DatacubeObject Functionality Report

Introduction
The DatacubeObject class provides a flexible and chainable interface for interacting with a remote Web Coverage Processing Service (WCPS) server to manipulate and retrieve data from a specified coverage.

Initialization
Upon instantiation of a DatacubeObject, the following parameters are provide

dbc: An instance of DatabaseConnectionObject managing the connection to the WCPS server.
coverage_name: The name of the coverage on which operations will be performed.

Supported Operations

Subset Operation (subset):
Functionality: Adds a subset operation for a specific dimension.
Parameters:
dimension: The dimension (e.g., 'axis0', 'axis1') on which the subset is applied.
range: The range specified in the format "start:end" to subset the dimension.
Return Value: Allows for method chaining.

Execute Operation (execute):
Functionality: Generates and executes a WCPS query based on accumulated operations.
Return Value: Returns the response from the WCPS server or None if an error occurs.
Conditional Filtering (add_condition):
Functionality: Adds a conditional filtering operation to the query.
Parameters:
condition: The condition to be applied.
Return Value: Allows for method chaining.

Aggregation Operation (aggregate):
Functionality: Adds an aggregation operation (e.g., mean, max, sum) to the query.
Parameters:
operation: The aggregation operation to be applied.
Return Value: Allows for method chaining.

Overloaded Indexing (__getitem__):
Functionality: Overloads the slicing operator to enable easier specification of subsetting operations.
Parameters:
slices: A tuple of slice objects representing ranges for each dimension.
Return Value: Allows for method chaining.

Example Usage

```
from database_connection import DatabaseConnectionObject
from datacube import DatacubeObject

def main():
    server_url = "https://ows.rasdaman.org/rasdaman/ows"

    # Creating an instance of DatabaseConnectionObject
    dbc = DatabaseConnectionObject(server_url)

    try:
        # Establish connection to the server
        dbc.establish_connection()

        # Create a DatacubeObject for a specific coverage
        coverage_name = "my_coverage"
        datacube = DatacubeObject(dbc, coverage_name)

        # Perform datacube operations
        # Example: Subset and aggregate operations
        result = datacube['axis0':0, 'axis1':100].aggregate('mean').execute()

        if result is not None:
            print("Query executed successfully. Received response:")
            print(result)
        else:
            print("Error executing query.")

    except Exception as e:
        print(f"An error occurred: {e}")

if __name__ == "__main__":
    main()
```

Conclusion
The DatacubeObject class simplifies the process of interacting with a WCPS server by allowing users to chain operations fluently, build complex queries incrementally, and retrieve data from specified coverages efficiently. The design promotes readability and flexibility in data manipulation tasks