data_storage_library.py

```python
import json
import os
import pickle
```

The DataStorageLibrary class will be used to store and retrieve data in multiple formats and destinations.

The library support the following functinality:

1. Record insertion/ Batch insertion
2. Record query/retrieval
3. Query filters (equality operations only), limit & offset
4. Update and delete operations

```python
class DataStorageLibrary:
```

init

```python
    def __init__(self, storage_format, destination):
```

This function will initilize the storage_format and destination, it will call when object is created

storage_format : Which formats of data can be stored like (Json, xml) **destination :** where to store the data (local, aws, ftp) **records :** initilize dictionary to save the data locally

```python
        self.storage_format = storage_format
        self.destination = destination
        self.records = {}
```

insert

```python
    def insert(self, record):
```

This function will insert single record in local storage.

record : User will send the single record in key:value format and it will save.

```python
        self.records[record['id']] = record
```

get

```python
    def get(self, id):
```

This function will get single record from local storage.

id : User will send the Id of record and it will get the record the of that ID

```python
        return self.records[id]
```

get_all

```python
    def get_all(self):
```

This function will get all record from local storage, irrespective of the record.

```python
        return self.records
```

delete

```python
    def delete(self, id):
```

This function will delete single record from local storage.

id : User will send the Id of record and it will delete the record the of that ID

```python
        del self.records[id]
```

update

```python
    def update(self, id, record):
```

This function will update single record in local storage. User will send Id and record to update the record

id : User will send the Id of record that wants to update
record : User will send the updated record here

```python
        self.records[id] = record
```

insert_batch

```python
    def insert_batch(self, records):
```

This function will insert list of records in local storage

records : List of records send by user and it will insert in local storage

```python
        for record in records:
            self.records[record['id']] = record
```

get_by_filter

This function will get records by filter (name, age etc)

records : If user wants all records whose name is aqeel etc, it will get all the records where name is aqeel

```python
def get_by_filter(self, filter_dict):

    return [record for record in self.records.values() if
```

filter_record

This function will get records by filter (name, age etc)

records : If user wants all records whose name is aqeel etc, it will get all the records where name is aqeel

```python
def filter_record(self, record, filter_dict):

    for key, value in filter_dict.items():
        if record[key] != value:
            return False
    return True
```

get_by_filter_with_limit_and_offset

This function will get records by filter (name, age etc) and limit + offset

filter_dict : User will pass the filter on which the user want to search for example {name : ‘aqeel’}
limit : How many records user wants from filtered records
offset : From where user want to start
Example : Total filtered records are 5 and user limit 3 and offset is 1 then it will fetch records id (2,3,4)

```python
def get_by_filter_with_limit_and_offset(self, filter_dict,

    records = self.get_by_filter(filter_dict)
    return records[offset:offset + limit]
```