Overview

We have to extract summary information from a database of a bicycle store company and storing the information in an XML file. The program will take input from the user regarding the starting and ending dates of the period and the name of the output file. Asked 3 queries have been written and the extracted information will be divided into three categories: customer information, product information, and store information. The program will use Java and the MySQL JDBC connection for Java to access the database. The program must use proper XML file format, including human readability. For that need to use the XML builtin libraries and document builders.

Files and external data

There will be total 3 files to execute the program.

1. RunApplication
2. PublicationLibrary
3. Authors
4. PaperCoversion
5. DataBaseConnection

RunApplication: It contains the main method. The purpose of this program is to get input from the user for the adding the publications, references, Venue, publishers.

PublicationLibrary: It contains the main logic of the problem statement, which gets the input and stores into database and retrieve as and when needed.

Authors: It contains the details of the authors like getName, getCitation, addCitation

DataBaseConnection: It connects the code with the MYSQL database. The class will implement the Singleton pattern, which ensures that only one instance of the class is created.

PaperConversion: This file will have method to convert the given link/ URL to proper IEEE style format.

Data structures and their relations to each other along with Key algorithms

---

To manage the information about publications, we will create a Java class called PublicationLibrary. Here, we will restrict the class to manage journal and conference publications, but the design should be ready to add other forms of publications in the future. We will use internal data structures to store information. We will also ensure that the information managed by the class should survive between the executions of programs that use the PublicationLibrary class.

The following key algorithms and data structures for the PublicationLibrary class can be implemented:

HashMap: It can be used to store the information about each publication. The key of the HashMap will be the publication identifier, and the value will be a Map that stores all the information about the publication.

ArrayList/List: ArrayList or List can be used to store the references for each publication. We can add a Set of strings that contains the publication identifiers cited by the paper.

Graphs: the references between publications can be represented as a directed graph. Each node of the graph will represent a publication, and an edge from node A to node B will indicate that publication A cites publication B.

Breadth-First Search (BFS): We can use BFS to find the shortest path between two publications in the citation graph. This can help answer the question of how many publications lie within X publication authorships from me, as potential grant collaborators.

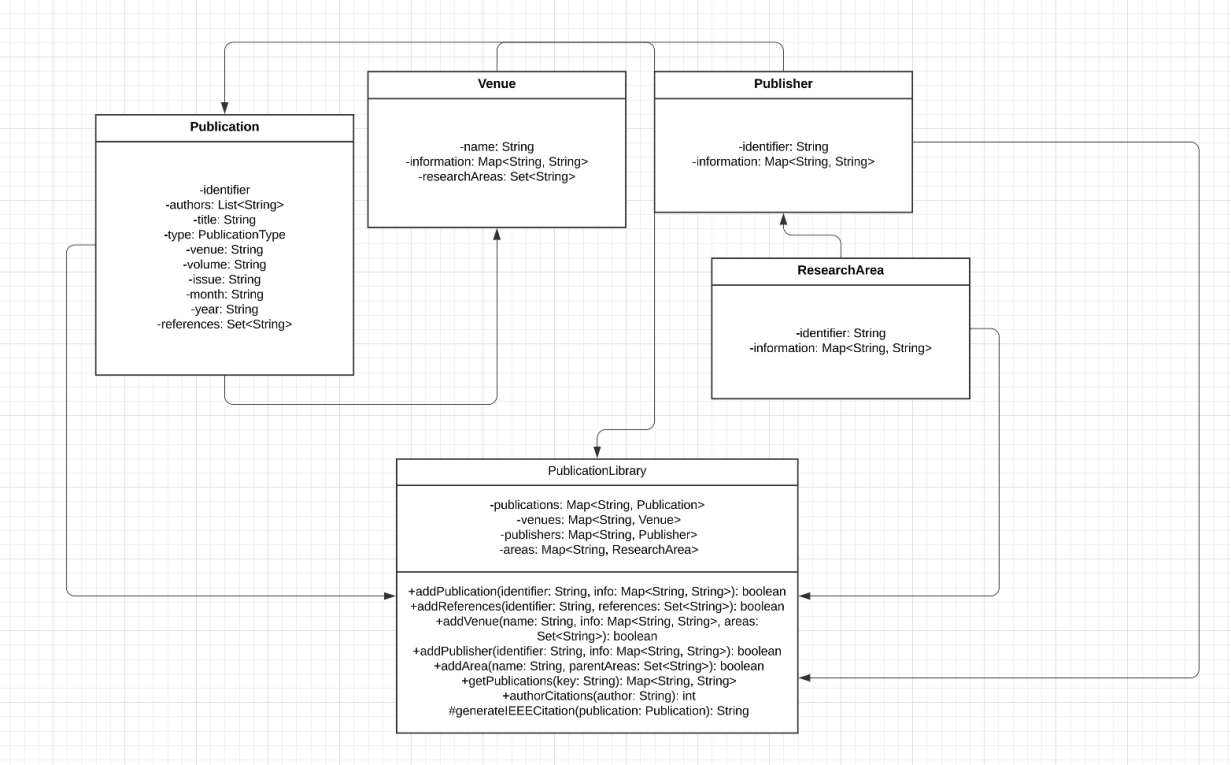
Depth-First Search (DFS): We can use DFS to traverse the citation graph and find the seminal papers in a given research area. We can define seminal papers as those that have been cited by a certain number of other papers in the same research area.

Set: We can use Set to store the research areas and the parent areas for each research area. Each research area can be a subset of zero or more other research areas.

Assumptions

- Author's full names will be unique, will not be hyphenated names, and are case insensitive.
- Data won't be removed from library.
- The input to generate IEEE format will be provided in the file format.

Design Elements



Rough database design and its tables for the "PublicationLibrary" class.
In the first step for designing the database. identified the following entities:

- Publication
- Author
- PublicationVenue (journal or conference)
- Organization
- Contact
- Publisher
- ResearchArea

Next, need to identify the relationships between these entities. Based on the problem statement, the following relationships exist:

- A publication can have one or more authors.
- A publication is published in a publication venue.
- A publication venue is organized by an organization.
- An organization has a contact person and a home office.
- A publication venue has a publisher.
- A publication belongs to one or more research areas.
- A publication can reference one or more other publications.

Based on these entities and relationships, the following database tables can be created:

- Table: Publication
- Table: Author
- Table: Publication_Author
- Table: Publication_Venue

- Table: Organization
- Table: Contact
- Table: Publisher
- Table: Research_Area
- Table: Publication_Reference

The Publication table stores information about the publication. the reference will be the Publication_Venue table. The Publication_Author table is a junction table that links the Publication and Author tables.

The Author table stores information about the authors. The Publication_Author table is a junction table that links the Publication and Author tables.

The Publication_Venue table stores information about the publication venue. the reference will be the Organization table, and another one references the Publisher table.

The Organization table stores information about the organization that organizes the publication venue. the reference is the Research_Area table and the Contact table.

The Contact table stores information about the contact person for the organization and the publisher.

The Publisher table stores information about the publisher.

The Research_Area table stores information about the research areas. The reference will be the Research_Area table.

The Publication_Reference table is a junction table that links the Publication table with other publications that it references.

These tables can be used to store and manage information about publications and their associated data. With this database design, asked methods can be implemented for the PublicationLibrary class to interact with the database and retrieve the required information.