CSCI 3901 Final Project

Name : Yogish Honnadevipura Gopalakrishna(B00928029)

Overview

The PublicationLibrary project is a software solution for managing information about publications in the academic world. The system is made to manage conference and journal publications, with the ability to extend its design for future publications like books and white papers. Additionally, the system has the ability to provide citations in IEEE format.

The primary class in the system is called PublicationLibrary, and it offers ways to add publications, references between publications, publication venues, publishers, and research areas. Information is stored by the system using databases, and this data is permanent across programme executions.

The system can provide answers to many different queries, including getting the details of publications, which papers in a field of study are regarded as seminal, what references are included in a particular article, what research fields an author focuses on, and how to convert citation strings into IEEE citations.

Files and external data

There main files are:\

- PublicationsLibrary -> This method contains several methods to add and retrieve information related to publications, venues, publishers, and research areas from a database.
- AddPublicationToDatabase.java ->This method verifies whether a publication is journal or conference and then adds them to the database.
- AddPublisherToDatabase -> Adds a new publisher to the database with the given identifier and information.
- AddReferenceToDatabase -> This method adds references to the reference identifier in the database. If the identifier already exists, it performs a union of the set of already present references with the given set of references. If the identifier does not exist, it adds the given references to the database.
- AddAreaToDatabase -> This method populates the research areas hierarchy by inserting a new record for a research area and its parent areas.
- AddVenueToDatabase -> This method populates the venue information into the database if the venue name is unique.
- GetPublicationsFromDatabase -> This method retrieves publication details based on a given key.

- IEEECitationConverter -> This method reads input from the user and processes the input file to replace citations with references
- GetAuthorCitations -> This method returns the number of times an author's publications have been cited by other publications.
- GetAuthorResearchAreas -> Returns a set of research areas for a given author if they have published papers having threshold values in that area.
- credentials.prop -> This method contains credentials needed to connect to the database

Choices

While storing references, I have considered to store the references as multi-valued attribute.
While storing Publications, I have stored identifiers with their respective authors in a separate table.
While storing Venues, I have stored venue names with their respective research areas in a separate table.

Assumptions

The author names provided are in correct format, ie, the first letter of their first and last names are capital.
While calling IEEECitationConverter, the input file contains the lines with references that are present in the database.

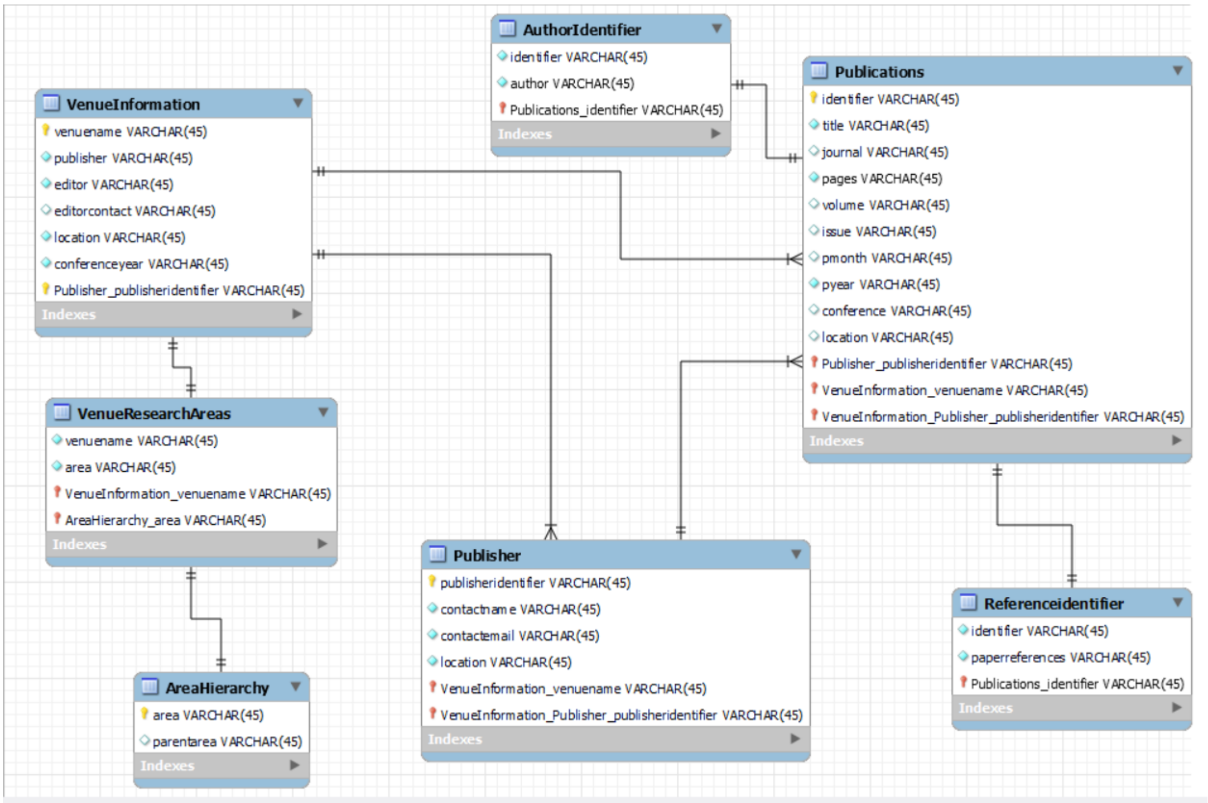
Key algorithms and design elements

Adding data into the database

- The java files namely AddPublication and AddVenue creates two tables in the database:
    1. AddPublication creates a single table to store both journal and conference publications details, while the authors and their identifiers are stored in a separate table for better design
    2. AddVenues creates a table to store venue name and its information in map, while it stores its name and its research areas in another tables

- The java files namely addReference, addPublisher, addArea creates a single table in the database.
    - While adding a Reference using addReference, the identifier should be present in the Publications table, else it will return false.

The database design is as follows:



Retrieving data from the database

GetPublications - This method runs 3 queries

1. Getting author names from the database
2. Getting references from the database
3. Getting Publication details from the database.

Finally an HashMap is created and all the data are stored using .put method of the map using correct key names and then the map is returned.

IEEECitationConverter - This method uses the following method:

1. Read the name of the input and output file by prompting the user for the names.
2. Initialize a HashMap to store article keys and their corresponding reference numbers
3. Read all lines from the input file into a single List
4. Initialize an empty List to hold output lines
5. Initialize a counter variable to keep track of the next reference number
6. For each line in the input file:
    a. Split the line into tokens at each occurrence of "\cite{"
    b. Use a StringBuilder to construct the output line

    c. For each citation token:
        i. Extract the article keys from the substring between the curly braces
        ii. Use a StringBuilder to construct the reference string
        iii. For each article key:
            1. If the article key has been referenced before, use its existing reference number
            2. Otherwise, create a new reference number and add it to the reference map
        iv. Append the reference string and text after the citation to the output line
    d. Add the completed output line to the output List
7. Write all output lines to the output file

Now the reference map contains all the identifiers with its corresponding number, the publication details of these reference identifiers are taken using GetPublications method. Then using the Map returned by the method, the references are formatted in the IEEE format.

GetAuthorCitation - This method runs a query to get the count of the author who was cited in their publications. This method returns 0 if the author is not cited or present in the database.

Limitations

Collaborators and seminalPapers methods are not implemented.

References

https://www.baeldung.com/java-substring
https://www.w3schools.com/java/ref_string_indexof.asp

Test Cases

addPublication

1. **Input Validations and boundary case**
    a. identifier and publication information is not null, returns false.
    b. identifier is an empty string and the publication information is not null, returns false.
    c. identifier is not null and the publication information is null, returns false.
    d. identifier is not null and the publication information is an empty map, returns false.
    e. identifier is not null and the publication information map is missing one or more required keys, returns false.
    f. identifier is not null and the publication information map has all required keys and values, returns true.
    g. Identifier is already present in the database, returns false.
    h. publicationInformation values are duplicate, returns false.

2. **Control Flow**
    a. Valid identifier and valid PublicationInformation
    b. Valid identifier and Invalid PublicationInformation
    c. InValid identifier and valid PublicationInformation
    d. InValid identifier and Invalid PublicationInformation

3. **Data Flow**
    a. calling this method at first
    b. calling this method after addReferences
    c. calling this method after addVenue
    d. Calling this method after addRefernces and addVenue

addReferences

1. **Input Validations and boundary case**
    a. identifier is null and references set is non-null, returns false.
    b. identifier is empty and references set is non-null, returns false.
    c. identifier has a valid value and references set is null, returns false.
    d. Identifier has a valid value and an empty references set, returns true
    e. Identifier has a valid value and a non-empty references set, with invalid values, returns false.
    f. Identifier is non-null and a non-empty references set, with all valid values, returns true.
    g. Adding references multiple times for the same publication, returns true

2. **Control Flow**
    a. Valid references and Valid identifier
    b. Invalid references and valid identifier, returns false

3. **Data Flow**
    a. calling this method at first
    b. calling this method after addPublication

addVenue

1. **Input Validations and boundary case**
    a. venueName is null or empty, returns false
    b. venueInformation is null or empty, returns false
    c. researchAreas is null or empty, returns false
    d. venueInformation should have at least one valid key-value pair, returns true
    e. venueInformation is non-null and non-empty strings, returns true
    f. venueInformation has a key with an empty string value, returns false.
    g. venueInformation has a null key or value, returns false.
    h. venueInformation has characters that exceeds the maximum allowable length, returns false.

2. **Control Flow**
    a. Valid venue name, venue Information
    b. Invalid venue name, venue Information

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after addPublisher

addPublisher

1. **Input Validations and boundary case**
    a. identifier is null or empty, returns false
    b. publisherInformation is null or empty, returns false
    c. publisherInformation has atleast one key-value pair, returns true
    d. publisherInformation is non-null and non-empty strings, returns true
    e. identifier is an empty string, return false.
    f. publisherInformation has a key with an empty string value, returns false.
    g. publisherInformation has a null key or value, returns false.

2. **Control Flow**
    a. Valid identifier, publisher information
    b. Invalid identifier, publisher information

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after addVenue

addArea

1. **Input Validations and boundary case**
    a. researchArea is an empty string, return false.
    b. If parentArea contains a null or empty string value, return true(can assume it is at the top of the research area hierarchy)
    c. researchArea is duplicated in the library, returns false.

2. **Control Flow**
    a. Valid and unique Researcharea
    b. Invalid ResearchArea

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after addPublication, addVenue and addPublisher

getPublications

1. **Input Validations and boundary case**
    a. key is null or empty, returns an empty map
    b. key is an empty string, returns an empty map
    c. key does not match any publication in the library, returns an empty map.
    d. Key is valid, return an map with valid data.

2. **Control Flow**
    a. Get an empty set if no publications found
    b. Get a map if publication found

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after addPublication, addReferences

authorCitations

1. **Input Validations and boundary case**
    a. Author has no citations, return 0.
    b. author does not match any author in the library, return 0.
    c. author has a match in the library with valid citations, return the count.

2. **Control Flow**
    a. Get an empty set if no citations found
    b. Get a count if citation found

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after calling all the add methods

seminalPapers

1. **Input Validations and boundary case**
    a. area is null or empty, returns false
    b. paperCitation should be a non-negative integer.
    c. otherCitations should be a non-negative integer.
    d. area is an empty string, return an empty set.
    e. paperCitation is zero and otherCitations is zero, return an empty set.
    f. paperCitation or otherCitations are negative integers, return an empty set.

2. **Control Flow**
    a. Get set of areas if no correct number of references found
    b. Get an empty set if invalid references found

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after calling all the add methods

Collaborators

1. **Input Validations and boundary case**
    a. author is null or empty, returns empty set
    b. distance should be a non-negative integer.
    c. author is an empty string, return an empty set.
    d. distance is zero, returns a set containing only the author.
    e. distance is negative, returns an empty set.

2. **Control Flow**
    a. Get a set of collaborators if author is in the database
    b. Get a empty set if invalid author name

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after calling all the add methods

authorResearchAreas

1. **Input Validations and boundary case**
    a. author should is null or empty, returns empty set.
    b. threshold is negative integer, returns empty set.
    c. author is an empty string, returns an empty set.
    d. threshold is zero, returns an empty set.
    e. threshold is positive, returns set of research areas.

2. **Control Flow**
    a. Get the areas if the threshold and author names are valid
    b. Get an empty set if data is invalid

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after calling all the add methods

IEEECitationConverter

1. **Input Validations and boundary case**
    a. Input file or output file is null
    b. Input file or output file is blank
    c. Input file does not exist
    d. Input file exists

2. **Control Flow**
    a. References in \cite is present in database
    b. References in \cite is absent in database

3. **Data Flow**
    a. Calling this method at first
    b. Calling this method after calling all the add methods