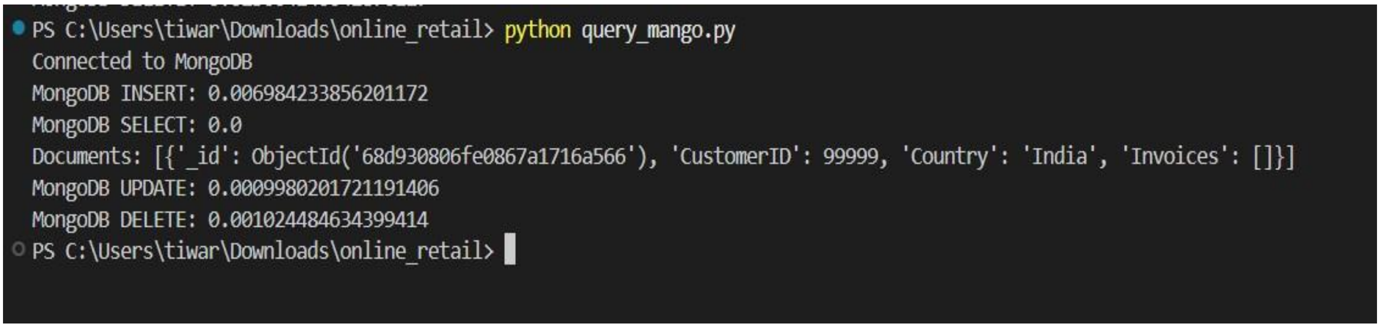
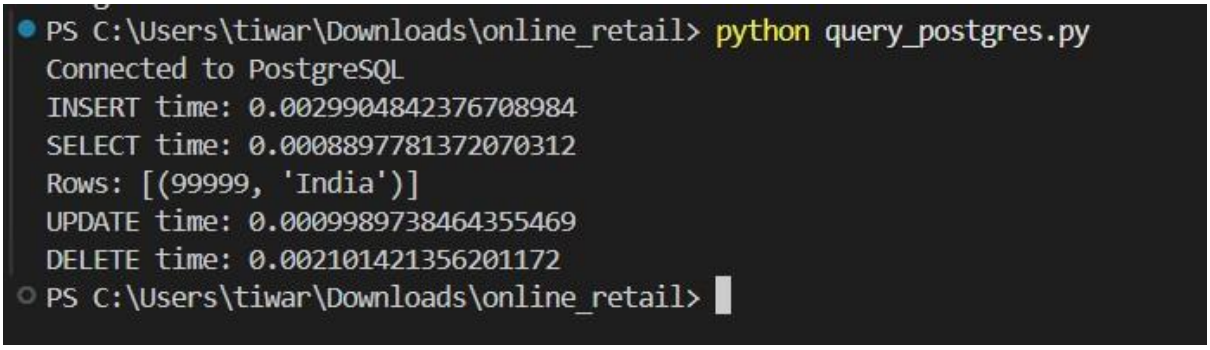
MongoDB Query



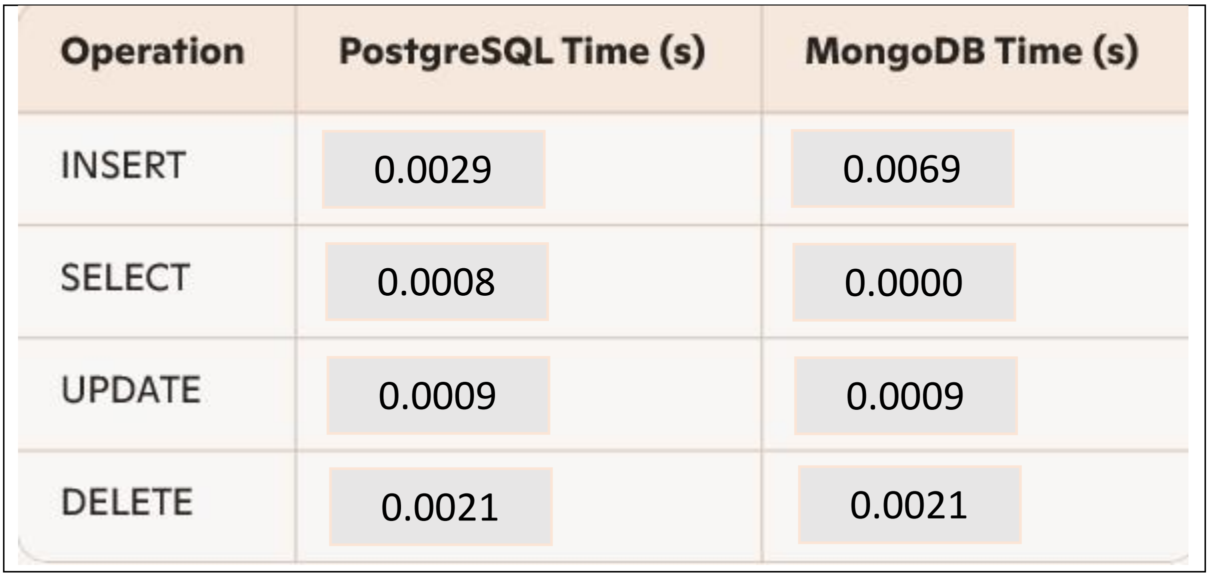
```
PS C:\Users\tiwar\Downloads\online_retail> python query_mango.py
Connected to MongoDB
MongoDB INSERT: 0.006984233856201172
MongoDB SELECT: 0.0
Documents: [{'_id': ObjectId('68d930806fe0867a1716a566'), 'CustomerID': 99999, 'Country': 'India', 'Invoices': []}]
MongoDB UPDATE: 0.0009980201721191406
MongoDB DELETE: 0.001024484634399414
PS C:\Users\tiwar\Downloads\online_retail>
```

PostgreSQL Query



```
PS C:\Users\tiwar\Downloads\online_retail> python query_postgres.py
Connected to PostgreSQL
INSERT time: 0.0029904842376708984
SELECT time: 0.0008897781372070312
Rows: [(99999, 'India')]
UPDATE time: 0.0009989738464355469
DELETE time: 0.002101421356201172
PS C:\Users\tiwar\Downloads\online_retail>
```

CRUD Operation Timings:



Operation	PostgreSQL Time (s)	MongoDB Time (s)
INSERT	0.0029	0.0069
SELECT	0.0008	0.0000
UPDATE	0.0009	0.0009
DELETE	0.0021	0.0021

Schema Comparison

PostgreSQL (Relational Model)

- **Tables**: Customers, Products, Invoices, InvoiceItems
- **Structure**: Normalized to 2nd Normal Form
- **Joins**: Required to reconstruct full invoice details
- **Integrity**: Enforced via foreign keys and constraints

MongoDB (Document Model)

- **Collections**: invoices (transaction-centric), customers (customer-centric)
- **Structure**: Nested documents with embedded arrays
- **Joins**: Not required — all invoice data stored in a single document
- **Flexibility**: Schema-less, allows dynamic fields and nesting

Query Complexity

- **PostgreSQL:**
- Requires multiple joins for invoice reconstruction
- SQL syntax is strict and verbose - Ideal for structured,

tabular data - **MongoDB:**

- Uses aggregation pipelines for analytics
- Easier for hierarchical data
- Less intuitive for relational-style joins

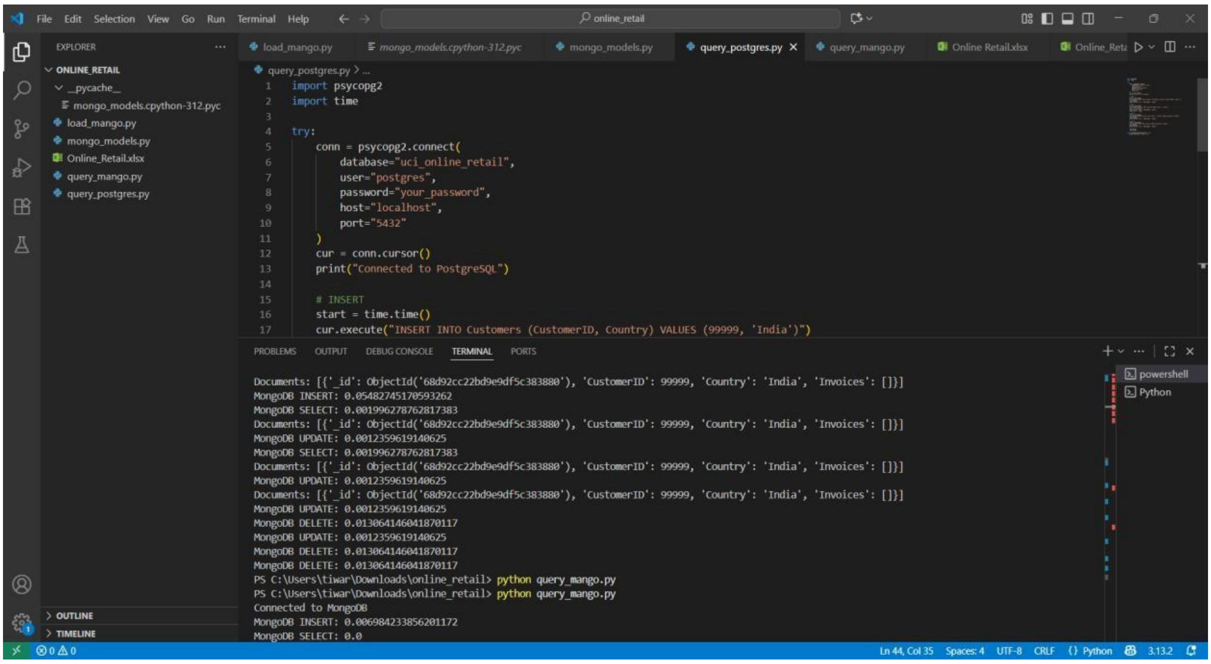
Flexibility and Performance Insights

- **MongoDB:**
- Faster for SELECT and UPDATE due to schema-less design
- Ideal for nested, hierarchical data like invoices with items
- Easier to scale horizontally - Supports flexible document formats
- **PostgreSQL:**
- Strong schema enforcement ensures data integrity
- Better suited for transactional systems with strict relationships
- Joins can slow down complex queries
- Easier to enforce constraints and normalization

Conclusion

MongoDB offers speed and flexibility for document-style data, while PostgreSQL provides structure and reliability for normalized models. The choice between them depends on the nature of the data and the requirements of the application.

Files Structure:



```python
import psycopg2
import time

try:
    conn = psycopg2.connect(
        database="uci_online_retail",
        user="postgres",
        password="your_password",
        host="localhost",
        port="5432"
    )
    cur = conn.cursor()
    print("Connected to PostgreSQL")

    # INSERT
    start = time.time()
    cur.execute("INSERT INTO Customers (CustomerID, Country) VALUES (99999, 'India')")
```

```
Documents: [{'_id': ObjectId('68d92cc22bd9e9df5c383880'), 'CustomerID': 99999, 'Country': 'India', 'Invoices': []}]
MongoDB INSERT: 0.05482745170593262
MongoDB SELECT: 0.001996278762817383
Documents: [{'_id': ObjectId('68d92cc22bd9e9df5c383880'), 'CustomerID': 99999, 'Country': 'India', 'Invoices': []}]
MongoDB UPDATE: 0.0012359619140625
MongoDB SELECT: 0.001996278762817383
Documents: [{'_id': ObjectId('68d92cc22bd9e9df5c383880'), 'CustomerID': 99999, 'Country': 'India', 'Invoices': []}]
MongoDB UPDATE: 0.0012359619140625
Documents: [{'_id': ObjectId('68d92cc22bd9e9df5c383880'), 'CustomerID': 99999, 'Country': 'India', 'Invoices': []}]
MongoDB UPDATE: 0.0012359619140625
MongoDB DELETE: 0.013064146041870117
MongoDB UPDATE: 0.0012359619140625
MongoDB DELETE: 0.013064146041870117
MongoDB DELETE: 0.013064146041870117
PS C:\Users\tiwar\Downloads\online_retail> python query_mango.py
PS C:\Users\tiwar\Downloads\online_retail> python query_mango.py
Connected to MongoDB
MongoDB INSERT: 0.006984233856201172
MongoDB SELECT: 0.0
```