Theoretical Questions

```
# 1. What are the key differences between SQL and NoSQL databases?

# SQL databases use structured tables, while NoSQL databases use flexible document or key-value stores.
# SQL follows a fixed schema; NoSQL allows dynamic schemas.
# SQL ensures ACID properties; NoSQL prioritizes scalability.



# 2. What makes MongoDB a good choice for modern applications?

# Flexible schema for handling unstructured data.
# High scalability using horizontal scaling.
# Faster performance due to indexing and in-memory processing.



# 3. Explain the concept of collections in MongoDB.

# A collection is a group of related documents, similar to a table in SQL but schema-less.


# 4. How does MongoDB ensure high availability using replication?

# Uses replica sets with one primary node and multiple secondary nodes for failover support.




# 5. What are the main benefits of MongoDB Atlas?

# Fully managed cloud database.
# Automated backups, security, and scaling.






# 6. What is the role of indexes in MongoDB, and how do they improve performance?

# Indexes speed up query execution by reducing the need for full collection scans.


# 7. Describe the stages of the MongoDB aggregation pipeline.

# Stages include $match, $group, $sort, $project, $limit, and $lookup for data transformation.


# 8. What is sharding in MongoDB? How does it differ from replication?

# Sharding distributes data across multiple servers for horizontal scaling.
# Replication duplicates data for redundancy.



# 9. What is PyMongo, and why is it used?

# PyMongo is a Python driver for interacting with MongoDB.


# 10. What are the ACID properties in the context of MongoDB transactions?

# Atomicity, Consistency, Isolation, Durability ensure reliable transactions.


# 11. What is the purpose of MongoDB's explain() function?

# Analyzes query execution to optimize performance.


# 12. How does MongoDB handle schema validation?

# Uses JSON schema validation for enforcing rules on documents.
```

```
# 13. What is the difference between a primary and a secondary node in a replica set?

# Primary handles write operations, while secondaries replicate data.

# 14. What security mechanisms does MongoDB provide for data protection?

# Authentication, authorization, encryption, and network security.

# 15. Explain the concept of embedded documents and when they should be used.

# Documents within documents for efficient data retrieval. Used for related data.

# 16. What is the purpose of MongoDB's $lookup stage in aggregation?

# Joins documents from different collections.

# 17. What are some common use cases for MongoDB?

# Real-time analytics, content management, IoT, and mobile apps.

# 18. What are the advantages of using MongoDB for horizontal scaling?

# Handles large volumes of data efficiently.

# 19. How do MongoDB transactions differ from SQL transactions?

# MongoDB supports multi-document transactions but is not as strict as SQL.

# 20. What are the main differences between capped collections and regular collections?

# Capped collections have a fixed size and automatically remove old documents.

# 21. What is the purpose of the $match stage in MongoDB's aggregation pipeline?

# Filters documents before further processing.

# 22. How can you secure access to a MongoDB database?

# Enable authentication, use firewalls, and encrypt data.

# 23. What is MongoDB's WiredTiger storage engine, and why is it important?

# Provides better concurrency and compression for performance improvement.
```

1. Write a Python script to load the Superstore dataset from a CSV file into MongoDB.

```
!pip install pymongo dnspython
```

```
Collecting pymongo
  Downloading pymongo-4.11.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (22 kB)
Collecting dnspython
  Downloading dnspython-2.7.0-py3-none-any.whl.metadata (5.8 kB)
Downloading pymongo-4.11.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.4/1.4 MB 15.3 MB/s eta 0:00:00
Downloading dnspython-2.7.0-py3-none-any.whl (313 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 313.6/313.6 kB 20.4 MB/s eta 0:00:00
Installing collected packages: dnspython, pymongo
Successfully installed dnspython-2.7.0 pymongo-4.11.1
```

```
import pymongo
print("PyMongo installed successfully!")
```

```
PyMongo installed successfully!
```

```
from pymongo import MongoClient

MONGO_URI = "mongodb+srv://dinesh2018:Dinesh2025%40@cluster0.do346.mongodb.net/"

try:
    client = MongoClient(MONGO_URI)
    print("Connected to MongoDB Atlas successfully!")
except Exception as e:
    print("Connection failed:", e)
```

```
Connected to MongoDB Atlas successfully!
```

```
import pandas as pd
from pymongo import MongoClient

MONGO_URI = "mongodb+srv://dinesh2018:Dinesh2025%40@cluster0.do346.mongodb.net/"

client = MongoClient(MONGO_URI)
db = client["superstore"]
collection = db["orders"]

csv_path = "/content/superstore.csv"
df = pd.read_csv(csv_path, encoding="ISO-8859-1")

data = df.to_dict(orient="records")
collection.insert_many(data)

print("Data inserted successfully into MongoDB Atlas.")
```

```
Data inserted successfully into MongoDB Atlas.
```

2. Retrieve and print all documents from the Orders collection.

```
for doc in collection.find():
    print(doc)
```

```
Streaming output truncated to the last 5000 lines.
{'_id': ObjectId('67adee9ab251f4c1282d55f7'), 'Row ID': 4995, 'Order ID': 'CA-2015-153038', 'Order Date': '12/18/2015', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d55f8'), 'Row ID': 4996, 'Order ID': 'CA-2014-132227', 'Order Date': '11/4/2014', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d55f9'), 'Row ID': 4997, 'Order ID': 'CA-2017-155824', 'Order Date': '3/10/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d55fa'), 'Row ID': 4998, 'Order ID': 'CA-2017-155824', 'Order Date': '3/10/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d55fb'), 'Row ID': 4999, 'Order ID': 'CA-2016-129238', 'Order Date': '1/31/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d55fc'), 'Row ID': 5000, 'Order ID': 'CA-2016-129238', 'Order Date': '1/31/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d55fd'), 'Row ID': 5001, 'Order ID': 'CA-2017-159688', 'Order Date': '5/7/2017', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d55fe'), 'Row ID': 5002, 'Order ID': 'CA-2016-136126', 'Order Date': '5/24/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d55ff'), 'Row ID': 5003, 'Order ID': 'CA-2016-136126', 'Order Date': '5/24/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5600'), 'Row ID': 5004, 'Order ID': 'CA-2016-155033', 'Order Date': '10/7/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5601'), 'Row ID': 5005, 'Order ID': 'CA-2014-156006', 'Order Date': '4/30/2014', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5602'), 'Row ID': 5006, 'Order ID': 'CA-2015-158659', 'Order Date': '11/10/2015', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5603'), 'Row ID': 5007, 'Order ID': 'CA-2015-169796', 'Order Date': '11/9/2015', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5604'), 'Row ID': 5008, 'Order ID': 'CA-2015-169796', 'Order Date': '11/9/2015', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5605'), 'Row ID': 5009, 'Order ID': 'CA-2015-102876', 'Order Date': '9/7/2015', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d5606'), 'Row ID': 5010, 'Order ID': 'CA-2015-102876', 'Order Date': '9/7/2015', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d5607'), 'Row ID': 5011, 'Order ID': 'CA-2015-102876', 'Order Date': '9/7/2015', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d5608'), 'Row ID': 5012, 'Order ID': 'US-2017-139647', 'Order Date': '5/11/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5609'), 'Row ID': 5013, 'Order ID': 'US-2017-160465', 'Order Date': '7/21/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d560a'), 'Row ID': 5014, 'Order ID': 'US-2017-160465', 'Order Date': '7/21/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d560b'), 'Row ID': 5015, 'Order ID': 'US-2017-160465', 'Order Date': '7/21/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d560c'), 'Row ID': 5016, 'Order ID': 'CA-2014-153850', 'Order Date': '11/24/2014', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d560d'), 'Row ID': 5017, 'Order ID': 'CA-2014-153850', 'Order Date': '11/24/2014', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d560e'), 'Row ID': 5018, 'Order ID': 'CA-2014-127558', 'Order Date': '11/15/2014', 'Ship Dat
```

```
{'_id': ObjectId('67adee9ab251f4c1282d560f'), 'Row ID': 5019, 'Order ID': 'CA-2014-127558', 'Order Date': '11/15/2014', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5610'), 'Row ID': 5020, 'Order ID': 'CA-2014-127558', 'Order Date': '11/15/2014', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5611'), 'Row ID': 5021, 'Order ID': 'CA-2017-136511', 'Order Date': '3/23/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5612'), 'Row ID': 5022, 'Order ID': 'CA-2016-133795', 'Order Date': '12/18/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5613'), 'Row ID': 5023, 'Order ID': 'CA-2016-133795', 'Order Date': '12/18/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5614'), 'Row ID': 5024, 'Order ID': 'CA-2016-133795', 'Order Date': '12/18/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5615'), 'Row ID': 5025, 'Order ID': 'US-2017-130953', 'Order Date': '7/29/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5616'), 'Row ID': 5026, 'Order ID': 'US-2017-130953', 'Order Date': '7/29/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5617'), 'Row ID': 5027, 'Order ID': 'US-2017-130953', 'Order Date': '7/29/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5618'), 'Row ID': 5028, 'Order ID': 'US-2017-130953', 'Order Date': '7/29/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5619'), 'Row ID': 5029, 'Order ID': 'CA-2014-151792', 'Order Date': '9/2/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d561a'), 'Row ID': 5030, 'Order ID': 'CA-2017-139304', 'Order Date': '1/29/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d561b'), 'Row ID': 5031, 'Order ID': 'CA-2017-139304', 'Order Date': '1/29/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d561c'), 'Row ID': 5032, 'Order ID': 'CA-2017-139304', 'Order Date': '1/29/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d561d'), 'Row ID': 5033, 'Order ID': 'CA-2016-155166', 'Order Date': '12/26/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d561e'), 'Row ID': 5034, 'Order ID': 'CA-2016-155166', 'Order Date': '12/26/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d561f'), 'Row ID': 5035, 'Order ID': 'CA-2015-103954', 'Order Date': '8/9/2015', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d5620'), 'Row ID': 5036, 'Order ID': 'CA-2014-169803', 'Order Date': '4/6/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d5621'), 'Row ID': 5037, 'Order ID': 'CA-2014-169803', 'Order Date': '4/6/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d5622'), 'Row ID': 5038, 'Order ID': 'CA-2017-141719', 'Order Date': '11/17/2017', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5623'), 'Row ID': 5039, 'Order ID': 'CA-2015-136469', 'Order Date': '7/11/2015', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5624'), 'Row ID': 5040, 'Order ID': 'CA-2015-136469', 'Order Date': '7/11/2015', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5625'), 'Row ID': 5041, 'Order ID': 'CA-2015-136469', 'Order Date': '7/11/2015', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d5626'), 'Row ID': 5042, 'Order ID': 'CA-2016-158694', 'Order Date': '11/10/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5627'), 'Row ID': 5043, 'Order ID': 'CA-2016-158694', 'Order Date': '11/10/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5628'), 'Row ID': 5044, 'Order ID': 'CA-2016-158694', 'Order Date': '11/10/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d5629'), 'Row ID': 5045, 'Order ID': 'CA-2016-158694', 'Order Date': '11/10/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d562a'), 'Row ID': 5046, 'Order ID': 'CA-2016-158694', 'Order Date': '11/10/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d562b'), 'Row ID': 5047, 'Order ID': 'CA-2016-158694', 'Order Date': '11/10/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d562c'), 'Row ID': 5048, 'Order ID': 'CA-2016-158694', 'Order Date': '11/10/2016', 'Ship Dat
{'_id': ObjectId('67adee9ab251f4c1282d562d'), 'Row ID': 5049, 'Order ID': 'US-2014-104759', 'Order Date': '3/31/2014', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d562e'), 'Row ID': 5050, 'Order ID': 'US-2014-104759', 'Order Date': '3/31/2014', 'Ship Date
```

3. Count and display the total number of documents in the Orders collection.

```
print(collection.count_documents({}))
```

```
9994
```

4. Write a query to fetch all orders from the "West" region.

```
for doc in collection.find({"Region": "West"}):
    print(doc)
```

```
enex B1-RE Series Chair Mats for Low Pile Carpets', 'Sales': 91.96, 'Quantity': 2, 'Discount': 0.0, 'Profit': 15.6332}
One CHATAttach 160 -\xa0speaker phone', 'Sales': 1487.976, 'Quantity': 3, 'Discount': 0.2, 'Profit': 185.997}
e': 'Compact Automatic Electric Letter Opener', 'Sales': 238.62, 'Quantity': 2, 'Discount': 0.0, 'Profit': 4.7724}
ame': 'Hoover Commercial Soft Guard Upright Vacuum And Disposable Filtration Bags', 'Sales': 7.77, 'Quantity': 1, 'Discount': 0.0,
o Sheffield Collection Coffee Table, End Table, Center Table, Corner Table', 'Sales': 285.48, 'Quantity': 5, 'Discount': 0.2, 'Pro
': 'DXL Angle-View Binders with Locking Rings, Black', 'Sales': 19.168, 'Quantity': 4, 'Discount': 0.2, 'Profit': 6.4692}
ech G35 7.1-Channel Surround Sound Headset', 'Sales': 519.96, 'Quantity': 4, 'Discount': 0.0, 'Profit': 176.7864}
in GC17055 Auxiliary Audio Cable', 'Sales': 57.568, 'Quantity': 4, 'Discount': 0.2, 'Profit': 5.7568}
 Profile Extra Capacity Storage Tub', 'Sales': 83.7, 'Quantity': 5, 'Discount': 0.0, 'Profit': 3.348}
/S 25825 Wireless digital phone', 'Sales': 415.968, 'Quantity': 4, 'Discount': 0.2, 'Profit': 51.996}
arina Media Storage Towers in Natural & Black', 'Sales': 304.9, 'Quantity': 5, 'Discount': 0.0, 'Profit': 6.098}
 Contemporary Wood Frame with Silver Metal Mat, Desktop, 11 x 14 Size', 'Sales': 80.96, 'Quantity': 4, 'Discount': 0.0, 'Profit': 2
ellowes Recycled Storage Drawers', 'Sales': 777.21, 'Quantity': 7, 'Discount': 0.0, 'Profit': 54.4047}
ox 203', 'Sales': 32.4, 'Quantity': 5, 'Discount': 0.0, 'Profit': 15.552}
ir - Contemporary Swivel Chair with Padded Adjustable Arms and Flex Back', 'Sales': 225.568, 'Quantity': 2, 'Discount': 0.2, 'Prof
antus Panel Wall Certificate Holder - 8.5x11', 'Sales': 36.6, 'Quantity': 3, 'Discount': 0.0, 'Profit': 15.372}
```

5. Write a query to find orders where Sales is greater than 500.

```
for doc in collection.find({"Sales": {"$gt": 500}}):
    print(doc)
```


```
{'_id': ObjectId('67adee9ab251f4c1282d4276'), 'Row ID': 2, 'Order ID': 'CA-2016-152156', 'Order Date': '11/8/2016', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4278'), 'Row ID': 4, 'Order ID': 'US-2015-108966', 'Order Date': '10/11/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d427c'), 'Row ID': 8, 'Order ID': 'CA-2014-115812', 'Order Date': '6/9/2014', 'Ship Date': '
{'_id': ObjectId('67adee9ab251f4c1282d427f'), 'Row ID': 11, 'Order ID': 'CA-2014-115812', 'Order Date': '6/9/2014', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4280'), 'Row ID': 12, 'Order ID': 'CA-2014-115812', 'Order Date': '6/9/2014', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4285'), 'Row ID': 17, 'Order ID': 'CA-2014-105893', 'Order Date': '11/11/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d428d'), 'Row ID': 25, 'Order ID': 'CA-2015-106320', 'Order Date': '9/25/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4290'), 'Row ID': 28, 'Order ID': 'US-2015-150630', 'Order Date': '9/17/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4298'), 'Row ID': 36, 'Order ID': 'CA-2016-117590', 'Order Date': '12/8/2016', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d429b'), 'Row ID': 39, 'Order ID': 'CA-2015-117415', 'Order Date': '12/27/2015', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d42ab'), 'Row ID': 55, 'Order ID': 'CA-2016-105816', 'Order Date': '12/11/2016', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d42b8'), 'Row ID': 68, 'Order ID': 'CA-2014-106376', 'Order Date': '12/5/2014', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d42bd'), 'Row ID': 73, 'Order ID': 'US-2015-134026', 'Order Date': '4/26/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d42ea'), 'Row ID': 118, 'Order ID': 'CA-2015-110457', 'Order Date': '3/2/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d42f1'), 'Row ID': 125, 'Order ID': 'US-2014-152030', 'Order Date': '12/26/2014', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d42f2'), 'Row ID': 126, 'Order ID': 'US-2014-134614', 'Order Date': '9/20/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d4305'), 'Row ID': 145, 'Order ID': 'CA-2017-155376', 'Order Date': '12/22/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d4306'), 'Row ID': 146, 'Order ID': 'CA-2015-110744', 'Order Date': '9/7/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d430a'), 'Row ID': 150, 'Order ID': 'CA-2016-114489', 'Order Date': '12/5/2016', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d4314'), 'Row ID': 160, 'Order ID': 'CA-2016-114104', 'Order Date': '11/20/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d431a'), 'Row ID': 166, 'Order ID': 'CA-2014-139892', 'Order Date': '9/8/2014', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d431c'), 'Row ID': 168, 'Order ID': 'CA-2014-139892', 'Order Date': '9/8/2014', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d432b'), 'Row ID': 183, 'Order ID': 'CA-2014-158274', 'Order Date': '11/19/2014', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d4332'), 'Row ID': 190, 'Order ID': 'CA-2015-102281', 'Order Date': '10/12/2015', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d4335'), 'Row ID': 193, 'Order ID': 'CA-2015-102281', 'Order Date': '10/12/2015', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d434c'), 'Row ID': 216, 'Order ID': 'CA-2015-146262', 'Order Date': '1/2/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4357'), 'Row ID': 227, 'Order ID': 'CA-2015-163055', 'Order Date': '8/9/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d435d'), 'Row ID': 233, 'Order ID': 'US-2017-100930', 'Order Date': '4/7/2017', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4360'), 'Row ID': 236, 'Order ID': 'US-2017-100930', 'Order Date': '4/7/2017', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4369'), 'Row ID': 245, 'Order ID': 'CA-2014-131926', 'Order Date': '6/1/2014', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d436c'), 'Row ID': 248, 'Order ID': 'CA-2014-131926', 'Order Date': '6/1/2014', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d4370'), 'Row ID': 252, 'Order ID': 'CA-2016-145625', 'Order Date': '9/11/2016', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d4376'), 'Row ID': 258, 'Order ID': 'US-2015-159982', 'Order Date': '11/28/2015', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d437b'), 'Row ID': 263, 'Order ID': 'US-2014-106992', 'Order Date': '9/19/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d437c'), 'Row ID': 264, 'Order ID': 'US-2014-106992', 'Order Date': '9/19/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d4383'), 'Row ID': 271, 'Order ID': 'CA-2017-163979', 'Order Date': '12/28/2017', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d438e'), 'Row ID': 282, 'Order ID': 'US-2015-161991', 'Order Date': '9/26/2015', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d438f'), 'Row ID': 283, 'Order ID': 'CA-2015-130890', 'Order Date': '11/2/2015', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d4395'), 'Row ID': 289, 'Order ID': 'CA-2016-112697', 'Order Date': '12/18/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d43a0'), 'Row ID': 300, 'Order ID': 'CA-2016-142545', 'Order Date': '10/28/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d43b3'), 'Row ID': 319, 'Order ID': 'CA-2014-164973', 'Order Date': '11/4/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d43bd'), 'Row ID': 329, 'Order ID': 'US-2016-141544', 'Order Date': '8/30/2016', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d43c4'), 'Row ID': 336, 'Order ID': 'CA-2015-137946', 'Order Date': '9/1/2015', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d43c7'), 'Row ID': 339, 'Order ID': 'CA-2014-129924', 'Order Date': '7/12/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d43ca'), 'Row ID': 342, 'Order ID': 'CA-2014-122336', 'Order Date': '4/13/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d43d6'), 'Row ID': 354, 'Order ID': 'CA-2016-129714', 'Order Date': '9/1/2016', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d43dc'), 'Row ID': 360, 'Order ID': 'CA-2017-155698', 'Order Date': '3/8/2017', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d43e6'), 'Row ID': 370, 'Order ID': 'CA-2016-155516', 'Order Date': '10/21/2016', 'Ship Date
{'_id': ObjectId('67adee9ab251f4c1282d43ea'), 'Row ID': 374, 'Order ID': 'US-2014-119137', 'Order Date': '7/23/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d43ee'), 'Row ID': 378, 'Order ID': 'US-2017-134481', 'Order Date': '8/27/2017', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d43fd'), 'Row ID': 393, 'Order ID': 'US-2014-135972', 'Order Date': '9/21/2014', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d4404'), 'Row ID': 400, 'Order ID': 'CA-2016-108987', 'Order Date': '9/8/2016', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d440b'), 'Row ID': 407, 'Order ID': 'CA-2017-117457', 'Order Date': '12/8/2017', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d440d'), 'Row ID': 409, 'Order ID': 'CA-2017-117457', 'Order Date': '12/8/2017', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d4411'), 'Row ID': 413, 'Order ID': 'CA-2017-117457', 'Order Date': '12/8/2017', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d441d'), 'Row ID': 425, 'Order ID': 'CA-2017-155705', 'Order Date': '8/21/2017', 'Ship Date'
{'_id': ObjectId('67adee9ab251f4c1282d4426'), 'Row ID': 434, 'Order ID': 'CA-2016-127369', 'Order Date': '6/6/2016', 'Ship Date':
```


6. Fetch the top 3 orders with the highest Profit.

```
for doc in collection.find().sort("Profit", -1).limit(3):
    print(doc)
```

```
{'_id': ObjectId('67adee9ab251f4c1282d5d1f'), 'Row ID': 6827, 'Order ID': 'CA-2016-118689', 'Order Date': '10/2/2016', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d624e'), 'Row ID': 8154, 'Order ID': 'CA-2017-140151', 'Order Date': '3/23/2017', 'Ship Date':
{'_id': ObjectId('67adee9ab251f4c1282d52d3'), 'Row ID': 4191, 'Order ID': 'CA-2017-166709', 'Order Date': '11/17/2017', 'Ship Date'
```

7. Update all orders with Ship Mode as "First Class" to "Premium Class."

```
collection.update_many({"Ship Mode": "First Class"}, {"$set": {"Ship Mode": "Premium Class"}})
```

```
UpdateResult({'n': 1538, 'electionId': ObjectId('7fffffff0000000000000003'), 'opTime': {'ts': Timestamp(1739452284, 1562), 't': 3},
'nModified': 1538, 'ok': 1.0, '$clusterTime': {'clusterTime': Timestamp(1739452284, 1562), 'signature': {'hash':
b'\x1c\xe8\x16a\xb5\x01\x9e \x86\xe1@\xe7\x04p.Ww\xfa\xfc\x98', 'keyId': 7470572334051491846}}, 'operationTime':
Timestamp(1739452284, 1562), 'updatedExisting': True}, acknowledged=True)
```

8. Delete all orders where Sales is less than 50.

```
collection.delete_many({"Sales": {"$lt": 50}})
```

```
DeleteResult({'n': 4849, 'electionId': ObjectId('7fffffff0000000000000003'), 'opTime': {'ts': Timestamp(1739452297, 528), 't': 3},
'ok': 1.0, '$clusterTime': {'clusterTime': Timestamp(1739452297, 528), 'signature': {'hash':
b'\x00x\xb9\x84X\x01\x15g\x1d\x19o,\x80\xa4Y\x9d\xb0e\xe7\xc5', 'keyId': 7470572334051491846}}, 'operationTime':
Timestamp(1739452297, 528)}, acknowledged=True)
```

9. Use aggregation to group orders by Region and calculate total sales per region.

```
pipeline = [
    {"$group": {"_id": "$Region", "Total Sales": {"$sum": "$Sales"}}}
]
result = collection.aggregate(pipeline)
for doc in result:
    print(doc)
```

```
{'_id': 'West', 'Total Sales': 694686.6195}
{'_id': 'East', 'Total Sales': 651137.705}
{'_id': 'Central', 'Total Sales': 479611.8458}
{'_id': 'South', 'Total Sales': 376023.312}
```

10. Fetch all distinct values for Ship Mode from the collection.

```
print(collection.distinct("Ship Mode"))
```

```
['Premium Class', 'Same Day', 'Second Class', 'Standard Class']
```

11. Count the number of orders for each category.

```
pipeline = [
    {"$group": {"_id": "$Category", "Total Orders": {"$sum": 1}}}
]
result = collection.aggregate(pipeline)
for doc in result:
    print(doc)
```

```
{'_id': 'Technology', 'Total Orders': 1496}
{'_id': 'Office Supplies', 'Total Orders': 2076}
{'_id': 'Furniture', 'Total Orders': 1573}
```

Start coding or generate with AI.