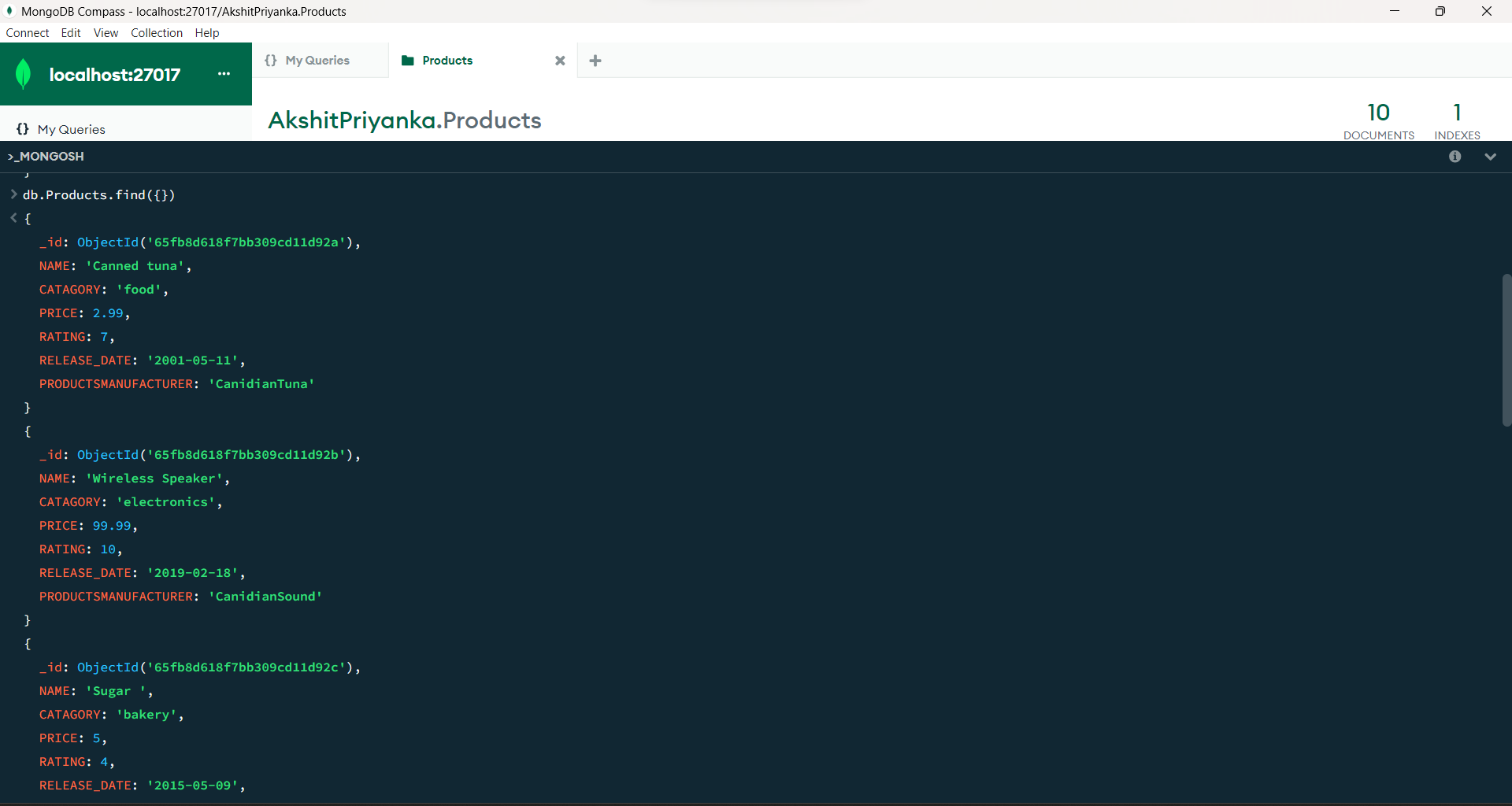
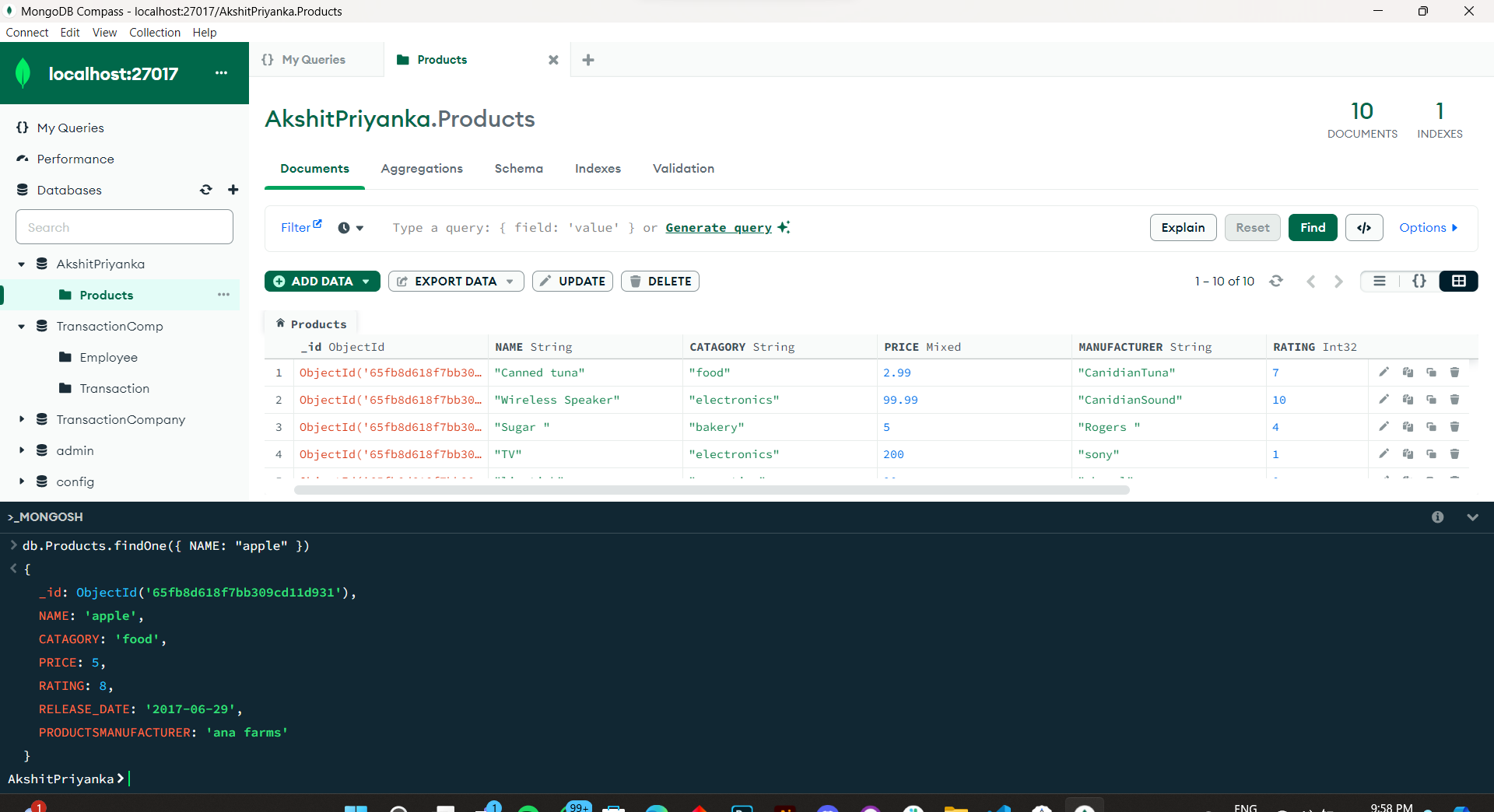
1)Retrieve details of all products in the collection.

db.Products({})



2) Find a product by name without using limit().

db.Products.findOne({ NAME: "apple" })



3) .Retrieve the 3rd most expensive products.

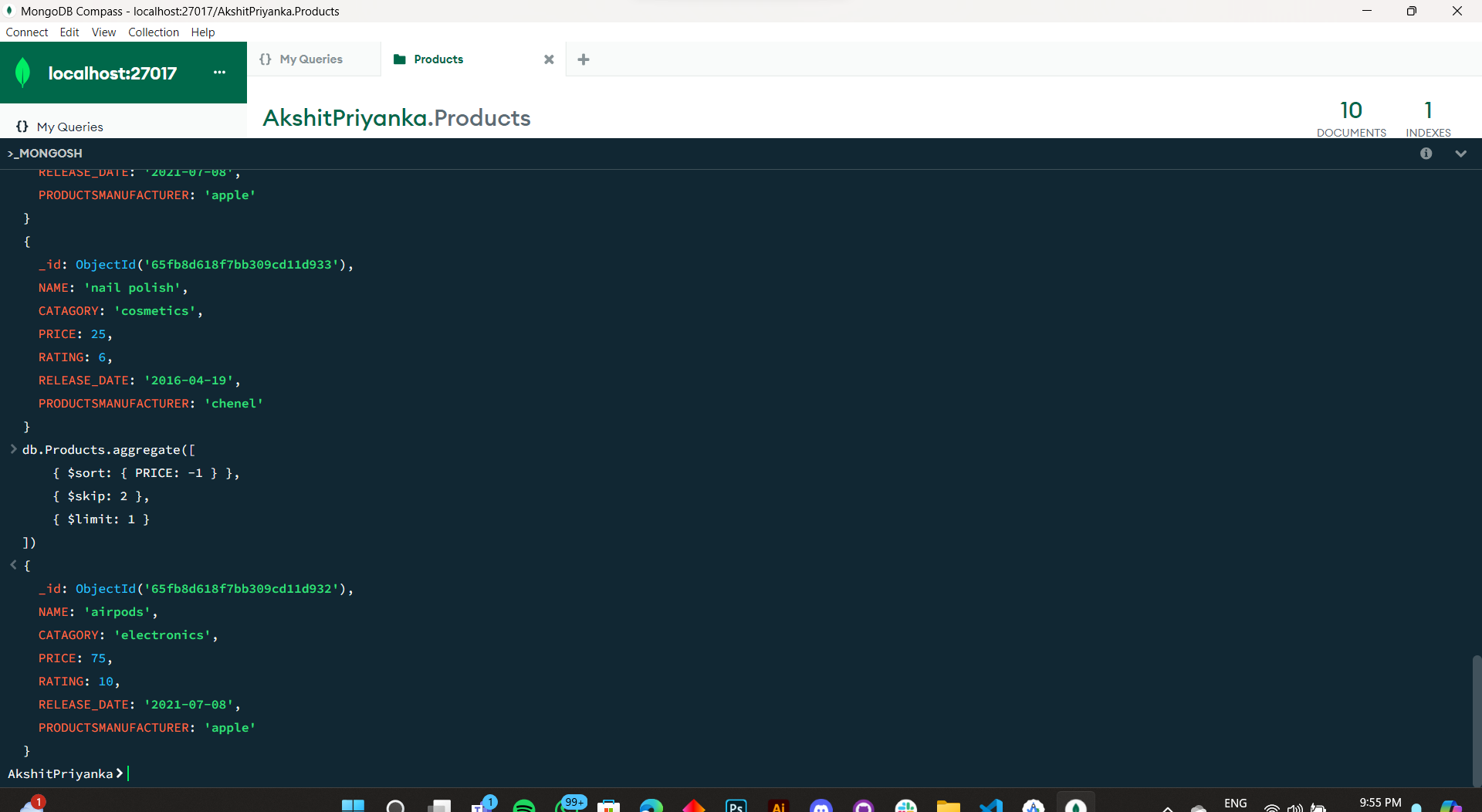
db.Products.aggregate([

{ $sort: { price: -1 } },

{ $skip: 2 },

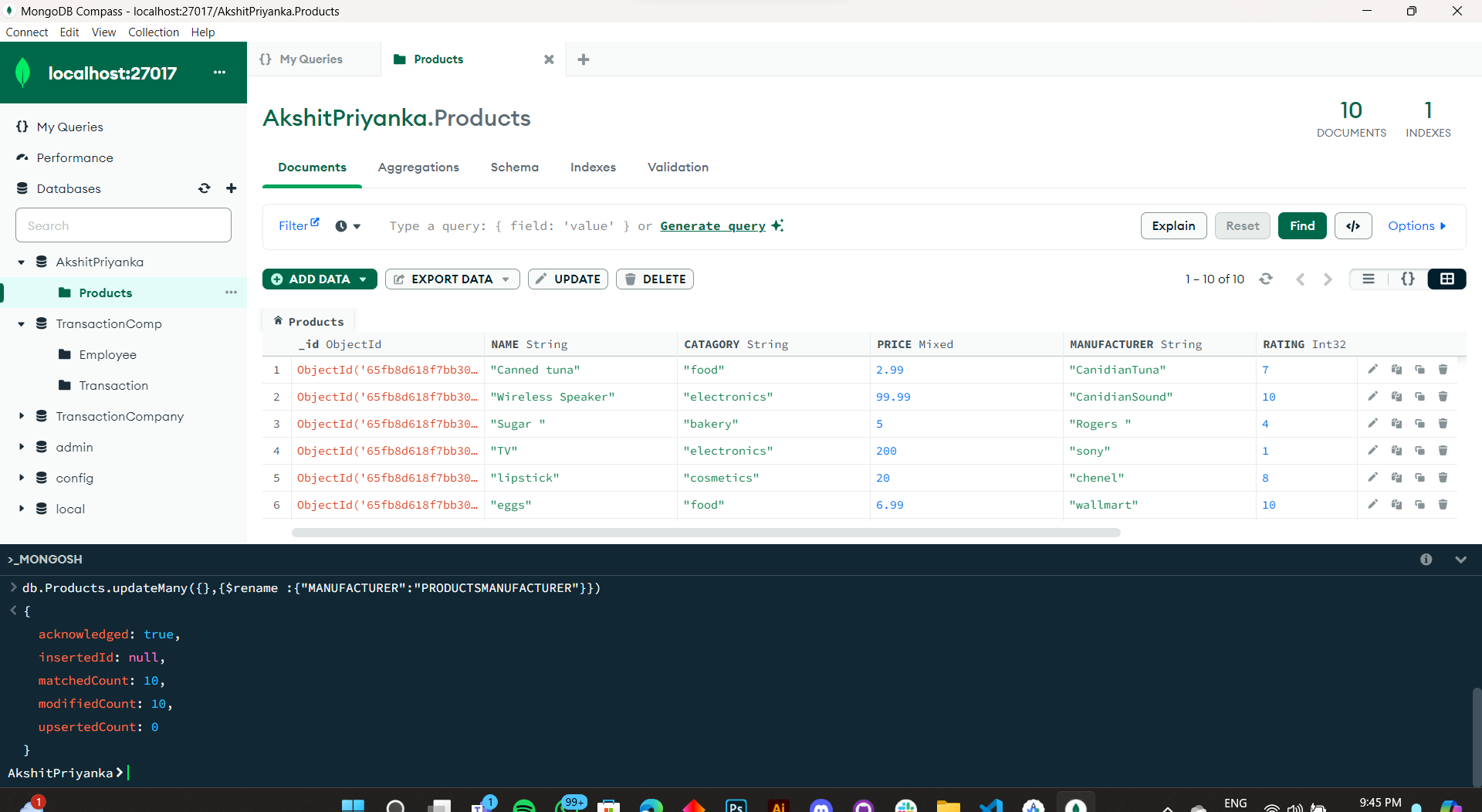
{ $limit: 1 }

])



4)Rename the field 'manufacturer' to 'Product Manufacturer.

db.Products.updateMany({},{$rename :{"MANUFACTURER":"PRODUCTSMANUFACTURER"}})



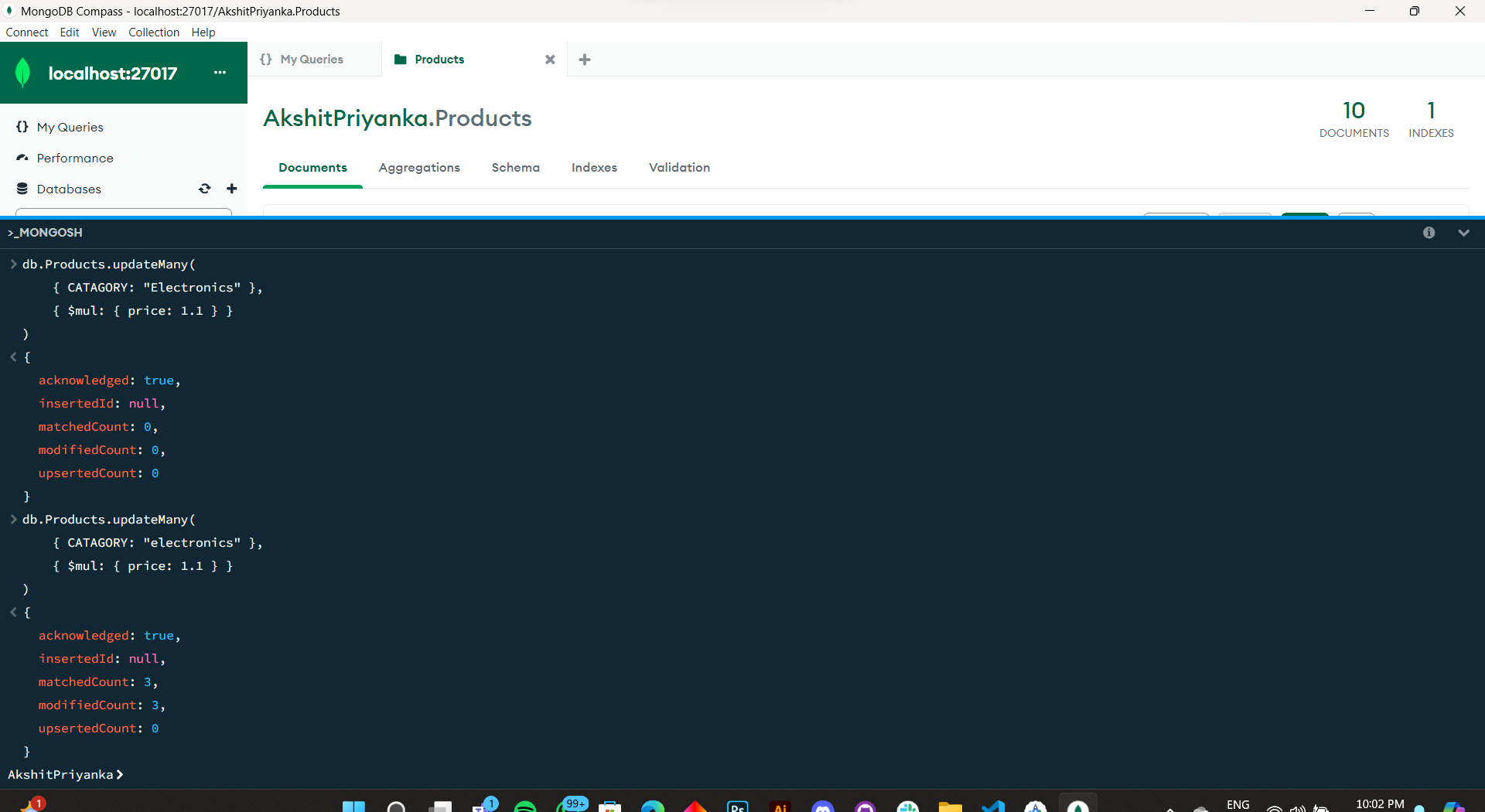
5). Increase the price of electronics category products by 10%

db.Products.updateMany(

{ CATAGORY: "electronics" },

{ $mul: { price: 1.1 } }

)



6) Find products with a price greater than the average price.

db.Products.aggregate([

{

$group: {

\_id: null,

averagePrice: { $avg: "$PRICE" } // Calculate the average price

}

},

{

$lookup: {

from: "Products",

let: { avgPrice: "$averagePrice" },

pipeline: [

{

$match: { $expr: { $gt: ["$PRICE", "$$avgPrice"] } } // Filter products with price greater than the average price

}

],

as: "aboveAverageProducts"

}

},

{

$unwind: "$aboveAverageProducts" // Flatten the array of products

},

{

$replaceRoot: { newRoot: "$aboveAverageProducts" } // Replace the root with the matching products

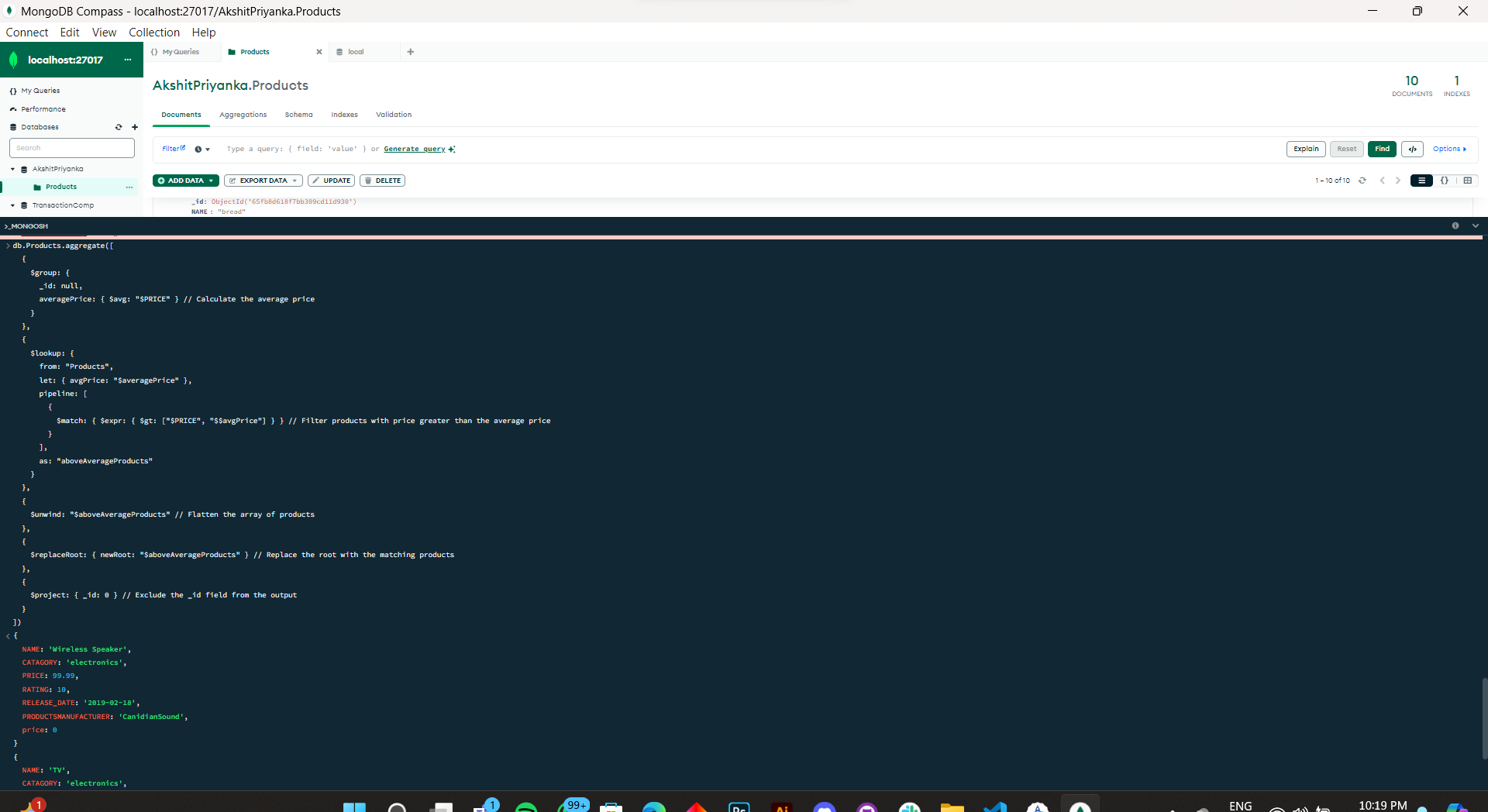
},

{

$project: { \_id: 0 } // Exclude the \_id field from the output

}

])



7) Update the 'release date' of the oldest product to the current date.

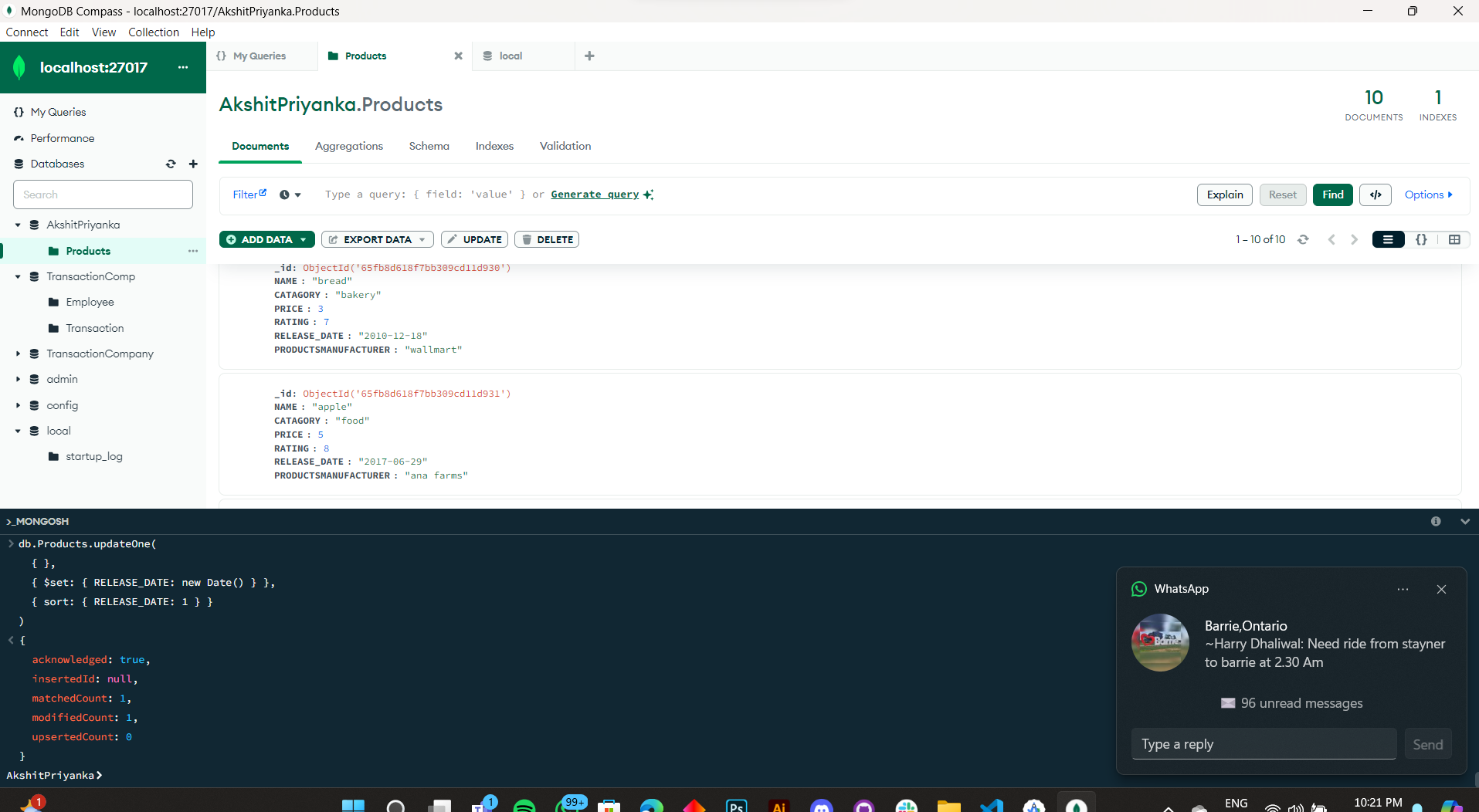
db.Products.updateOne(

{ },

{ $set: { RELEASE\_DATE: new Date() } },

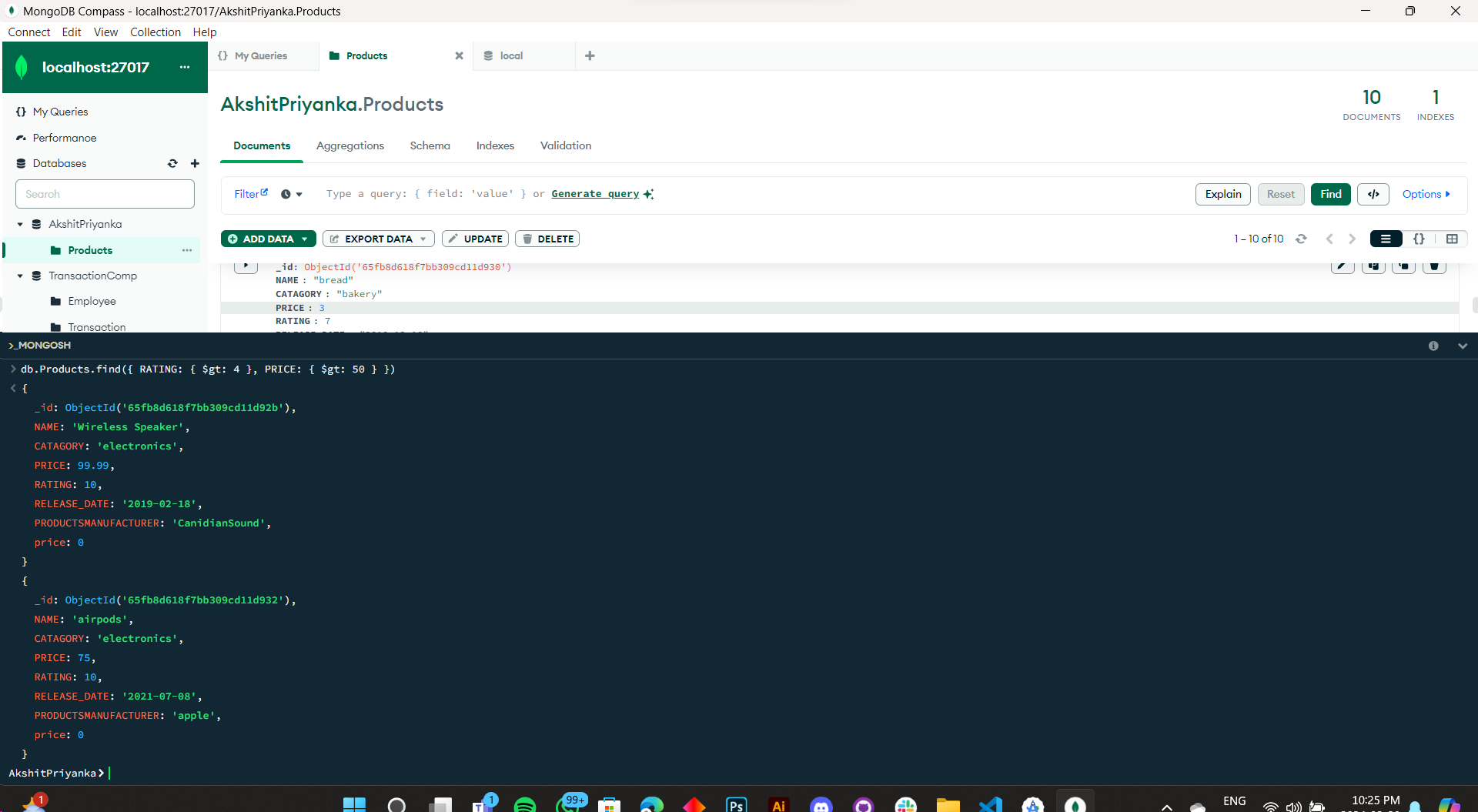
{ sort: { RELEASE\_DATE: 1 } }

)



8) . Identify and list premium products with a rating greater than 4

db.Products.find({ RATING: { $gt: 4 }, PRICE: { $gt: 50 } })



\*\* DIDN'T HAD A PREMIUM CATEGORY

9) Add a new product to the 'products' collection with relevant details.

db.Products.insertOne({

NAME: "unaliving super cool laser weapon",

CATEGORY: "weapon",

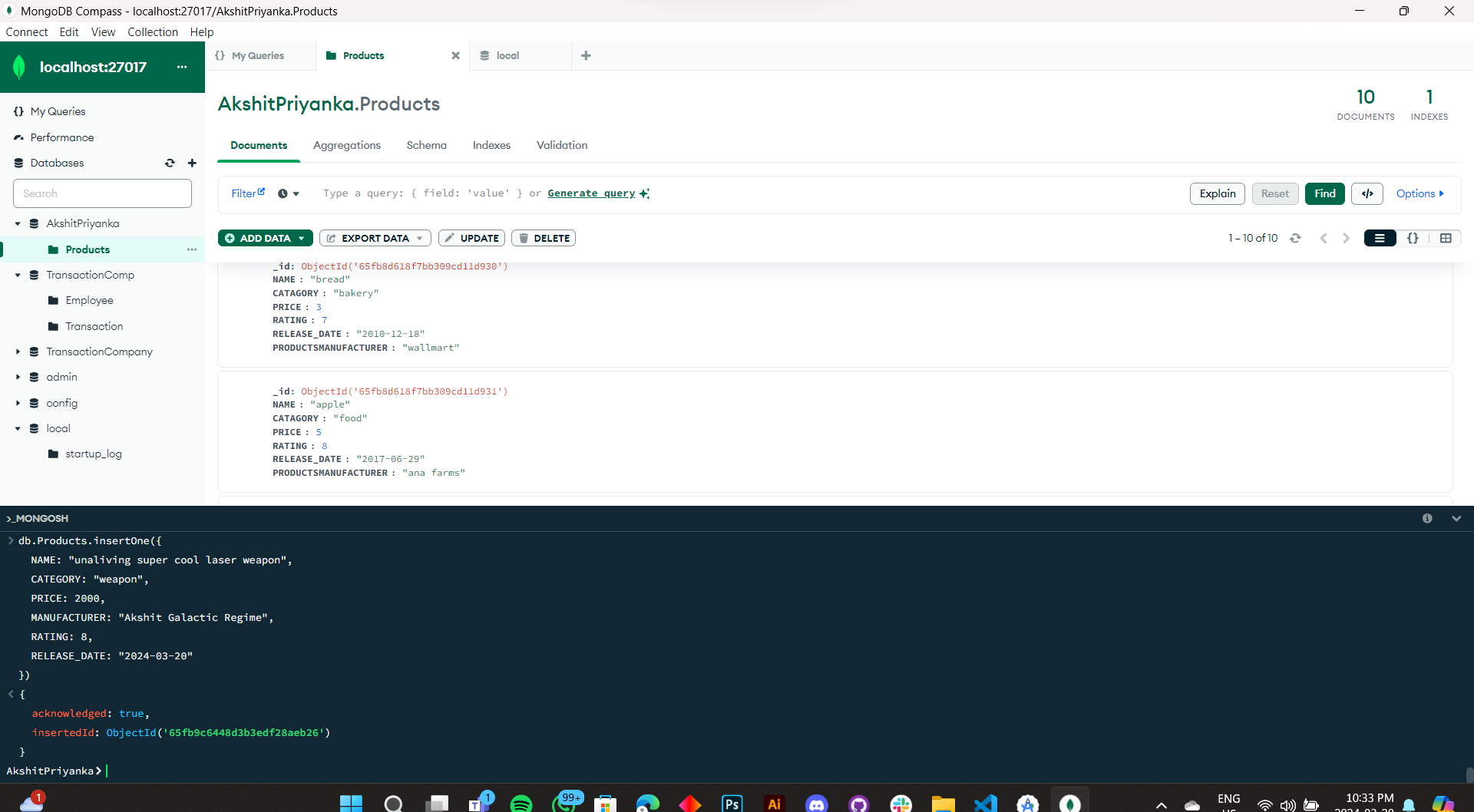
PRICE: 2000,

MANUFACTURER: "Akshit Galactic Regime",

RATING: 8,

RELEASE\_DATE: "2024-03-20"

})



10) Remove all products with a price less than $20.

db.Products.deleteMany({ PRICE: { $lt: 20 } })

