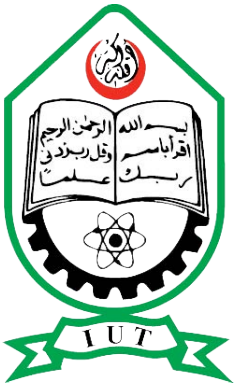
Islamic University of Technology



RELATIONAL DATABASE MANAGEMENT SYSTEM LAB

CSE 4508

Lab 5: Complex Data Types

Author:
Ahmed M. S. Albreem (210041258)

November 5, 2024

Task 1: Parse JSON Data

Objective: Parse the provided JSON data and print the store's `"name"` and its `"city"` location.

Explanation

This task involves reading a JSON object containing details about a store, such as its name, location, and departments. The goal is to extract and display the store's name and the city where it is located. JSON (JavaScript Object Notation) is a lightweight data interchange format that's easy for humans to read and write, and easy for machines to parse and generate.

Code

```
import json

# Load the JSON data
json_data = '''{
    "store": {
        "name": "Tech Hub",
        "location": {
            "city": "San Francisco",
            "address": {
                "street": "123 Market St",
                "postalCode": "94103"
            }
        },
        "departments": [
            {
                "name": "Electronics",
                "products": [
                    {
                        "id": 201,
                        "name": "Laptop",
                        "brand": "BrandX",
                        "price": 999.99,
                        "specifications": {
                            "processor": "Intel i7",
                            "memory": "16GB",
                            "storage": "512GB SSD"
                        }
                    },
```

```
                    {
                        "id": 202,
                        "name": "Smartphone",
                        "brand": "BrandY",
                        "price": 799.99,
                        "specifications": {
                            "screenSize": "6.1 inches",
                            "battery": "4000mAh",
                            "camera": "12MP"
                        }
                    }
                ]
            },
            {
                "name": "Accessories",
                "products": [
                    {
                        "id": 301,
                        "name": "Wireless Mouse",
                        "brand": "BrandZ",
                        "price": 29.99,
                        "specifications": {
                            "batteryLife": "12 months",
                            "connectivity": "Bluetooth"
                        }
                    },
                    {
                        "id": 302,
                        "name": "Keyboard",
                        "brand": "BrandX",
                        "price": 49.99,
                        "specifications": {
                            "layout": "QWERTY",
                            "connectivity": "Wireless"
                        }
                    }
                ]
            }
        ]
    }
}
'''

data = json.loads(json_data)

# Extract the store's name and city location
```

```python
store_name = data["store"]["name"]
city = data["store"]["location"]["city"]

print(f"Store Name: {store_name}")
print(f"City: {city}")


# what i learn at all the tasks deoumented here in initalization of the tasks
# i just have been rembered by teh python strucuture on interaction with json
# which the head of the stucture of the  json is the key of the json object presented as [] key
# if i want to extract one of the instead head i can use [] the looped on the structured
# else if i can't expect the strucuture i can make the biggest loop to check the value on the strucutere as next on the heads of it
# the heads present with it's name with for the next one
# next() is from the json imported on the pyhton
# product defined as the strucure as variable to hold the data for the each next operation as check condition and so on
```

Output

```
PS C:\Users\ahmed> & C:/Users/ahmed/AppData/Local/Microsoft/
WindowsApps/python3.12.exe "c:/Users/ahmed/Desktop/lab tasks/210041258_task1_.py"
Store Name: Tech Hub
City: San Francisco
PS C:\Users\ahmed>
```

Task 2: Print Product Names and Prices

Objective: Iterate through the products in the JSON data and print each product's name and price.

Explanation

The task requires iterating through a list of products contained in the JSON data to extract and print the name and price of each product. This demonstrates handling nested data structures within JSON.

Code

```
import json

json_data = '''{
    "store": {
        "name": "Tech Hub",
        "location": {
            "city": "San Francisco",
            "address": {
                "street": "123 Market St",
                "postalCode": "94103"
            }
        },
        "departments": [
            {
                "name": "Electronics",
                "products": [
                    {
                        "id": 201,
                        "name": "Laptop",
                        "brand": "BrandX",
                        "price": 999.99,
                        "specifications": {
                            "processor": "Intel i7",
                            "memory": "16GB",
                            "storage": "512GB SSD"
                        }
                    },
                    {
                        "id": 202,
                        "name": "Smartphone",
                        "brand": "BrandY",
                        "price": 799.99,
                        "specifications": {
                            "screenSize": "6.1 inches",
                            "battery": "4000mAh",
                            "camera": "12MP"
                        }
                    }
                ]
            },
            {
                "name": "Accessories",
                "products": [
                    {
```

```
                        "id": 301,
                        "name": "Wireless Mouse",
                        "brand": "BrandZ",
                        "price": 29.99,
                        "specifications": {
                            "batteryLife": "12 months",
                            "connectivity": "Bluetooth"
                        }
                    },
                    {
                        "id": 302,
                        "name": "Keyboard",
                        "brand": "BrandX",
                        "price": 49.99,
                        "specifications": {
                            "layout": "QWERTY",
                            "connectivity": "Wireless"
                        }
                    }
                ]
            }
        ]
    }
}
'''
data = json.loads(json_data)

#Task 2: Access and print the names and prices of all products in the "Electronics" department.
# Extract the Electronics department
electronics = next(dept for dept in data["store"]["departments"] if dept["name"] == "Electronics")

# Print names and prices of products in Electronics
for product in electronics["products"]:
    print(f"Product Name: {product['name']}, Price: {product['price']}")
```

Output

```
PS C:\Users\ahmed> & C:/Users/ahmed/AppData/Local/Microsoft/
WindowsApps/python3.12.exe "c:/Users/ahmed/Desktop/lab tasks/210041258_task2.py"
Product Name: Laptop, Price: 999.99
Product Name: Smartphone, Price: 799.99
```

```
PS C:\Users\ahmed>
```

Task 3: Print Specific Product Details

Objective: Print details of the "Wireless Mouse" product, including brand and specifications.

Explanation

This task focuses on retrieving and displaying detailed information about a specific product. The details include the product's brand and specifications, such as battery life and connectivity options.

Code

```
import json


json_data = '''{
    "store": {
        "name": "Tech Hub",
        "location": {
            "city": "San Francisco",
            "address": {
                "street": "123 Market St",
                "postalCode": "94103"
            }
        },
        "departments": [
            {
                "name": "Electronics",
                "products": [
                    {
                        "id": 201,
                        "name": "Laptop",
                        "brand": "BrandX",
                        "price": 999.99,
                        "specifications": {
                            "processor": "Intel i7",
                            "memory": "16GB",
                            "storage": "512GB SSD"
                        }
```

```
                    },
                    {
                        "id": 202,
                        "name": "Smartphone",
                        "brand": "BrandY",
                        "price": 799.99,
                        "specifications": {
                            "screenSize": "6.1 inches",
                            "battery": "4000mAh",
                            "camera": "12MP"
                        }
                    }
                ]
            },
            {
                "name": "Accessories",
                "products": [
                    {
                        "id": 301,
                        "name": "Wireless Mouse",
                        "brand": "BrandZ",
                        "price": 29.99,
                        "specifications": {
                            "batteryLife": "12 months",
                            "connectivity": "Bluetooth"
                        }
                    },
                    {
                        "id": 302,
                        "name": "Keyboard",
                        "brand": "BrandX",
                        "price": 49.99,
                        "specifications": {
                            "layout": "QWERTY",
                            "connectivity": "Wireless"
                        }
                    }
                ]
            }
        ]
    }
}
'''
data = json.loads(json_data)

```

```python
#Task 3: Extract and print the "brand" and "specifications" of the product with the name "Wireless Mouse".

# Extract the Accessories department
accessories = next(dept for dept in data["store"]["departments"] if dept["name"] == "Accessories")

# Find the Wireless Mouse
wireless_mouse = next(product for product in accessories["products"] if product["name"] == "Wireless Mouse")

# Print brand and specifications
print(f"Brand: {wireless_mouse['brand']}")
print("Specifications:")
for key, value in wireless_mouse["specifications"].items():
    print(f"  {key}: {value}")
```

Output

```
PS C:\Users\ahmed> & C:/Users/ahmed/AppData/Local/
Microsoft/WindowsApps/python3.12.exe
"c:/Users/ahmed/Desktop/lab tasks/210041258_task3.py"
Brand: BrandZ
Specifications:
  batteryLife: 12 months
  connectivity: Bluetooth
PS C:\Users\ahmed>
```

Task 4: Print the JSON Data

Objective: Display the entire JSON structure of the store.

Explanation

The aim of this task is to print the complete JSON structure, showcasing all the store's details, including name, location, departments, and products. This helps in understanding the overall structure and content of the JSON data.

Code

```
import json


json_data = '''{
    "store": {
        "name": "Tech Hub",
        "location": {
            "city": "San Francisco",
            "address": {
                "street": "123 Market St",
                "postalCode": "94103"
            }
        },
        "departments": [
            {
                "name": "Electronics",
                "products": [
                    {
                        "id": 201,
                        "name": "Laptop",
                        "brand": "BrandX",
                        "price": 999.99,
                        "specifications": {
                            "processor": "Intel i7",
                            "memory": "16GB",
                            "storage": "512GB SSD"
                        }
                    },
                    {
                        "id": 202,
                        "name": "Smartphone",
                        "brand": "BrandY",
                        "price": 799.99,
                        "specifications": {
                            "screenSize": "6.1 inches",
                            "battery": "4000mAh",
                            "camera": "12MP"
                        }
                    }
                ]
            },
            {
                "name": "Accessories",
                "products": [
                    {
```

```
                            "id": 301,
                            "name": "Wireless Mouse",
                            "brand": "BrandZ",
                            "price": 29.99,
                            "specifications": {
                                "batteryLife": "12 months",
                                "connectivity": "Bluetooth"
                            }
                        },
                        {
                            "id": 302,
                            "name": "Keyboard",
                            "brand": "BrandX",
                            "price": 49.99,
                            "specifications": {
                                "layout": "QWERTY",
                                "connectivity": "Wireless"
                            }
                        }
                    ]
                }
            ]
        }
}
'''
# Load the JSON data
data = json.loads(json_data)

# Find the Electronics department
electronics = next(department for department in data["store"]["
    departments"] if department["name"] == "Electronics")

# Find the Smartphone in Electronics
smartphone = next(product for product in electronics["products"] if
    product["name"] == "Smartphone")

# Update the price
smartphone["price"] = 749.99

# Print the updated JSON
updated_json = json.dumps(data, indent=4)
print(updated_json)
```

Output

```
PS C:\Users\ahmed> & C:/Users/ahmed/AppData/Local/Microsoft/WindowsApps/python3.12.ex
{
    "store": {
        "name": "Tech Hub",
        "location": {
            "city": "San Francisco",
            "address": {
                "street": "123 Market St",
                "postalCode": "94103"
            }
        },
        "departments": [
            {
                "name": "Electronics",
                "products": [
                    {
                        "id": 201,
                        "name": "Laptop",
                        "brand": "BrandX",
                        "price": 999.99,
                        "specifications": {
                            "processor": "Intel i7",
                            "memory": "16GB",
                            "storage": "512GB SSD"
                        }
                    },
                    {
                        "id": 202,
                        "name": "Smartphone",
                        "brand": "BrandY",
                        "price": 749.99,
                        "specifications": {
                            "screenSize": "6.1 inches",
                            "battery": "4000mAh",
                            "camera": "12MP"
                        }
                    }
```

```
                ]
            },
            {
                "name": "Accessories",
                "products": [
                    {
                        "id": 301,
                        "name": "Wireless Mouse",
                        "brand": "BrandZ",
                        "price": 29.99,
                        "specifications": {
                            "batteryLife": "12 months",
                            "connectivity": "Bluetooth"
                        }
                    },
                    {
                        "id": 302,
                        "name": "Keyboard",
                        "brand": "BrandX",
                        "price": 49.99,
                        "specifications": {
                            "layout": "QWERTY",
                            "connectivity": "Wireless"
                        }
                    }
                ]
            }
        ]
    }
}
PS C:\Users\ahmed>
```


SQLite Task

Objective: Perform database operations using SQLAlchemy and log the output.

Explanation

This task involves creating a database using SQLite and performing various operations such as inserting, updating, selecting, and deleting records. SQLAlchemy is used to interact with the SQLite database in Python. Logging is enabled to monitor these operations.

Code

```

from sqlalchemy.orm import sessionmaker
from sqlalchemy import create_engine, Column, Integer, String, Float
from sqlalchemy.ext.declarative import declarative_base

engine = create_engine('sqlite:///books.db', echo=True)
Base = declarative_base()

class Book(Base):
    __tablename__ = 'books'

    id = Column(Integer, primary_key=True)
    title = Column(String, nullable=False)
    author = Column(String, nullable=False)
    published_year = Column(Integer, nullable=False)
    price = Column(Float, nullable=False)

Base.metadata.create_all(engine)

Session = sessionmaker(bind=engine)
session = Session()

# Tracking the status of each book
book_status = {}

# Add a new book
new_book = Book(title="The Great Gatsby", author="F. Scott
    Fitzgerald", published_year=1925, price=10.99)
session.add(new_book)
session.commit()
book_status[new_book.title] = 'added'

books = session.query(Book).all()
for book in books:
    status = book_status.get(book.title, 'unknown')
```

```
    print(f"Id : {book.id},Title: {book.title}, Author: {book.author}, Year: {book.published_year}, Price: {book.price}, Status: {status}")

book_to_update = session.query(Book).filter_by(title="The Great Gatsby").first()
if book_to_update:
    book_to_update.price = 12.99
    session.commit()
    book_status[book_to_update.title] = 'updated'

books = session.query(Book).all()
for book in books:
    status = book_status.get(book.title, 'unknown')
    print(f"Id : {book.id},Title: {book.title}, Author: {book.author}, Year: {book.published_year}, Price: {book.price}, Status: {status}")

book_to_delete = session.query(Book).filter_by(title="The Great Gatsby").first()
if book_to_delete:
    session.delete(book_to_delete)
    session.commit()
    book_status[book_to_delete.title] = 'deleted'

books = session.query(Book).all()
for book in books:
    status = book_status.get(book.title, 'unknown')
    print(f"Id : {book.id},Title: {book.title}, Author: {book.author}, Year: {book.published_year}, Price: {book.price}, Status: {status}")

if "The Great Gatsby" in book_status:
    print(f"Title: The Great Gatsby, Status: {book_status['The Great Gatsby']}")




# creation of the session is important
# i see that the session is created and commit each time
# i see also that table on createion with attrubute initalization on __tablename__ = 'books' as branch on the  database
# i relise the  use of the declarative_base() function to create a base class for declarative class definitions
```

```
69 # i see that the use of the Column() function to define the columns
       of the table
70 # i see that the use of the Integer, String, Float to define the
       type of the columns and of curse the nullable attrubute and the
       nulablity of each column defined
71 # i also make the stauts of the book to check the operation of the
       book to confirm the operation side becuase on sqllite i configure
        that
72 # it depneds on extra data showen up on it which its musn't be
       appered on the output
73 # i added on it  to be showen for any one will see the code so on
```

Output

```
PS C:\Users\ahmed> & C:/Users/ahmed/AppData/Local/Microsoft/WindowsApps/python3.12.ex
c:\Users\ahmed\Desktop\lab tasks\210041258_sqlite.py:6: MovedIn20Warning: The ‘‘decla
  Base = declarative_base()
2024-11-05 09:11:34,601 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2024-11-05 09:11:34,601 INFO sqlalchemy.engine.Engine PRAGMA main.table_info("books")
2024-11-05 09:11:34,601 INFO sqlalchemy.engine.Engine [raw sql] ()
2024-11-05 09:11:34,602 INFO sqlalchemy.engine.Engine COMMIT
2024-11-05 09:11:34,603 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2024-11-05 09:11:34,608 INFO sqlalchemy.engine.Engine INSERT INTO books (title, autho
2024-11-05 09:11:34,608 INFO sqlalchemy.engine.Engine [generated in 0.00075s] ('The G
2024-11-05 09:11:34,610 INFO sqlalchemy.engine.Engine COMMIT
2024-11-05 09:11:34,614 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2024-11-05 09:11:34,618 INFO sqlalchemy.engine.Engine SELECT books.id AS books_id, bo

FROM books
WHERE books.id = ?
2024-11-05 09:11:34,618 INFO sqlalchemy.engine.Engine [generated in 0.00100s] (1,)
2024-11-05 09:11:34,619 INFO sqlalchemy.engine.Engine SELECT books.id AS books_id, bo

FROM books
2024-11-05 09:11:34,621 INFO sqlalchemy.engine.Engine [generated in 0.00058s] ()
Id : 1,Title: The Great Gatsby, Author: F. Scott Fitzgerald, Year: 1925, Price: 10.99
2024-11-05 09:11:34,622 INFO sqlalchemy.engine.Engine SELECT books.id AS books_id, bo

FROM books
```

```
WHERE books.title = ?
 LIMIT ? OFFSET ?
2024-11-05 09:11:34,623 INFO sqlalchemy.engine.Engine [generated in 0.00051s] ('The G
2024-11-05 09:11:34,624 INFO sqlalchemy.engine.Engine UPDATE books SET price=? WHERE
2024-11-05 09:11:34,625 INFO sqlalchemy.engine.Engine [generated in 0.00060s] (12.99,
2024-11-05 09:11:34,626 INFO sqlalchemy.engine.Engine COMMIT
2024-11-05 09:11:34,629 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2024-11-05 09:11:34,632 INFO sqlalchemy.engine.Engine SELECT books.id AS books_id, bo

FROM books
WHERE books.id = ?
2024-11-05 09:11:34,632 INFO sqlalchemy.engine.Engine [cached since 0.01468s ago] (1,
2024-11-05 09:11:34,633 INFO sqlalchemy.engine.Engine SELECT books.id AS books_id, bo

FROM books
2024-11-05 09:11:34,633 INFO sqlalchemy.engine.Engine [cached since 0.01398s ago] ()
Id : 1,Title: The Great Gatsby, Author: F. Scott Fitzgerald, Year: 1925, Price: 12.99
2024-11-05 09:11:34,634 INFO sqlalchemy.engine.Engine SELECT books.id AS books_id, bo

FROM books
WHERE books.title = ?
 LIMIT ? OFFSET ?
2024-11-05 09:11:34,634 INFO sqlalchemy.engine.Engine [cached since 0.01334s ago] ('T
2024-11-05 09:11:34,634 INFO sqlalchemy.engine.Engine DELETE FROM books WHERE books.i
2024-11-05 09:11:34,634 INFO sqlalchemy.engine.Engine [generated in 0.00032s] (1,)
2024-11-05 09:11:34,634 INFO sqlalchemy.engine.Engine COMMIT
2024-11-05 09:11:34,640 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2024-11-05 09:11:34,641 INFO sqlalchemy.engine.Engine SELECT books.id AS books_id, bo

FROM books
2024-11-05 09:11:34,641 INFO sqlalchemy.engine.Engine [cached since 0.02129s ago] ()
Title: The Great Gatsby, Status: deleted
PS C:\Users\ahmed>
 *  History restored
```