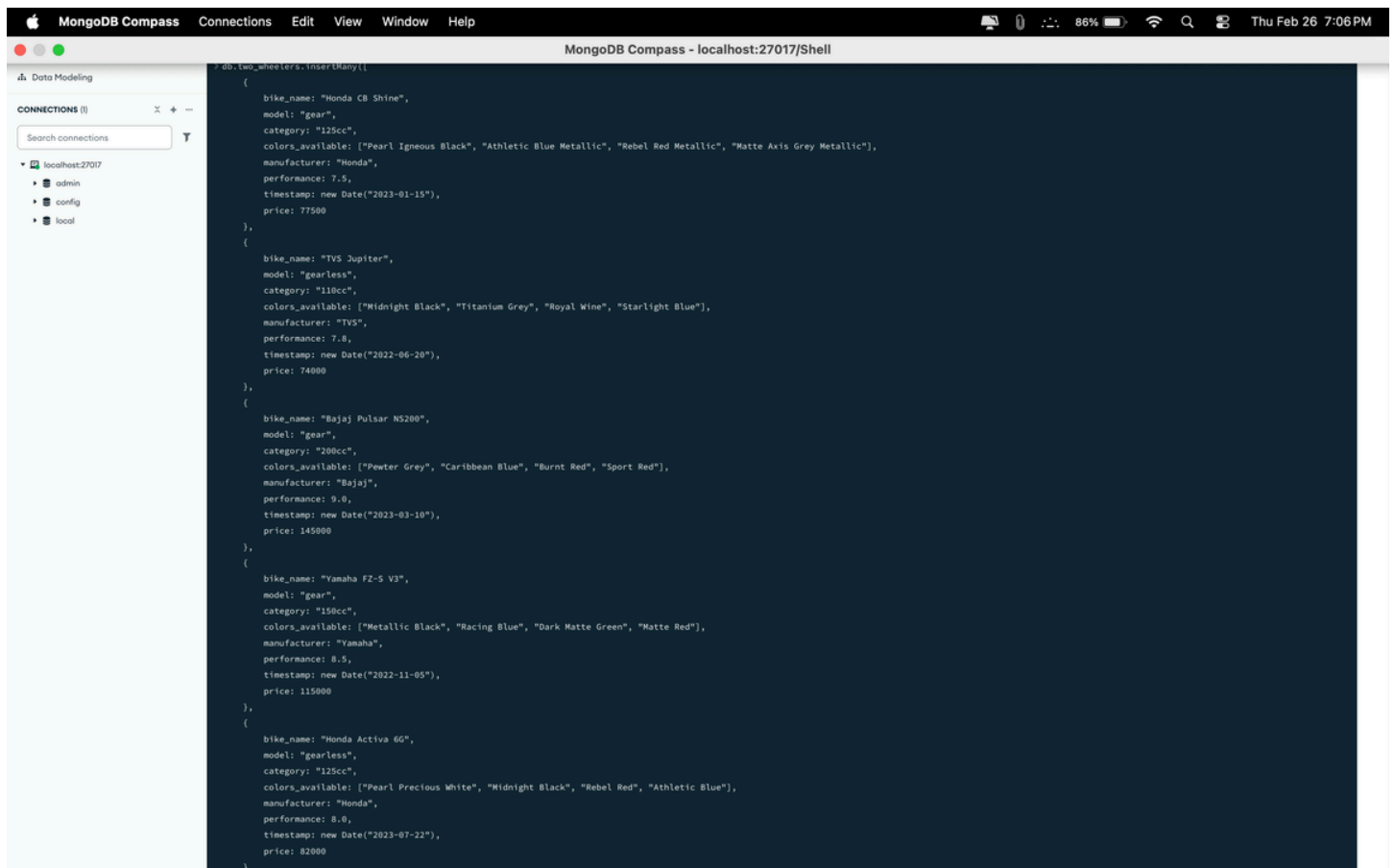
2) Write a MongoDB query to display all the databases.



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



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.



```
< {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('69a04b6ac53de4f649a5842a'),
    '1': ObjectId('69a04b6ac53de4f649a5842b'),
    '2': ObjectId('69a04b6ac53de4f649a5842c'),
    '3': ObjectId('69a04b6ac53de4f649a5842d'),
    '4': ObjectId('69a04b6ac53de4f649a5842e')
  }
}
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.

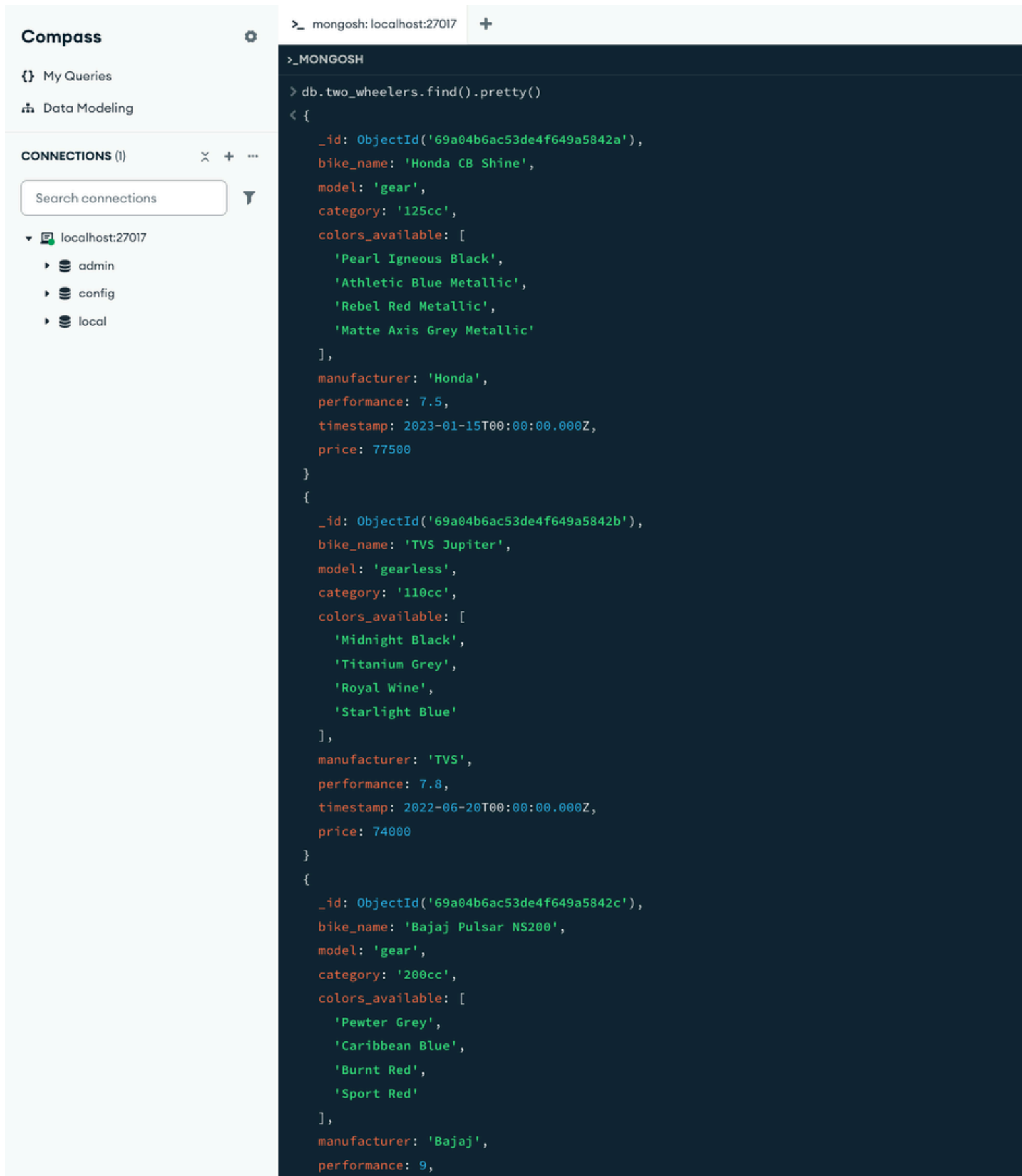
```
db.four_wheelers.insertMany([
  {
    vehicle_name: "Rolls-Royce Silver Shadow",
    model: "own",
    category: "car",
    variants: ["Standard Saloon", "Long Wheelbase", "Two-Door Saloon", "Drophead Coupe"],
    manufacturer: "Rolls-Royce",
    performance: 9.5,
    timestamp: new Date("1965-10-01"),
    price: 35000000
  },
  {
    vehicle_name: "Bentley Continental",
    model: "own",
    category: "car",
    variants: ["GT V8", "GT W12", "GT Speed", "GTC Convertible V8", "GTC Convertible W12"],
    manufacturer: "Bentley",
    performance: 9.7,
    timestamp: new Date("1952-01-01"),
    price: 42000000
  },
  {
    vehicle_name: "Aston Martin DB5",
    model: "own",
    category: "car",
    variants: ["Standard Coupe", "Vantage Coupe", "Convertible", "Shooting Brake"],
    manufacturer: "Aston Martin",
    performance: 9.4,
    timestamp: new Date("1963-07-01"),
    price: 55000000
  },
  {
    vehicle_name: "Porsche 911 Air-Cooled",
    model: "own",
    category: "car",
    variants: ["911 Carrera", "911 Targa", "911 Turbo", "911 RS", "911 Cabriolet"],
    manufacturer: "Porsche",
    performance: 9.6,
    timestamp: new Date("1963-09-12"),
    price: 38000000
  },
  {
    vehicle_name: "Mercedes-Benz 300SL Gullwing",
    model: "own",
    category: "car",
    variants: ["Coupe Gullwing", "Roadster", "Racing Specification"],
    manufacturer: "Mercedes-Benz",
    performance: 9.8,
    timestamp: new Date("1954-02-06"),
    price: 95000000
  }
])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("69a04c89c53de4f649a5842f"),
    '1': ObjectId("69a04c89c53de4f649a58430"),
    '2': ObjectId("69a04c89c53de4f649a58431"),
    '3': ObjectId("69a04c89c53de4f649a58432"),
    '4': ObjectId("69a04c89c53de4f649a58433")
  }
}
```

vehicles >

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

→ Documents from two_wheelers :



The screenshot shows the MongoDB Compass interface. On the left, the 'CONNECTIONS (1)' panel shows a connection to 'localhost:27017' with databases 'admin', 'config', and 'local'. The main panel displays a MongoDB query and its results.

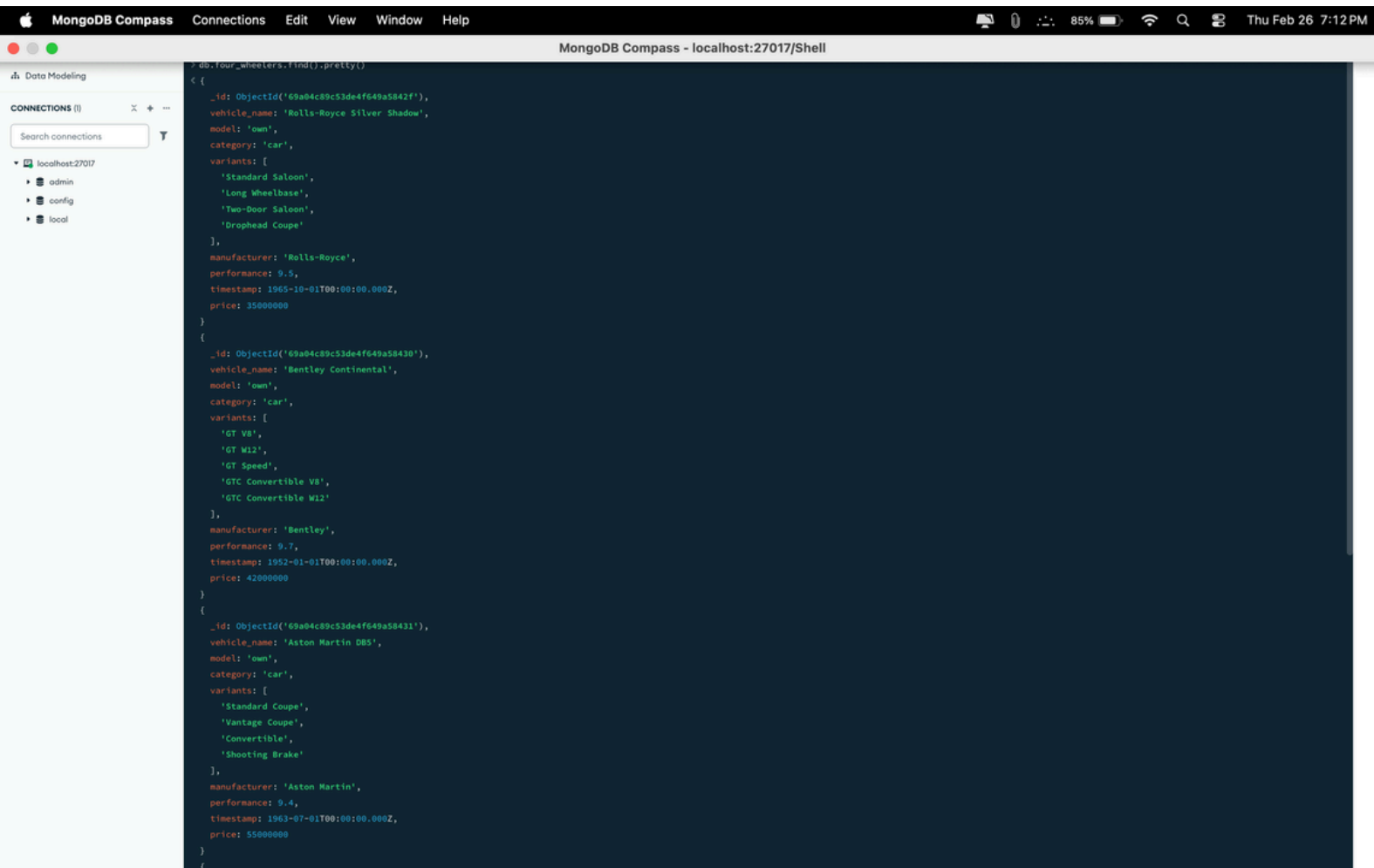
Query:

```
> db.two_wheelers.find().pretty()
```

Results:

```
< {
  _id: ObjectId('69a04b6ac53de4f649a5842a'),
  bike_name: 'Honda CB Shine',
  model: 'gear',
  category: '125cc',
  colors_available: [
    'Pearl Igneous Black',
    'Athletic Blue Metallic',
    'Rebel Red Metallic',
    'Matte Axis Grey Metallic'
  ],
  manufacturer: 'Honda',
  performance: 7.5,
  timestamp: 2023-01-15T00:00:00.000Z,
  price: 77500
}
{
  _id: ObjectId('69a04b6ac53de4f649a5842b'),
  bike_name: 'TVS Jupiter',
  model: 'gearless',
  category: '110cc',
  colors_available: [
    'Midnight Black',
    'Titanium Grey',
    'Royal Wine',
    'Starlight Blue'
  ],
  manufacturer: 'TVS',
  performance: 7.8,
  timestamp: 2022-06-20T00:00:00.000Z,
  price: 74000
}
{
  _id: ObjectId('69a04b6ac53de4f649a5842c'),
  bike_name: 'Bajaj Pulsar NS200',
  model: 'gear',
  category: '200cc',
  colors_available: [
    'Pewter Grey',
    'Caribbean Blue',
    'Burnt Red',
    'Sport Red'
  ],
  manufacturer: 'Bajaj',
  performance: 9,
```

→ Documents from four.wheelers :

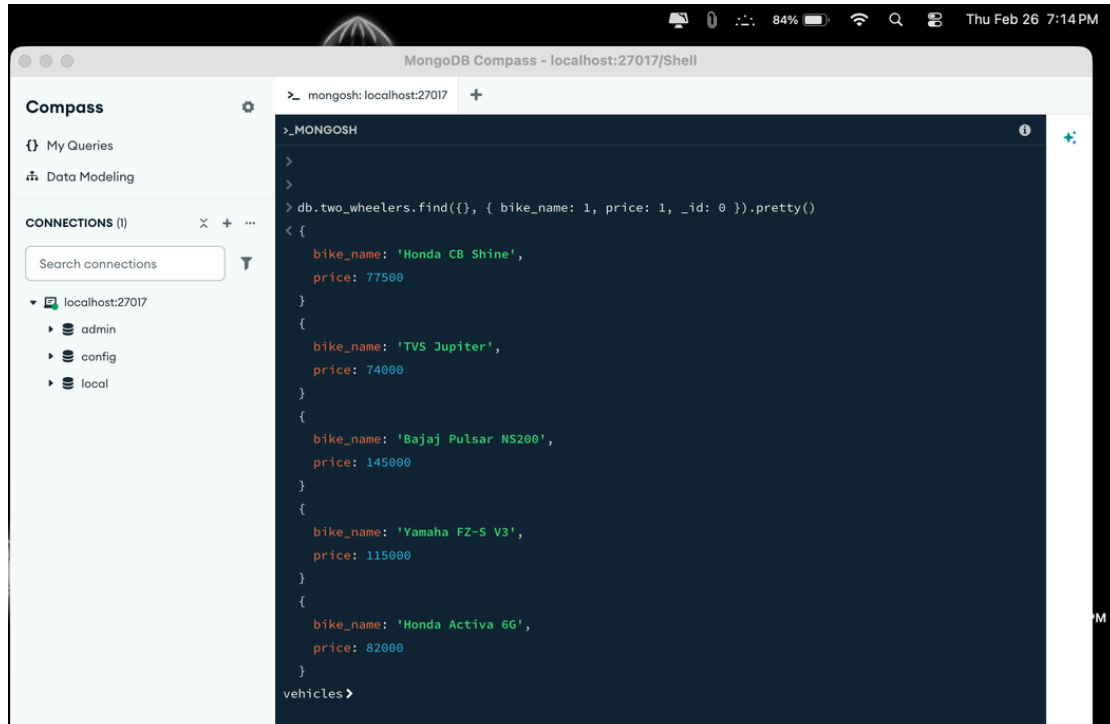


The screenshot shows the MongoDB Compass interface. The left sidebar displays the 'CONNECTIONS' panel with a search bar and a tree view showing the 'localhost:27017' connection, including databases 'admin', 'config', and 'local'. The main panel shows the 'Data Modeling' view with a query editor containing the command `db.four_wheelers.find().pretty()`. The results pane displays three documents from the `four.wheelers` collection, each with fields for `_id`, `vehicle_name`, `model`, `category`, `variants`, `manufacturer`, `performance`, `timestamp`, and `price`.

```
db.four_wheelers.find().pretty()
{
  "_id": ObjectId("69a04c89c53de4f649a5842f"),
  "vehicle_name": "Rolls-Royce Silver Shadow",
  "model": "own",
  "category": "car",
  "variants": [
    "Standard Saloon",
    "Long Wheelbase",
    "Two-Door Saloon",
    "Drophead Coupe"
  ],
  "manufacturer": "Rolls-Royce",
  "performance": 9.5,
  "timestamp": 1965-10-01T00:00:00.000Z,
  "price": 35000000
}
{
  "_id": ObjectId("69a04c89c53de4f649a58430"),
  "vehicle_name": "Bentley Continental",
  "model": "own",
  "category": "car",
  "variants": [
    "GT V8",
    "GT W12",
    "GT Speed",
    "GTC Convertible V8",
    "GTC Convertible W12"
  ],
  "manufacturer": "Bentley",
  "performance": 9.7,
  "timestamp": 1952-01-01T00:00:00.000Z,
  "price": 42000000
}
{
  "_id": ObjectId("69a04c89c53de4f649a58431"),
  "vehicle_name": "Aston Martin DB5",
  "model": "own",
  "category": "car",
  "variants": [
    "Standard Coupe",
    "Vantage Coupe",
    "Convertible",
    "Shooting Brake"
  ],
  "manufacturer": "Aston Martin",
  "performance": 9.4,
  "timestamp": 1963-07-01T00:00:00.000Z,
  "price": 55000000
}
```

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

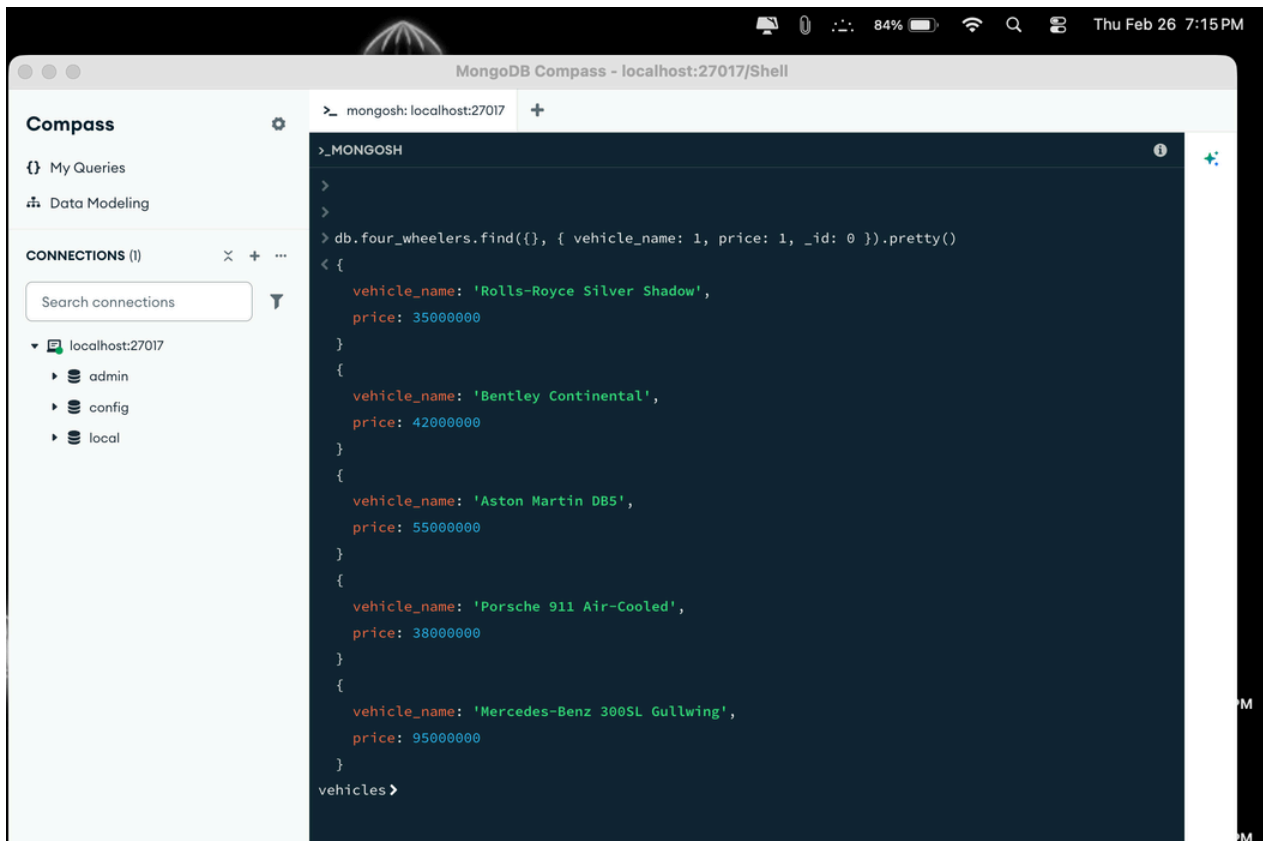
→ From two.wheelers :



The screenshot shows the MongoDB Compass interface. On the left, the 'CONNECTIONS' panel lists 'localhost:27017' with databases 'admin', 'config', and 'local'. The main panel shows a MongoDB shell session with the following query and result:

```
>_MONGOSH
>
>
> db.two_wheelers.find({}, { bike_name: 1, price: 1, _id: 0 }).pretty()
< {
  bike_name: 'Honda CB Shine',
  price: 77500
}
{
  bike_name: 'TVS Jupiter',
  price: 74000
}
{
  bike_name: 'Bajaj Pulsar NS200',
  price: 145000
}
{
  bike_name: 'Yamaha FZ-S V3',
  price: 115000
}
{
  bike_name: 'Honda Activa 6G',
  price: 82000
}
}
vehicles>
```

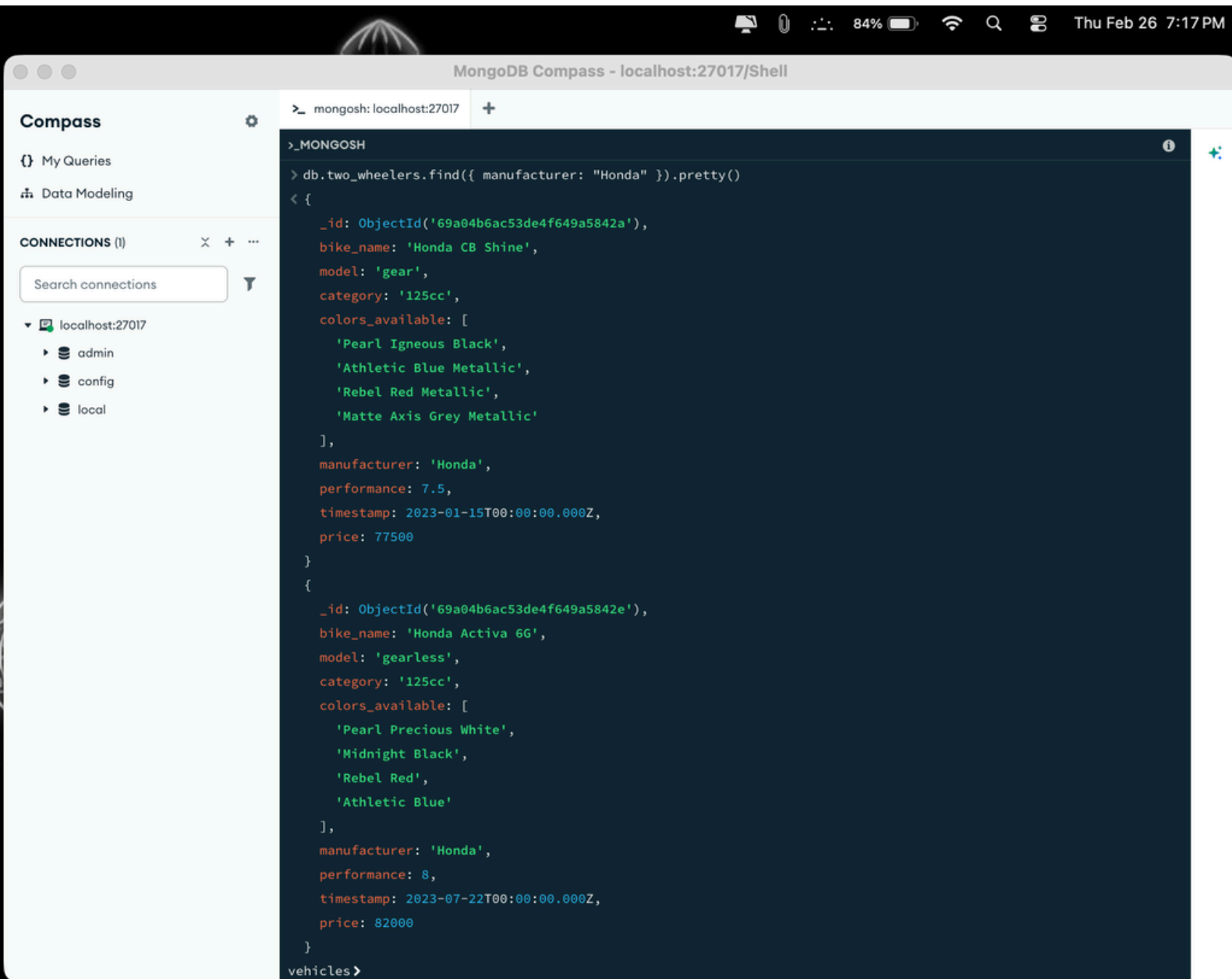
→ From four.wheelers :



The screenshot shows the MongoDB Compass interface. On the left, the 'CONNECTIONS' panel lists 'localhost:27017' with databases 'admin', 'config', and 'local'. The main panel shows a MongoDB shell session with the following query and result:

```
>_MONGOSH
>
>
> db.four_wheelers.find({}, { vehicle_name: 1, price: 1, _id: 0 }).pretty()
< {
  vehicle_name: 'Rolls-Royce Silver Shadow',
  price: 35000000
}
{
  vehicle_name: 'Bentley Continental',
  price: 42000000
}
{
  vehicle_name: 'Aston Martin DB5',
  price: 55000000
}
{
  vehicle_name: 'Porsche 911 Air-Cooled',
  price: 38000000
}
{
  vehicle_name: 'Mercedes-Benz 300SL Gullwing',
  price: 95000000
}
}
vehicles>
```

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



The screenshot shows the MongoDB Compass application running on a Mac. The title bar indicates 'MongoDB Compass - localhost:27017/Shell'. The interface is divided into three main sections: a left sidebar, a top toolbar, and a main workspace.

Left Sidebar:

- Compass** (selected)
- My Queries
- Data Modeling
- CONNECTIONS (1)**
 - localhost:27017
 - admin
 - config
 - local

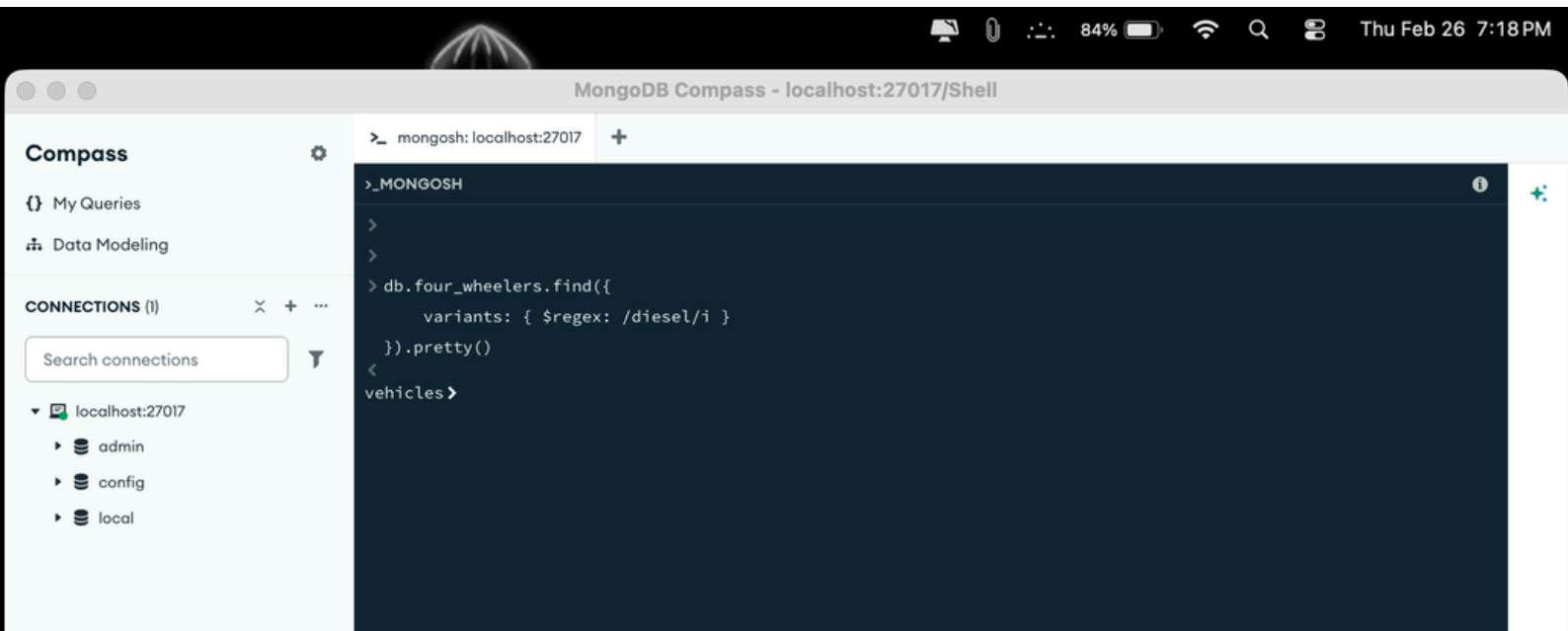
Top Toolbar: Includes icons for a terminal, a document, a search icon, and a refresh icon. The system status bar at the top right shows 'Thu Feb 26 7:17 PM' and '84%' battery.

Main Workspace:

The workspace is titled 'mongosh: localhost:27017'. It contains a terminal window with the following content:

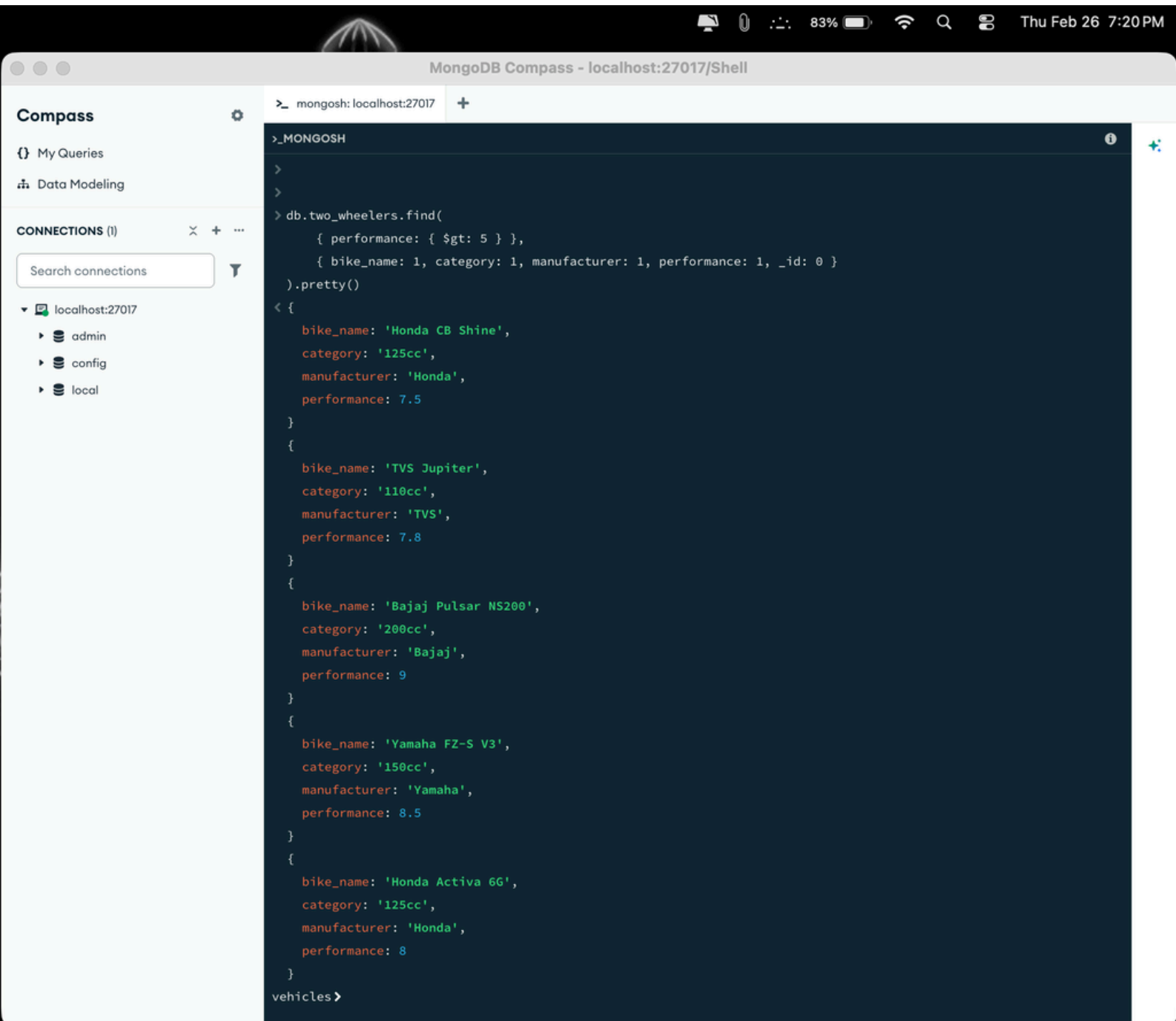
```
>_MONGOSH
> db.two_wheelers.find({ manufacturer: "Honda" }).pretty()
< {
  _id: ObjectId('69a04b6ac53de4f649a5842a'),
  bike_name: 'Honda CB Shine',
  model: 'gear',
  category: '125cc',
  colors_available: [
    'Pearl Igneous Black',
    'Athletic Blue Metallic',
    'Rebel Red Metallic',
    'Matte Axis Grey Metallic'
  ],
  manufacturer: 'Honda',
  performance: 7.5,
  timestamp: 2023-01-15T00:00:00.000Z,
  price: 77500
}
{
  _id: ObjectId('69a04b6ac53de4f649a5842e'),
  bike_name: 'Honda Activa 6G',
  model: 'gearless',
  category: '125cc',
  colors_available: [
    'Pearl Precious White',
    'Midnight Black',
    'Rebel Red',
    'Athletic Blue'
  ],
  manufacturer: 'Honda',
  performance: 8,
  timestamp: 2023-07-22T00:00:00.000Z,
  price: 82000
}
vehicles >
```


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



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

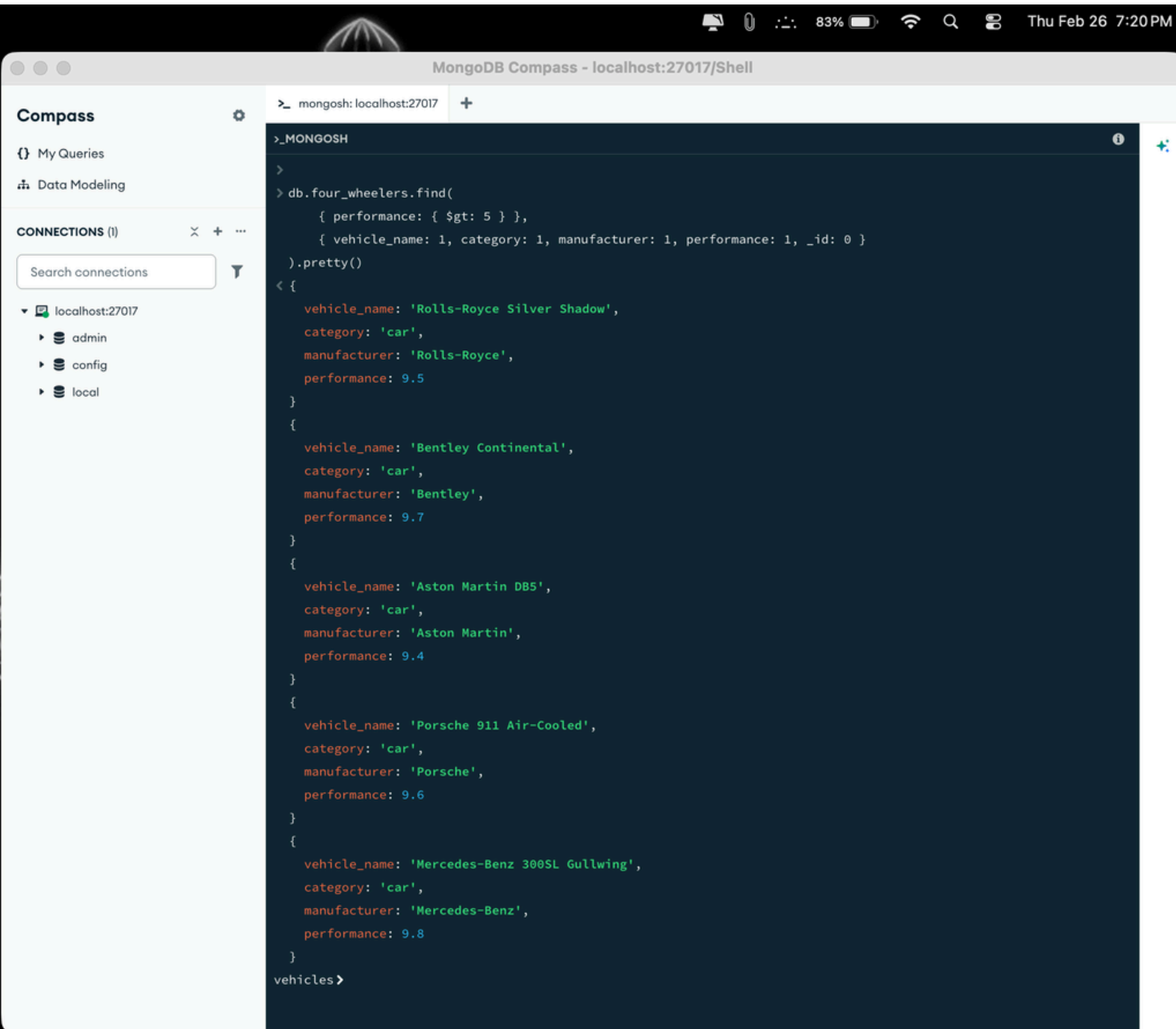
→ From two wheelers :



The screenshot shows the MongoDB Compass application interface. The title bar reads "MongoDB Compass - localhost:27017/Shell". The left sidebar contains a "Compass" header, "My Queries" (empty), "Data Modeling", and "CONNECTIONS (1)". Under "CONNECTIONS", there is a search bar and a list of connections: "localhost:27017" (expanded) showing "admin", "config", and "local" databases. The main panel shows a MongoDB shell session with the following command and output:

```
>_MONGOSH
>
>
> db.two_wheelers.find(
  { performance: { $gt: 5 } },
  { bike_name: 1, category: 1, manufacturer: 1, performance: 1, _id: 0 }
).pretty()
< {
  bike_name: 'Honda CB Shine',
  category: '125cc',
  manufacturer: 'Honda',
  performance: 7.5
}
{
  bike_name: 'TVS Jupiter',
  category: '110cc',
  manufacturer: 'TVS',
  performance: 7.8
}
{
  bike_name: 'Bajaj Pulsar NS200',
  category: '200cc',
  manufacturer: 'Bajaj',
  performance: 9
}
{
  bike_name: 'Yamaha FZ-S V3',
  category: '150cc',
  manufacturer: 'Yamaha',
  performance: 8.5
}
{
  bike_name: 'Honda Activa 6G',
  category: '125cc',
  manufacturer: 'Honda',
  performance: 8
}
vehicles>
```

→ From four wheelers :

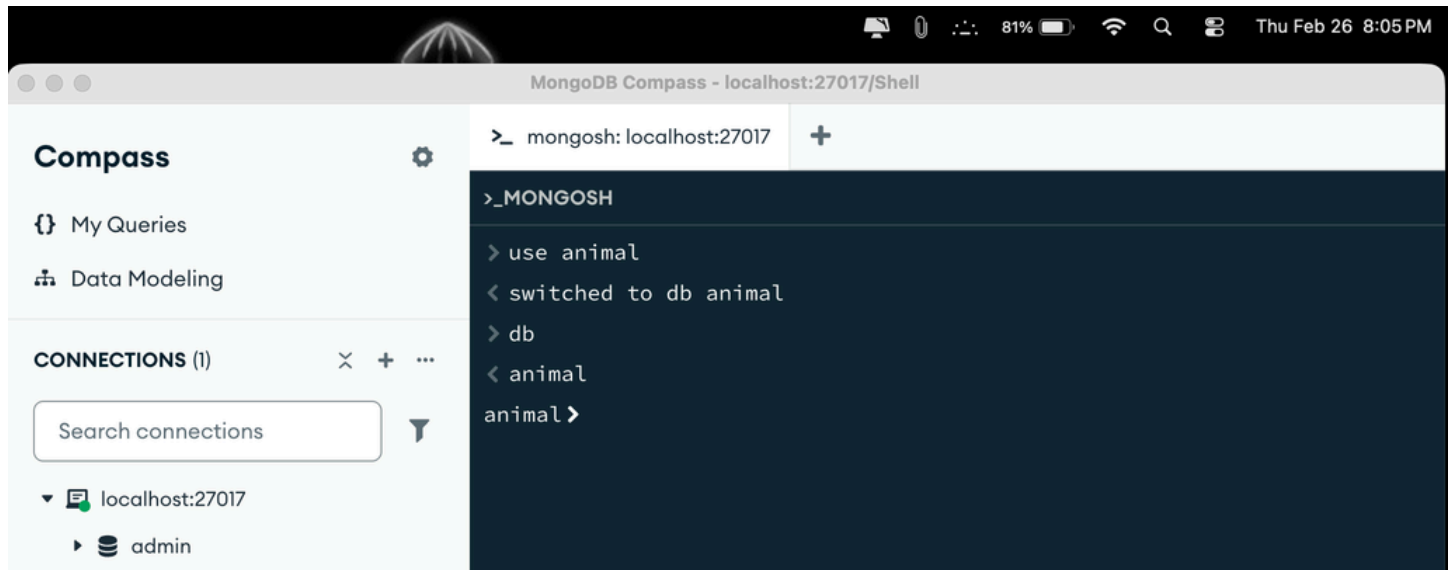


The screenshot shows the MongoDB Compass application running on a Mac. The title bar indicates the connection is to 'localhost:27017/Shell'. The left sidebar contains navigation options: 'My Queries', 'Data Modeling', and 'CONNECTIONS (1)'. Under 'CONNECTIONS (1)', there is a search bar and a list of connections: 'localhost:27017', 'admin', 'config', and 'local'. The main panel displays a MongoDB shell session with the following commands and results:

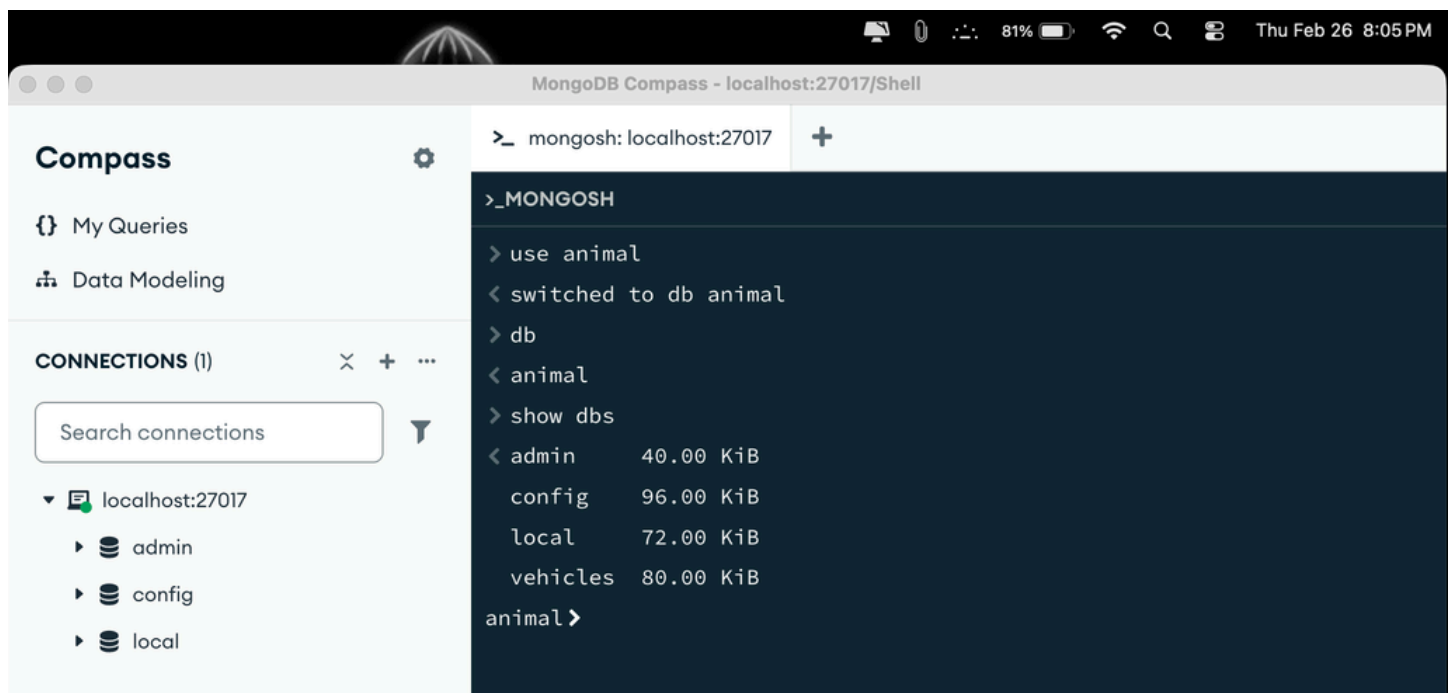
```
>_ mongosh: localhost:27017 +
>_MONGOSH
>
> db.four_wheelers.find(
  { performance: { $gt: 5 } },
  { vehicle_name: 1, category: 1, manufacturer: 1, performance: 1, _id: 0 }
).pretty()
< {
  vehicle_name: 'Rolls-Royce Silver Shadow',
  category: 'car',
  manufacturer: 'Rolls-Royce',
  performance: 9.5
}
{
  vehicle_name: 'Bentley Continental',
  category: 'car',
  manufacturer: 'Bentley',
  performance: 9.7
}
{
  vehicle_name: 'Aston Martin DB5',
  category: 'car',
  manufacturer: 'Aston Martin',
  performance: 9.4
}
{
  vehicle_name: 'Porsche 911 Air-Cooled',
  category: 'car',
  manufacturer: 'Porsche',
  performance: 9.6
}
{
  vehicle_name: 'Mercedes-Benz 300SL Gullwing',
  category: 'car',
  manufacturer: 'Mercedes-Benz',
  performance: 9.8
}
vehicles>
```

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

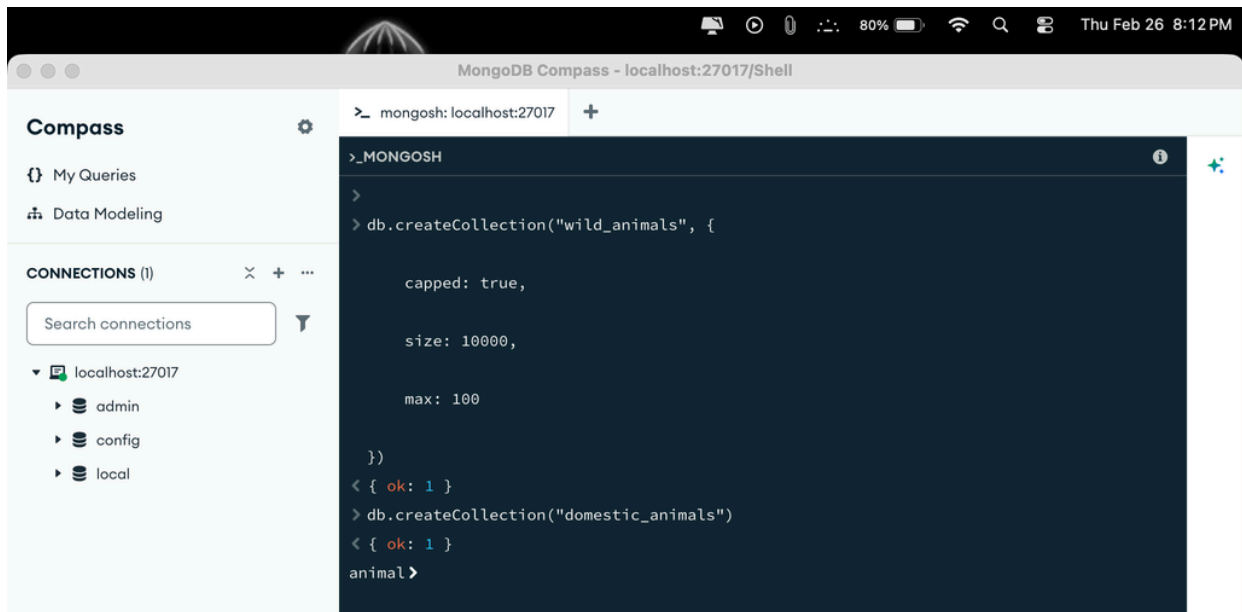
1) Create a database called 'animal' and write a MongoDB query to select database as "animal".



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



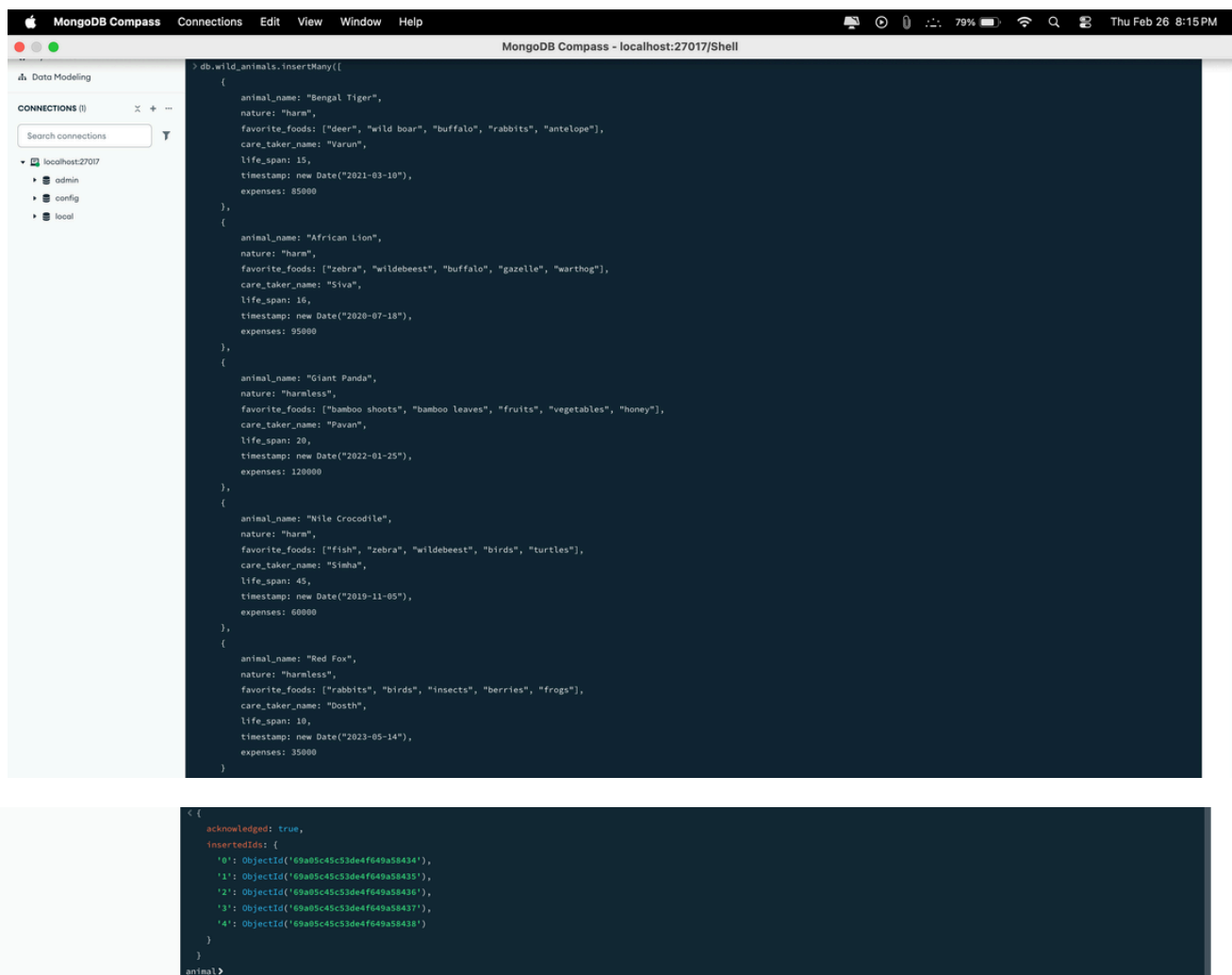
3) Create a collection called 'wild_animals'. (use capping) and Create a collection called 'domestic_animals'.



The screenshot shows the MongoDB Compass interface. On the left, the 'CONNECTIONS' panel shows a connection to 'localhost:27017' with databases 'admin', 'config', and 'local'. The main panel shows the MongoDB Shell with the following commands and output:

```
>_MONGOSH
>
> db.createCollection("wild_animals", {
  capped: true,
  size: 10000,
  max: 100
})
< { ok: 1 }
> 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.



The screenshot shows the MongoDB Compass interface. On the left, the 'CONNECTIONS' panel shows a connection to 'localhost:27017' with databases 'admin', 'config', and 'local'. The main panel shows the MongoDB Shell with the following commands and output:

```
> db.wild_animals.insertMany([
  {
    animal_name: "Bengal Tiger",
    nature: "harm",
    favorite_foods: ["deer", "wild boar", "buffalo", "rabbits", "antelope"],
    care_taker_name: "Varun",
    life_span: 15,
    timestamp: new Date("2021-03-10"),
    expenses: 85000
  },
  {
    animal_name: "African Lion",
    nature: "harm",
    favorite_foods: ["zebra", "wildebeest", "buffalo", "gazelle", "warthog"],
    care_taker_name: "Siva",
    life_span: 16,
    timestamp: new Date("2020-07-18"),
    expenses: 95000
  },
  {
    animal_name: "Giant Panda",
    nature: "harmless",
    favorite_foods: ["bamboo shoots", "bamboo leaves", "fruits", "vegetables", "honey"],
    care_taker_name: "Pavan",
    life_span: 20,
    timestamp: new Date("2022-01-25"),
    expenses: 120000
  },
  {
    animal_name: "Nile Crocodile",
    nature: "harm",
    favorite_foods: ["fish", "zebra", "wildebeest", "birds", "turtles"],
    care_taker_name: "Simha",
    life_span: 45,
    timestamp: new Date("2019-11-05"),
    expenses: 60000
  },
  {
    animal_name: "Red Fox",
    nature: "harmless",
    favorite_foods: ["rabbits", "birds", "insects", "berries", "frogs"],
    care_taker_name: "Dosth",
    life_span: 10,
    timestamp: new Date("2023-05-14"),
    expenses: 35000
  }
])
< {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('69a05c45c53de4f649a58434'),
    '1': ObjectId('69a05c45c53de4f649a58435'),
    '2': ObjectId('69a05c45c53de4f649a58436'),
    '3': ObjectId('69a05c45c53de4f649a58437'),
    '4': ObjectId('69a05c45c53de4f649a58438')
  }
}
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.

The screenshot shows the MongoDB Compass application interface. The top menu bar includes 'MongoDB Compass', 'Connections', 'Edit', 'View', 'Window', and 'Help'. The status bar at the top right shows 'Thu Feb 26 8:17 PM' and a battery level of 79%.

On the left sidebar, the 'CONNECTIONS (1)' section is expanded, showing a connection to 'localhost:27017'. Under this connection, there are three databases listed: 'admin', 'config', and 'local'.

The main workspace displays the MongoDB Shell with the following command and its output:

```
animal> db.domestic_animals.insertMany([
  {
    animal_name: "Dog",
    gender: "male",
    favorite_foods: ["meat", "bones", "dog biscuits", "milk", "rice"],
    animal_petname: "Bruno",
    life_span: 13,
    timestamp: new Date("2021-04-12"),
    expenses: 15000
  },
  {
    animal_name: "Cat",
    gender: "female",
    favorite_foods: ["fish", "milk", "chicken", "cat food", "tuna"],
    animal_petname: "Kitty",
    life_span: 15,
    timestamp: new Date("2022-02-20"),
    expenses: 12000
  },
  {
    animal_name: "Cow",
    gender: "female",
    favorite_foods: ["grass", "hay", "wheat", "corn", "vegetables"],
    animal_petname: "Ganga",
    life_span: 20,
    timestamp: new Date("2020-08-15"),
    expenses: 25000
  },
  {
    animal_name: "Rabbit",
    gender: "male",
    favorite_foods: ["carrots", "lettuce", "cabbage", "hay", "apples"],
    animal_petname: "Fluffy",
    life_span: 8,
    timestamp: new Date("2023-01-10"),
    expenses: 8000
  },
  {
    animal_name: "Parrot",
    gender: "male",
    favorite_foods: ["seeds", "fruits", "nuts", "vegetables", "berries"],
    animal_petname: "Mittu",
    life_span: 25,
    timestamp: new Date("2022-09-05"),
    expenses: 10000
  }
])
```

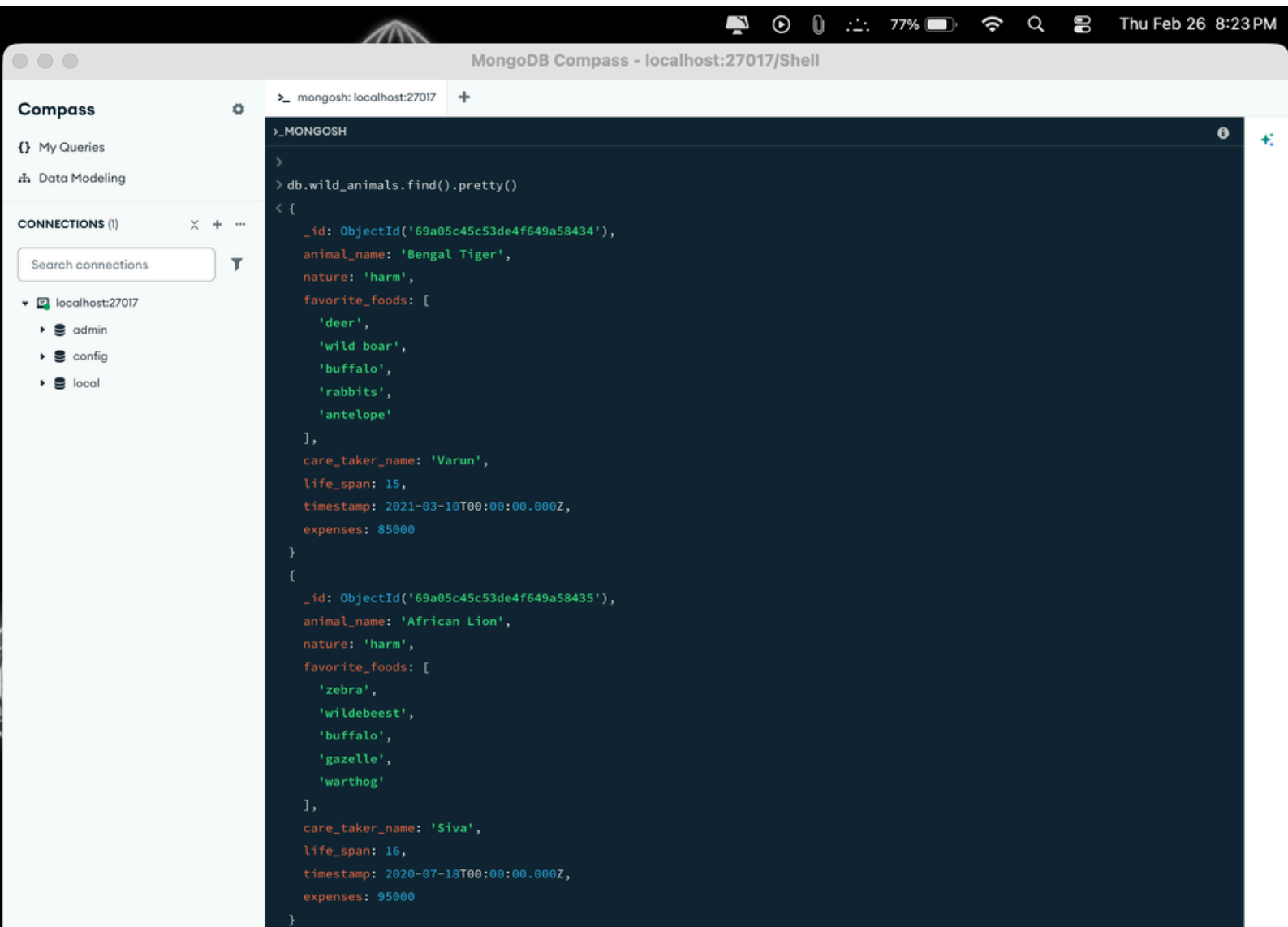
The output of the command is displayed below the shell prompt:

```
< {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('69a05cfc53de4f649a58439'),
    '1': ObjectId('69a05cfc53de4f649a5843a'),
    '2': ObjectId('69a05cfc53de4f649a5843b'),
    '3': ObjectId('69a05cfc53de4f649a5843c'),
    '4': ObjectId('69a05cfc53de4f649a5843d')
  }
}
```

The prompt 'animal>' is visible at the bottom of the shell window.

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

→ Documents from wild animals :



The screenshot shows the MongoDB Compass interface. The left sidebar contains 'My Queries', 'Data Modeling', and 'CONNECTIONS (1)'. The 'CONNECTIONS' section shows a connection to 'localhost:27017' with databases 'admin', 'config', and 'local'. The main panel displays a MongoDB shell session with the following query and results:

```
> mongosh: localhost:27017
> db.wild_animals.find().pretty()
< {
  _id: ObjectId('69a05c45c53de4f649a58434'),
  animal_name: 'Bengal Tiger',
  nature: 'harm',
  favorite_foods: [
    'deer',
    'wild boar',
    'buffalo',
    'rabbits',
    'antelope'
  ],
  care_taker_name: 'Varun',
  life_span: 15,
  timestamp: 2021-03-10T00:00:00.000Z,
  expenses: 85000
}
{
  _id: ObjectId('69a05c45c53de4f649a58435'),
  animal_name: 'African Lion',
  nature: 'harm',
  favorite_foods: [
    'zebra',
    'wildebeest',
    'buffalo',
    'gazelle',
    'warthog'
  ],
  care_taker_name: 'Siva',
  life_span: 16,
  timestamp: 2020-07-18T00:00:00.000Z,
  expenses: 95000
}
```

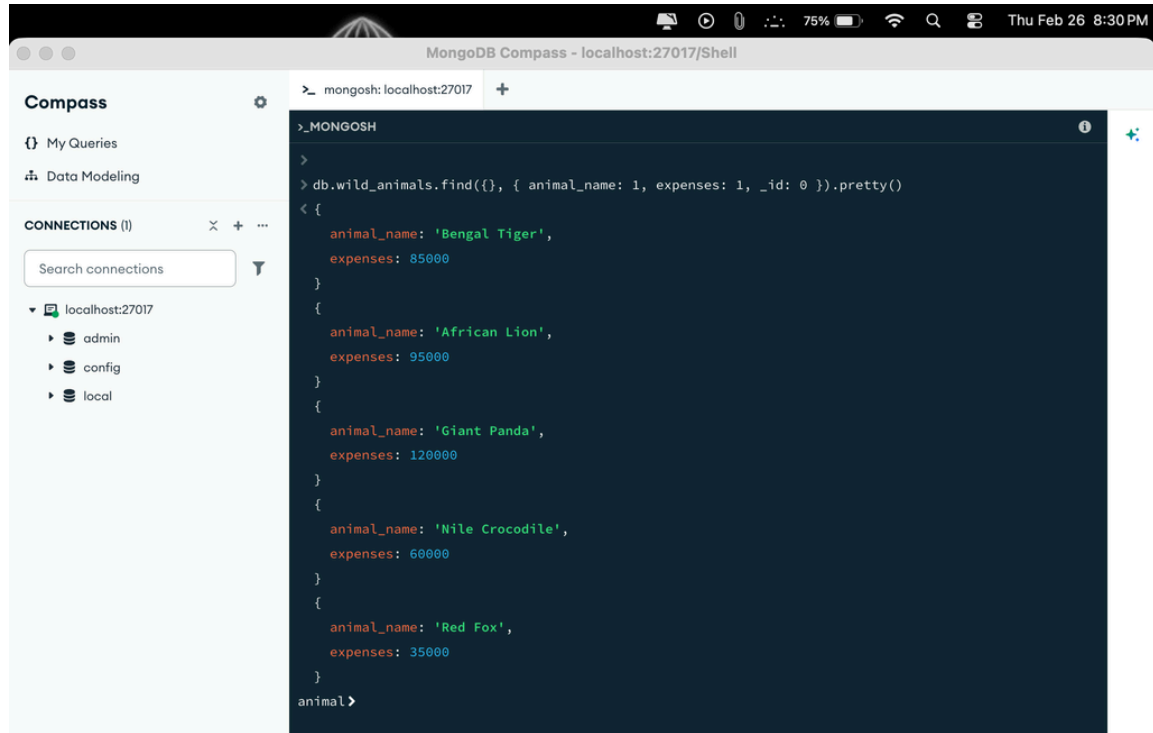
→ Documents from domestic_animals :

The screenshot shows the MongoDB Compass interface. The left sidebar contains 'My Queries', 'Data Modeling', and 'CONNECTIONS (1)'. The 'CONNECTIONS' section shows a connection to 'localhost:27017' with databases 'admin', 'config', and 'local'. The main panel displays the MongoDB Shell with the command `db.domestic_animals.find().pretty()` and its output. The output shows three documents: a Dog named Bruno, a Cat named Kitty, and a Cow named Ganga.

```
>_MONGOSH
> db.domestic_animals.find().pretty()
< {
  _id: ObjectId('69a05cf1c53de4f649a58439'),
  animal_name: 'Dog',
  gender: 'male',
  favorite_foods: [
    'meat',
    'bones',
    'dog biscuits',
    'milk',
    'rice'
  ],
  animal_petname: 'Bruno',
  life_span: 13,
  timestamp: 2021-04-12T00:00:00.000Z,
  expenses: 15000
}
{
  _id: ObjectId('69a05cf1c53de4f649a5843a'),
  animal_name: 'Cat',
  gender: 'female',
  favorite_foods: [
    'fish',
    'milk',
    'chicken',
    'cat food',
    'tuna'
  ],
  animal_petname: 'Kitty',
  life_span: 15,
  timestamp: 2022-02-20T00:00:00.000Z,
  expenses: 12000
}
{
  _id: ObjectId('69a05cf1c53de4f649a5843b'),
  animal_name: 'Cow',
  gender: 'female',
  favorite_foods: [
    'grass',
    'hay',
    'wheat',
    'corn',
    'vegetables'
  ],
  animal_petname: 'Ganga',
  life_span: 20,
  timestamp: 2020-08-15T00:00:00.000Z,
```


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

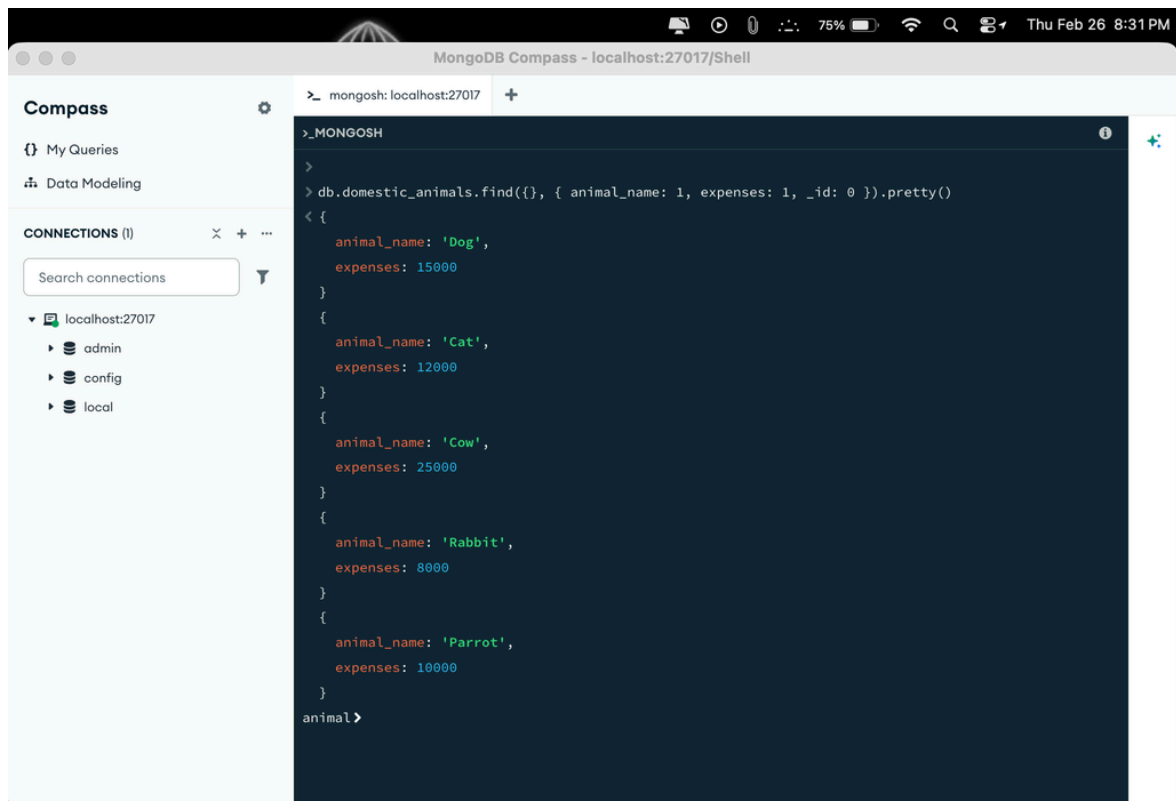
→ From wild animals:



The screenshot shows the MongoDB Compass interface. The left sidebar displays the 'Connections' panel with a search bar and a list of connections: 'localhost:27017' (expanded), 'admin', 'config', and 'local'. The main panel shows the 'MongoShell' tab with the following query and results:

```
>_MONGOSH
>
> db.wild_animals.find({}, { animal_name: 1, expenses: 1, _id: 0 }).pretty()
< {
  animal_name: 'Bengal Tiger',
  expenses: 85000
}
{
  animal_name: 'African Lion',
  expenses: 95000
}
{
  animal_name: 'Giant Panda',
  expenses: 120000
}
{
  animal_name: 'Nile Crocodile',
  expenses: 60000
}
{
  animal_name: 'Red Fox',
  expenses: 35000
}
}
```

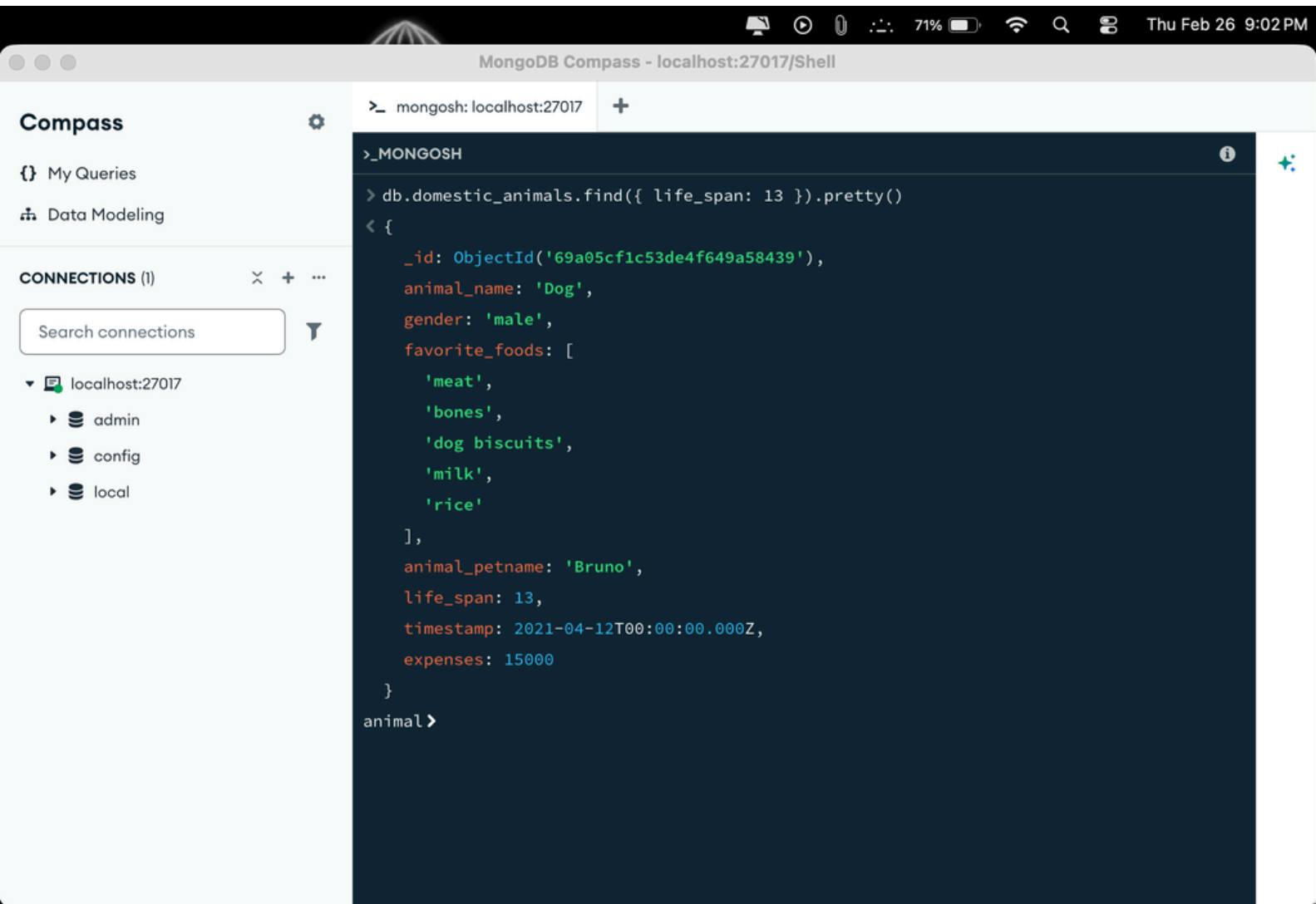
→ From domestic animals:



The screenshot shows the MongoDB Compass interface. The left sidebar displays the 'Connections' panel with a search bar and a list of connections: 'localhost:27017' (expanded), 'admin', 'config', and 'local'. The main panel shows the 'MongoShell' tab with the following query and results:

```
>_MONGOSH
>
> db.domestic_animals.find({}, { animal_name: 1, expenses: 1, _id: 0 }).pretty()
< {
  animal_name: 'Dog',
  expenses: 15000
}
{
  animal_name: 'Cat',
  expenses: 12000
}
{
  animal_name: 'Cow',
  expenses: 25000
}
{
  animal_name: 'Rabbit',
  expenses: 8000
}
{
  animal_name: 'Parrot',
  expenses: 10000
}
}
```

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



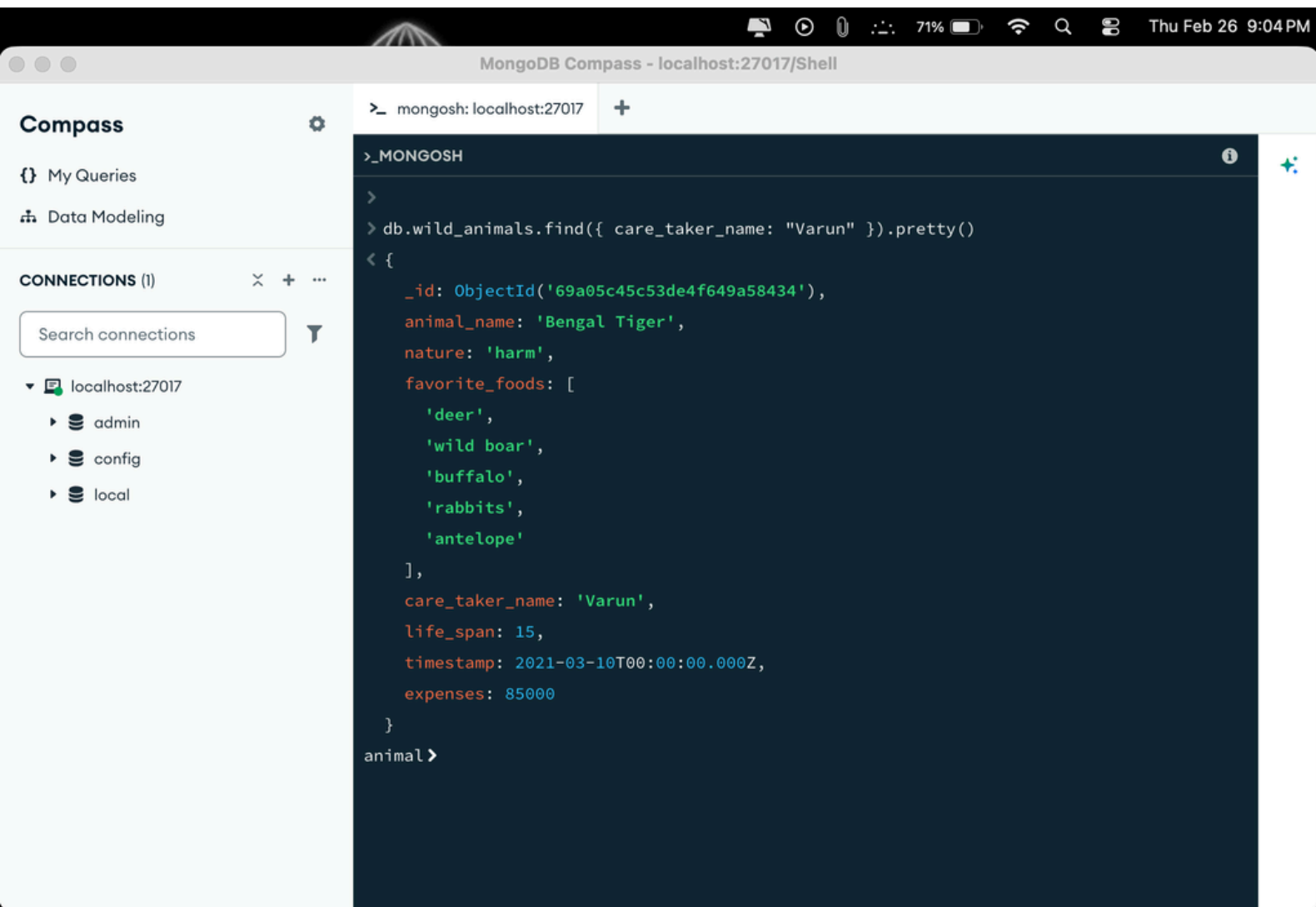
The screenshot shows the MongoDB Compass application interface. The top bar indicates the connection is to 'localhost:27017/Shell'. The left sidebar contains the 'Compass' logo, 'My Queries', 'Data Modeling', and 'CONNECTIONS (1)' section. Under 'CONNECTIONS (1)', there is a search bar and a list of connections: 'localhost:27017', 'admin', 'config', and 'local'. The main panel displays a MongoDB shell session. The prompt is '> mongosh: localhost:27017'. The command entered is 'db.domestic_animals.find({ life_span: 13 }).pretty()'. The output is a JSON document representing a dog named Bruno, with a life span of 13 years, favorite foods of meat, bones, dog biscuits, milk, and rice, and expenses of 15000.

```
> mongosh: localhost:27017 +

>_MONGOSH

> db.domestic_animals.find({ life_span: 13 }).pretty()
< {
  _id: ObjectId('69a05cf1c53de4f649a58439'),
  animal_name: 'Dog',
  gender: 'male',
  favorite_foods: [
    'meat',
    'bones',
    'dog biscuits',
    'milk',
    'rice'
  ],
  animal_petname: 'Bruno',
  life_span: 13,
  timestamp: 2021-04-12T00:00:00.000Z,
  expenses: 15000
}
animal>
```

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



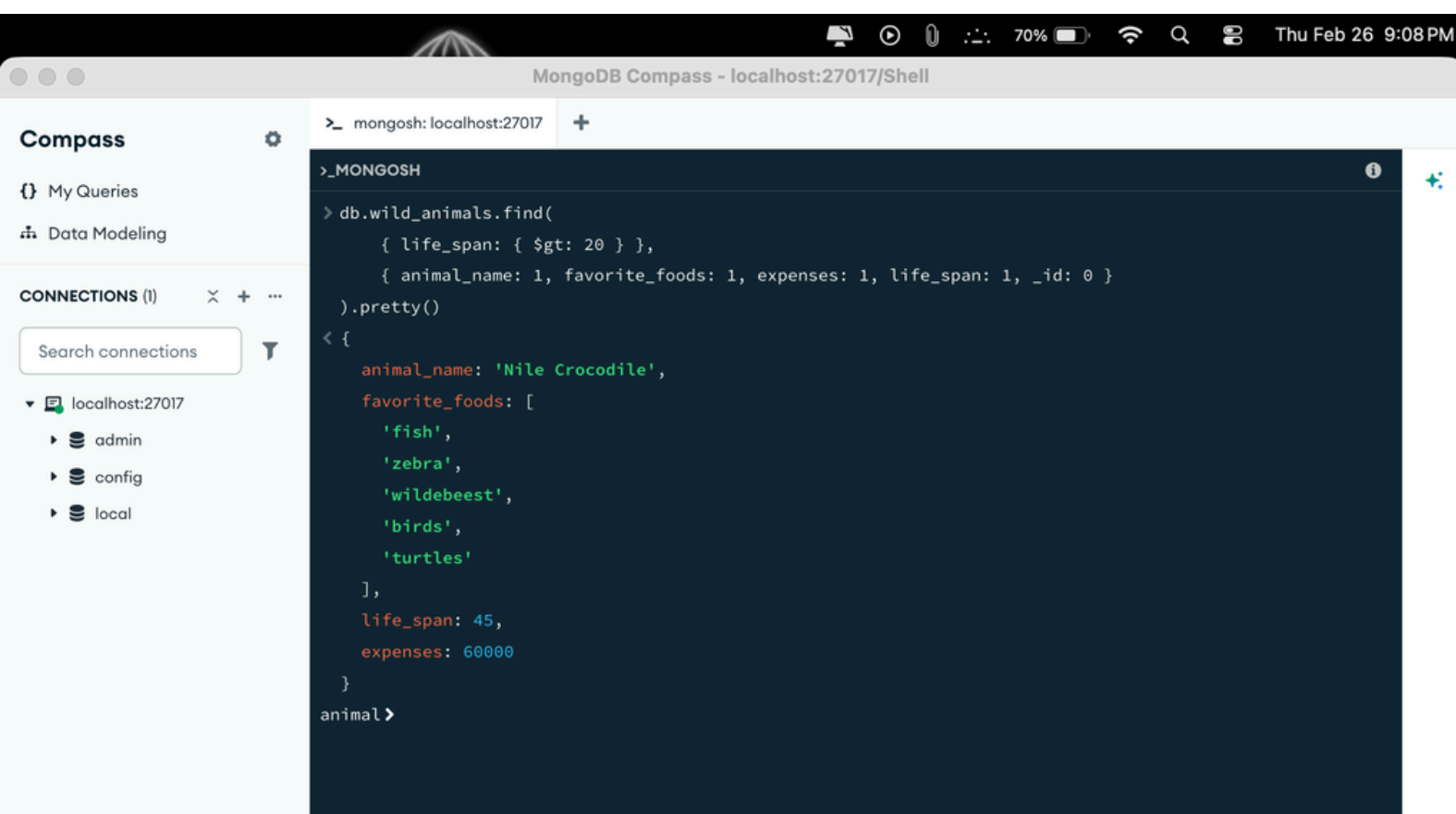
The screenshot shows the MongoDB Compass application interface. On the left, the 'Connections' panel shows a connection to 'localhost:27017' with databases 'admin', 'config', and 'local'. The main panel displays a MongoDB query in the 'Shell' tab:

```
> mongosh: localhost:27017 +
>_MONGOSH
>
> db.wild_animals.find({ care_taker_name: "Varun" }).pretty()
< {
  _id: ObjectId('69a05c45c53de4f649a58434'),
  animal_name: 'Bengal Tiger',
  nature: 'harm',
  favorite_foods: [
    'deer',
    'wild boar',
    'buffalo',
    'rabbits',
    'antelope'
  ],
  care_taker_name: 'Varun',
  life_span: 15,
  timestamp: 2021-03-10T00:00:00.000Z,
  expenses: 85000
}
```

The result shows a document for a Bengal Tiger, cared for by Varun, with a life span of 15 years, a timestamp of 2021-03-10T00:00:00.000Z, and expenses of 85000. The favorite foods listed are deer, wild boar, buffalo, rabbits, and antelope.

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

→ From Wild Animals :



The screenshot shows the MongoDB Compass application interface. The top bar indicates the connection is 'localhost:27017/Shell'. The left sidebar contains 'My Queries', 'Data Modeling', and 'CONNECTIONS (1)' with a search bar and a list of connections: 'localhost:27017', 'admin', 'config', and 'local'. The main panel shows a MongoDB query in the 'MONGOSH' shell:

```
> db.wild_animals.find(
  { life_span: { $gt: 20 } },
  { animal_name: 1, favorite_foods: 1, expenses: 1, life_span: 1, _id: 0 }
).pretty()
< {
  animal_name: 'Nile Crocodile',
  favorite_foods: [
    'fish',
    'zebra',
    'wildebeest',
    'birds',
    'turtles'
  ],
  life_span: 45,
  expenses: 60000
}
```

The result shows a single document for a Nile Crocodile with a life span of 45 years, favorite foods of fish, zebra, wildebeest, birds, and turtles, and expenses of 60000.

→ From Domestic Animals:

MongoDB Compass - localhost:27017/Shell

Compass

My Queries

Data Modeling

CONNECTIONS (1)

Search connections

localhost:27017

- admin
- config
- local

```
> mongosh: localhost:27017 +

>_MONGOSH

> db.domestic_animals.find(
  { life_span: { $gt: 15 } },
  { animal_name: 1, favorite_foods: 1, expenses: 1, life_span: 1, _id: 0 }
).pretty()
< {
  animal_name: 'Cow',
  favorite_foods: [
    'grass',
    'hay',
    'wheat',
    'corn',
    'vegetables'
  ],
  life_span: 20,
  expenses: 25000
}
{
  animal_name: 'Parrot',
  favorite_foods: [
    'seeds',
    'fruits',
    'nuts',
    'vegetables',
    'berries'
  ],
  life_span: 25,
  expenses: 10000
}
>
```