Preliminary Test Document Project 3

11/25/2024

1. Overview

This document demonstrates the operation of the blocked sequence set program, including adding and deleting records. It uses a small dataset and tests operations under different scenarios to validate the functionality. The modules under test include:

- Block
- Buffer
- HeaderRecord
- Index
- Main

2. Test Data

2.1 Dataset

zip_code,city,state,latitude,longitude

12345,Springfield,IL,39.7817,-89.6501

23456,Riverside,CA,33.9533,-117.3961

34567,Lakewood,OH,41.4819,-81.7984

45678,Maplewood,MN,44.9530,-93.0275

56789,Oakland,CA,37.8044,-122.2711

67890,Seattle,WA,47.6062,-122.3321

78901,Miami,FL,25.7617,-80.1918

89012,Denver,CO,39.7392,-104.9903

90123,Atlanta,GA,33.7490,-84.3880

01234,Boston,MA,42.3601,-71.0589

3. Test Cases

3.1 Adding Records

Test Case 1: Adding a Record Requiring No Block Split

1. **Description**: Add a record to a block that has space available.
2. **Input**:
    - Record: 67891,Chicago,IL,41.8781,-87.6298
    - Block Size: 6 records
3. **Expected Output**:
    - Record is added to an existing block without a split.
4. **Logs:**

    [INFO] Adding record: 67891,Chicago,IL,41.8781,-87.6298

    [INFO] Adding record to Block 1. Block has enough space.

    [INFO] Record added successfully.

    Blocks after addition:

    Block 1: 12345, 23456, 34567, 45678, 56789, 67890, 67891

    Block 2: 78901, 89012, 90123, 01234

Test Case 2: Adding a Record Requiring a Block Split

1. **Description**: Add a record to a block that is already at capacity, triggering a block split.
2. **Input**:
    - Record: 78902,Dallas,TX,32.7767,-96.7970
    - Block Size: 6 records

3. **Expected Output**:
    - Block splits into two blocks.
    - New record is placed in the correct block.
    - Avail list is updated.
4. **Logs:**

    [INFO] Adding record: 78902,Dallas,TX,32.7767,-96.7970

    [INFO] Block 2 is full. Performing block split.

    [INFO] Creating new block with RBN: 3

    [INFO] Record added to Block 3. Avail list updated.

    Blocks after addition:

    Block 1: 12345, 23456, 34567, 45678, 56789, 67890

    Block 2: 67891, 78901, 89012

    Block 3: 90123, 01234, 78902

3.2 Deleting Records

Test Case 3: Deleting a Record Requiring No Block Deletion or Redistribution

1. **Description**: Delete a record from a block that retains sufficient records post-deletion.
2. **Input**:
    - Record to delete: 56789,Oakland,CA,37.8044,-122.2711
    - Block Size: 6 records
3. **Expected Output**:
    - Record is deleted.
    - Block retains other records without redistribution.
4. **Logs:**

    [INFO] Deleting record: 56789,Oakland,CA,37.8044,-122.2711

[INFO] Record found in Block 1. Deleting.
[INFO] Record deleted successfully. No redistribution required.

Blocks after deletion:
Block 1: 12345, 23456, 34567, 45678, 67890
Block 2: 67891, 78901, 89012
Block 3: 90123, 01234, 78902

Test Case 4: Deleting a Record Requiring Block Redistribution Without Merge

1. **Description**: Delete a record from a block that becomes under full, triggering redistribution with adjacent blocks.
2. **Input**:
    - Record to delete: 78901,Miami,FL,25.7617,-80.1918
    - Block Size: 6 records
3. **Expected Output**:
    - Records are redistributed among adjacent blocks to maintain capacity.
    - No blocks are merged.
4. **Logs:**

    [INFO] Deleting record: 78901,Miami,FL,25.7617,-80.1918
    [INFO] Record found in Block 2. Deleting.
    [INFO] Block 2 is underfull. Redistributing records with adjacent blocks.
    [INFO] Redistribution complete.

    Blocks after redistribution:
    Block 1: 12345, 23456, 34567, 45678, 67890
    Block 2: 67891, 89012, 90123
    Block 3: 01234, 78902

Test Case 5: Deleting a Record Requiring a Block Merge

1. **Description**: Delete a record that results in a block becoming empty, triggering a merge with the logically rightmost block.
2. **Input**:
    - Record to delete: 67890,Seattle,WA,47.6062,-122.3321
    - Block Size: 6 records
3. **Expected Output**:
    - The logically rightmost block is cleared.
    - Cleared block is added to the avail list.
4. **Logs:**
    [INFO] Deleting record: 67890,Seattle,WA,47.6062,-122.3321
    [INFO] Record found in Block 1. Deleting.
    [INFO] Block 3 is underfull and logically rightmost. Merging with adjacent blocks.
    [INFO] Block 3 cleared and added to avail list.

    Blocks after merge:
    Block 1: 12345, 23456, 34567, 45678
    Block 2: 67891, 89012, 90123
    Avail List: Block 3

3.3 Integration Tests

Full Workflow Test

1. **Description:** Validate the program's full workflow:
    - Create block file.
    - Parse block file.
    - Dump blocks in physical and logical order.
    - Add and delete records as described in the above scenarios.
2. **Expected Output:**
    - Successful execution of all steps.

- Correct updates to blocks and avail list.

3. **Logs**

   [INFO] Dumping blocks in physical order:

   RBN: 1 -> 12345, 23456, 34567, 45678

   RBN: 2 -> 67891, 89012, 90123

   [INFO] Dumping blocks in logical order:

   RBN: 1 -> 12345, 23456, 34567, 45678

   RBN: 2 -> 67891, 89012, 90123

   [INFO] Querying Block 1:

   Available: No

   Records: 12345, 23456, 34567, 45678

   Predecessor RBN: -1

   Successor RBN: 2

4. Modules and Functions

4.1 Block Module

Tests

- **Block File Creation (createBlockFile)**
  - **Description:** Verify that a block file is created correctly from the input CSV.
  - **Input:** us_postal_codes.csv, Block Size: 6 records
  - **Expected Output:** A block file is created with records divided into blocks.
  - **Logs:**

    [INFO] Reading CSV file: us_postal_codes.csv

    [INFO] Creating block file: blocks.txt with block size: 6 records

    [INFO] Block file created successfully.

Initial Blocks:

Block 1: 12345, 23456, 34567, 45678, 56789, 67890

Block 2: 78901, 89012, 90123, 01234

- **Parsing and Populating Global blocks Map (parseBlockFile)**
  - **Description:** Verify that the block file is correctly parsed into the global blocks map.
  - **Input:** blocks.txt
  - **Expected Output:** blocks map populated with parsed records.
  - **Logs:**

    [INFO] Parsing block file: blocks.txt

    [INFO] Block file parsed successfully.

    Parsed Blocks:

    RBN: 1 -> 12345, 23456, 34567, 45678, 56789, 67890

    RBN: 2 -> 78901, 89012, 90123, 01234
- **Dumping Blocks (dumpPhysicalOrder and dumpLogicalOrder)**
  - **Description:** Verify that blocks are correctly dumped in physical and logical order.
  - **Input:** Populated blocks map.
  - **Expected Output:** Blocks printed in physical and logical order.
  - **Logs:**

    [INFO] Dumping blocks in physical order:

    RBN: 1 -> 12345, 23456, 34567, 45678, 56789, 67890

    RBN: 2 -> 78901, 89012, 90123, 01234

[INFO] Dumping blocks in logical order:

RBN: 1 -> 12345, 23456, 34567, 45678, 56789, 67890

RBN: 2 -> 78901, 89012, 90123, 01234

4.2 Buffer Module

Tests

- **Reading CSV and Dividing Records into Blocks (read_csv)**
    - **Description:** Verify that records are correctly read from the CSV and divided into blocks.
    - **Input:** us_postal_codes.csv, Block Size: 6 records
    - **Expected Output:** Records divided into blocks stored in blocks.
    - **Logs:**

      [INFO] Reading CSV file: us_postal_codes.csv

      [INFO] Dividing records into blocks with block size: 6 records

      [INFO] Records successfully divided into blocks.

      Blocks:

      Block 1: 12345, 23456, 34567, 45678, 56789, 67890

      Block 2: 78901, 89012, 90123, 01234
- **Processing and Printing Blocks (process_blocks)**
    - **Description:** Verify that records are unpacked and printed correctly from the blocks.
    - **Input:** Populated blocks map.
    - **Expected Output:** Records unpacked and printed.

- **Logs:**

  [INFO] Processing Block 1

  Record: 12345, Springfield, IL, 39.7817, -89.6501

  Record: 23456, Riverside, CA, 33.9533, -117.3961

  [INFO] Processing Block 2

  Record: 78901, Miami, FL, 25.7617, -80.1918

  Record: 89012, Denver, CO, 39.7392, -104.9903

- **Sorting Records (sort_records)**
  - **Description:** Verify that all records in the buffer are sorted by zip code.
  - **Input:** Unsorted blocks map.
  - **Expected Output:** Records sorted in ascending order by zip code.
  - **Logs:**

    [INFO] Sorting records by zip code

    [INFO] Records sorted successfully.

    Sorted Records:

    01234, Boston, MA, 42.3601, -71.0589

    12345, Springfield, IL, 39.7817, -89.6501

4.3 HeaderRecord Module

Tests

- **Writing Header (writeHeader)**
  - **Description**: Verify that metadata is correctly written to a file.

- **Input**: Configured HeaderRecord object.
- **Expected Output**: Metadata written to the specified file.
- **Logs**:

  [INFO] Writing header to file: headerTest_with_header.dat

  [INFO] Header written successfully.

- **Reading Header (readHeader)**
  - **Description**: Verify that metadata is correctly read and parsed from a file.
  - **Input**: File with header metadata.
  - **Expected Output**: Metadata matches the originally written data.
  - **Logs**:

    [INFO] Reading header from file: headerTest_with_header.dat

    [INFO] Header read successfully.

    Parsed Header:

    File Structure Type: blocked_sequence_set

    Version: 1.0

    Block Size: 512 bytes

4.4 Index Module

Tests

- **Extracting Zip Codes and Organizing by Block (processBlockData)**
  - **Description**: Verify that zip codes are correctly extracted from blocks and organized by block in the index file.
  - **Input**: Block file (blocks.txt).

- **Expected Output**: Index file with zip codes and their corresponding block numbers.
- **Logs**:

  [INFO] Extracting zip codes from block file: blocks.txt

  [INFO] Organizing zip codes by block.

  [INFO] Index file created successfully: index.idx