MongoDB Coding Challenge

Understanding Relationship

Inventory Management System

Suppliers:

```
[
  { "_id": 1, "name": "ABC Traders", "contact": "abc@abc.com" },
  { "_id": 2, "name": "XYZ Distributors", "contact": "xyz@xyz.com" },
  { "_id": 3, "name": "QRS Wholesale", "contact": "qrs@qrs.com" }
]
```

Products:

```
[
  { "_id": 101, "name": "Laptop", "price": 50000, "stock": 20, "supplier_id": 1 },
  { "_id": 102, "name": "Mouse", "price": 500, "stock": 100, "supplier_id": 1 },
  { "_id": 103, "name": "Keyboard", "price": 1200, "stock": 50, "supplier_id": 2 }
]
```

Orders:

```
[
  {
    "_id": 1,
    "customer_name": "Alice",
    "items": [
      { "product_id": 101, "quantity": 1 },
      { "product_id": 102, "quantity": 2 }
    ],
    "order_date": "2025-07-25"
  },
  {
    "_id": 2,
    "customer_name": "Bob",
    "items": [
      { "product_id": 103, "quantity": 1 }
    ],
    "order_date": "2025-07-26"
  },
  {
    "_id": 3,
    "customer_name": "Charlie",
    "items": [
      { "product_id": 102, "quantity": 1 }
    ],
    "order_date": "2025-07-27"
  }
]
```

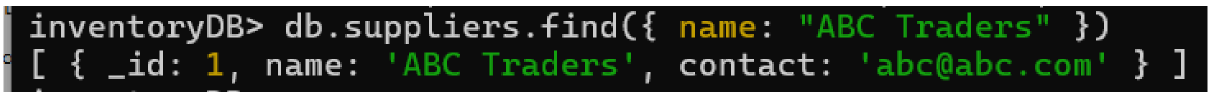
Questions:

1. Find all products supplied by "ABC Traders"

Relationship: One-to-Many (suppliers → products)

Step 1: Find supplier ID

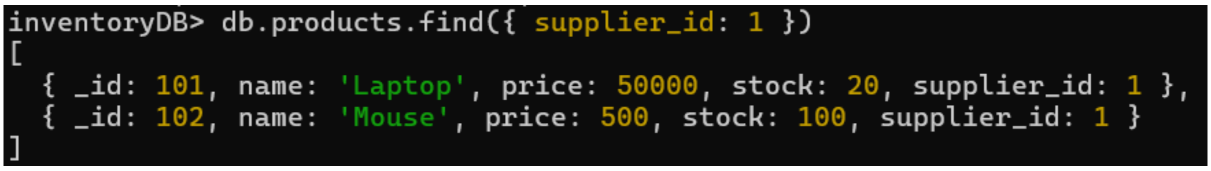
db.suppliers.find({ name: "ABC Traders" })



```
inventoryDB> db.suppliers.find({ name: "ABC Traders" })
[ { _id: 1, name: 'ABC Traders', contact: 'abc@abc.com' } ]
```

Step 2: Find products with that supplier_id

db.products.find({ supplier_id: 1 })

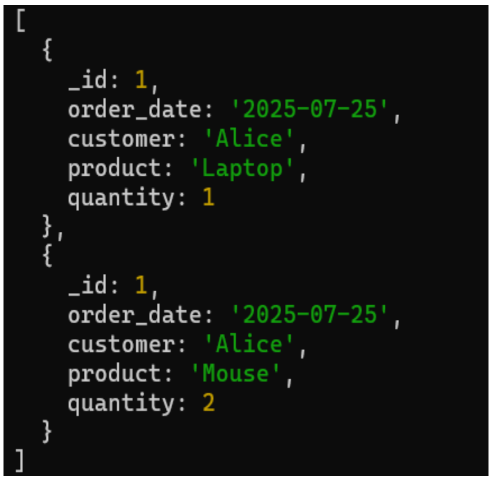


```
inventoryDB> db.products.find({ supplier_id: 1 })
[
  { _id: 101, name: 'Laptop', price: 50000, stock: 20, supplier_id: 1 },
  { _id: 102, name: 'Mouse', price: 500, stock: 100, supplier_id: 1 }
]
```

2. Get all orders placed by "Alice" with product names

Relationship: Many-to-Many (orders ↔ products via items array)

```
db.orders.aggregate([
 { $match: { customer_name: "Alice" } },
 { $unwind: "$items" },
 {
  $lookup: {
   from: "products",
   localField: "items.product_id",
   foreignField: "_id",
   as: "product_info"
  }
 },
 { $unwind: "$product_info" },
 {
  $project: {
   customer: "$customer_name",
   product: "$product_info.name",
   quantity: "$items.quantity",
   order_date: 1
  }
 }
])
```

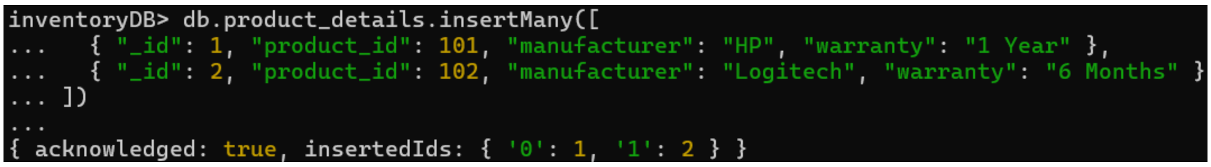

```
[
  {
    _id: 1,
    order_date: '2025-07-25',
    customer: 'Alice',
    product: 'Laptop',
    quantity: 1
  },
  {
    _id: 1,
    order_date: '2025-07-25',
    customer: 'Alice',
    product: 'Mouse',
    quantity: 2
  }
]
```

3. Show product details with embedded manufacturer info

Relationship: One-to-One (products ↔ product_details)

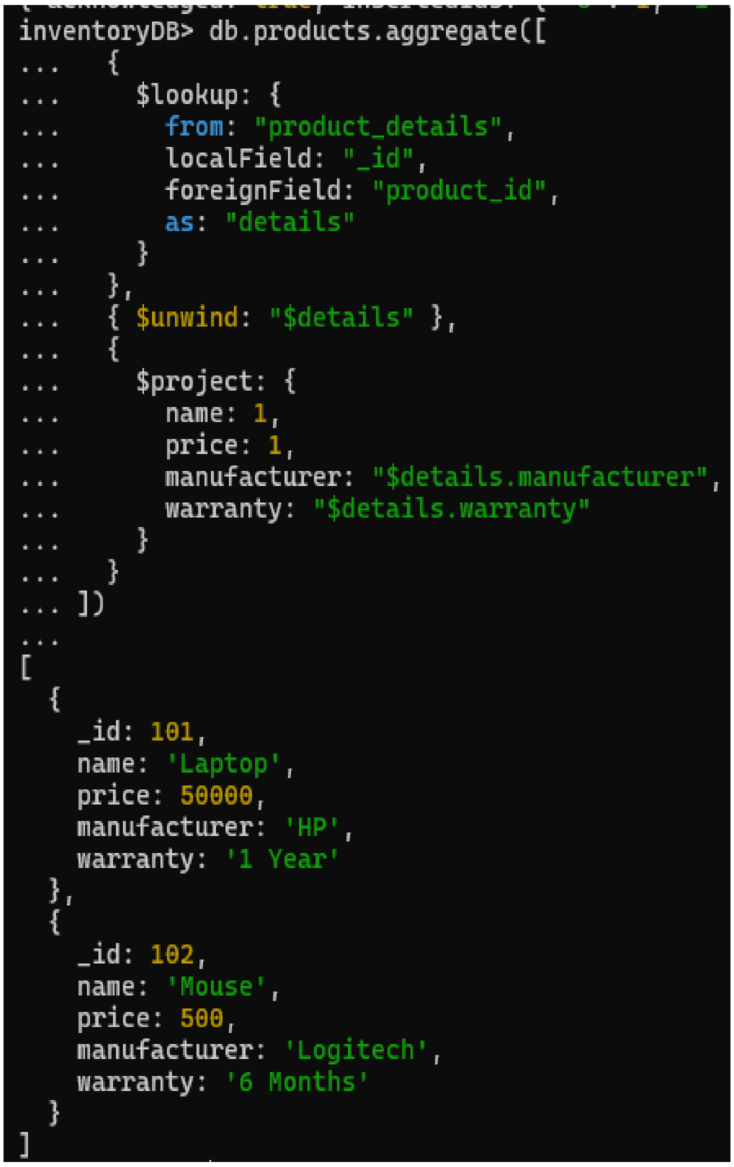
Step 1: Insert product details

```
db.product_details.insertMany([

 { "_id": 1, "product_id": 101, "manufacturer": "HP", "warranty": "1 Year" },

 { "_id": 2, "product_id": 102, "manufacturer": "Logitech", "warranty": "6 Months" }

])
```


```
inventoryDB> db.product_details.insertMany([
...   { "_id": 1, "product_id": 101, "manufacturer": "HP", "warranty": "1 Year" },
...   { "_id": 2, "product_id": 102, "manufacturer": "Logitech", "warranty": "6 Months" }
... ])
...
{ acknowledged: true, insertedIds: { '0': 1, '1': 2 } }
```

Step 2: Aggregate with lookup

```
db.products.aggregate([

 {

  $lookup: {

   from: "product_details",

   localField: "_id",

   foreignField: "product_id",

   as: "details"

  }

 },
 { $unwind: "$details" },
 {
   $project: {
     name: 1,
     price: 1,
     manufacturer: "$details.manufacturer",
     warranty: "$details.warranty"
   }
 }
])
```



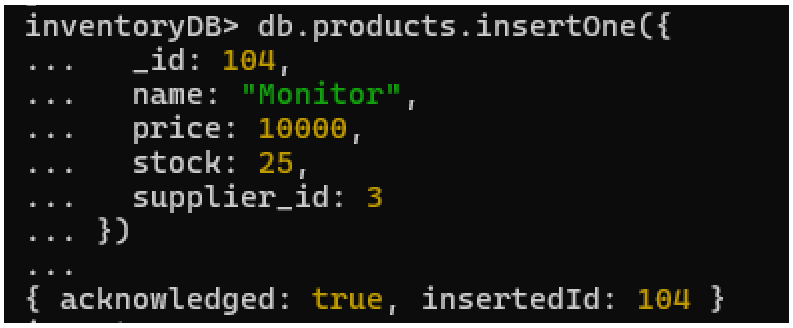
```
inventoryDB> db.products.aggregate([
...   {
...     $lookup: {
...       from: "product_details",
...       localField: "_id",
...       foreignField: "product_id",
...       as: "details"
...     }
...   },
...   { $unwind: "$details" },
...   {
...     $project: {
...       name: 1,
...       price: 1,
...       manufacturer: "$details.manufacturer",
...       warranty: "$details.warranty"
...     }
...   }
... ])
...
[
  {
    _id: 101,
    name: 'Laptop',
    price: 50000,
    manufacturer: 'HP',
    warranty: '1 Year'
  },
  {
    _id: 102,
    name: 'Mouse',
    price: 500,
    manufacturer: 'Logitech',
    warranty: '6 Months'
  }
]
```

4. Add a new product and a customer who ordered it

Relationship: Many-to-Many (products ↔ orders)

Insert a new product:

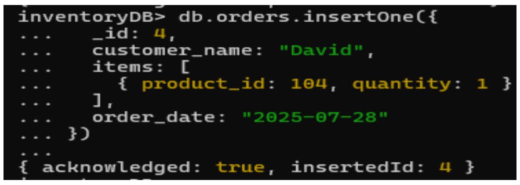
```
db.products.insertOne({
  _id: 104,
  name: "Monitor",
  price: 10000,
  stock: 25,
  supplier_id: 3
})
```



```
inventoryDB> db.products.insertOne({
...    _id: 104,
...    name: "Monitor",
...    price: 10000,
...    stock: 25,
...    supplier_id: 3
... })
...
{ acknowledged: true, insertedId: 104 }
```

Insert a new order:

```
db.orders.insertOne({
  _id: 4,
  customer_name: "David",
  items: [
    { product_id: 104, quantity: 1 }
  ],
  order_date: "2025-07-28"
})
```

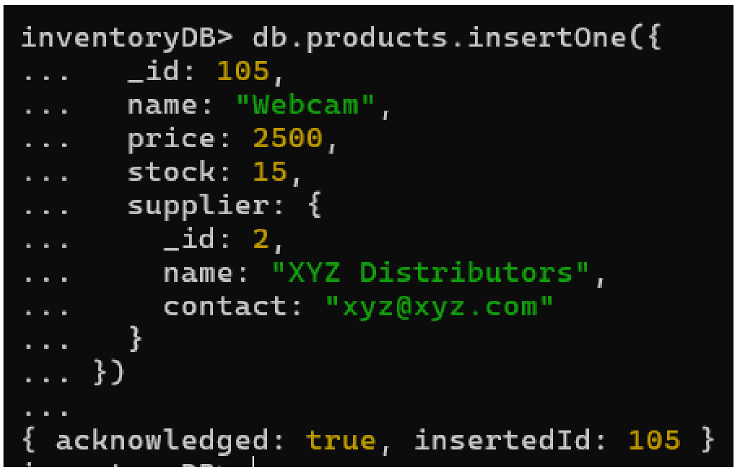


```
inventoryDB> db.orders.insertOne({
...    _id: 4,
...    customer_name: "David",
...    items: [
...      { product_id: 104, quantity: 1 }
...    ],
...    order_date: "2025-07-28"
... })
...
{ acknowledged: true, insertedId: 4 }
```

5. Embed supplier info inside a product document (alternate model)

Relationship: One-to-One (embedded)

```
db.products.insertOne({
  _id: 105,
  name: "Webcam",
  price: 2500,
  stock: 15,
  supplier: {
    _id: 2,
    name: "XYZ Distributors",
    contact: "xyz@xyz.com"
  }
})
```



```
inventoryDB> db.products.insertOne({
...    _id: 105,
...    name: "Webcam",
...    price: 2500,
...    stock: 15,
...    supplier: {
...      _id: 2,
...      name: "XYZ Distributors",
...      contact: "xyz@xyz.com"
...    }
... })
...
{ acknowledged: true, insertedId: 105 }
```