Essential Redis Commands using redis-py and Redis Stack

This guide demonstrates key Redis commands using the Python connector (redis-py) and includes examples for Redis Stack modules such as RedisJSON and RedisSearch.

1. Import and Connection Setup

```
import redis

# Connect to the Redis server (adjust host/port as needed)
client = redis.Redis(host='localhost', port=6379, decode_responses=True)

# Set a key-value pair
client.set('name', 'Alice')

# Retrieve the value of a key
print(client.get('name'))

# Initialize a counter and increment it
client.set('counter', 0)
client.incr('counter')
print(client.get('counter'))

# Push elements to a list (LPUSH adds to the head)
client.lpush('mylist', 'value1')
client.lpush('mylist', 'value2')

# Retrieve all elements from the list
print(client.lrange('mylist', 0, -1))

# Add members to a set
client.sadd('myset', 'apple', 'banana', 'cherry')

# Retrieve all members of the set
print(client.smembers('myset'))

# Set multiple fields in a hash
client.hset('user:1000', mapping={'name': 'Alice', 'age': 30})

# Retrieve all fields and values of the hash
print(client.hgetall('user:1000'))

# Add members with scores to a sorted set
client.zadd('leaders', {'Alice': 100, 'Bob': 95})

# Retrieve all members with their scores from the sorted set
print(client.zrange('leaders', 0, -1, withscores=True))

# Store a JSON document using RedisJSON
client.execute_command('JSON.SET', 'doc:1', '.', '{"name": "Alice", "age": 30}')

# Retrieve the JSON document
print(client.execute_command('JSON.GET', 'doc:1'))

# Create a full-text search index on a JSON field
client.execute_command(
    'FT.CREATE', 'idx:users', 'ON', 'JSON',
    'SCHEMA', '$.name', 'AS', 'name', 'TEXT'
)

# Index a document using RedisJSON
client.execute_command('JSON.SET', 'user:1', '.', '{"name": "Bob", "age": 25}')

# Search the index for the term 'Bob'
result = client.execute_command('FT.SEARCH', 'idx:users', 'Bob')
print(result)
```