Overview

Objective is to answer some queries and convert citation strings into IEEE format. IEEE formatting guide is used to understand which information should be there in a citation and which will be optional.

Journal publication have information such as the set of authors, paper title, journal name, page range, volume, issue number, month and year of the publication.

Conference publication have set of authors, paper title, conference name, page range, location where the conference is held and the year during which the conference is organized.

An organizer organizes - the journal publication and conference publication. Both of these have a research area that the journal and conference publication covers. They also have an organizer who organizes it as well as the contact number of the organizer. These organizations also have their own contact information and their home office.

There are two researchAreas – narrow research area and broad researchArea. Broad research area is includes narrowresearcharea.

Each publication will be associated with an alphanumeric string. Authors will use this alphanumeric string to cite the publication in their paper. Our objective is to convert these citation strings into actual IEEE references for the paper being cited.

Following are the queries that this project will answer

1. How many citations does an author have?

2. Which are the seminal papers in a given research area?

3. What are the references in a given article?

4. Who lies within X publication authorships from me, as potential grant collaborators?
A. This answer can be solved by calling the method collaborators (my name, distance)

5. In what areas does an author do research?

6. Given a paper, convert the citation strings into actual IEEE citations.

Data Structures, Code Design and Key Algorithms

I am intending to create following classes, variables and methods to achieve the result.

Classes	Variables	Methods

publicationManager	Set<String> authors = new HashSet<>();
	String paper_title;
	String journal_name;
		managePublicationInfo()
		manageJournal()
		manageConferences()

	String page_range;
	String volume;
	String issue_number;
	String month_publication;
	String year_publication
	String conference_name;
	String locationConference;
	Set<String> references;
	Citation citation;	

publicationVenue	String Organization name String Contact information String Home office address ResearchArea area	organizationInfo()
	String Editor/Organizer name String Contact Information	organizerInfo()
ResearchAreas	String researchAreaName Set<String> researchAreaParent	

Citations	String citation	inputFormat()
		outputFormat()

Starter	File inputfile = new File();
	File outputFile = new File();
		transformCitations()
		// this would take the alphanumeric citation string from inputFile, and transform that as per the IEEE format of the references

publicationLibrary	Private Map<String, Map<String,String>> allPublications;
	
	Private Map<String, Map<String,String>> allReferences;
	
	publicationVenue venue = new publicationVenue()
		boolean addPublication (String identifier, Map<String, String> publicationInformation)
		
		boolean addReferences (String identifier, Set<String > references)
		
		boolean addVenue (String venueName, Map<String, String> venueInformation, Set<String>
		researchAreas)

	ResearchAreas research = new ResearchAreas() Private Map<String,Map<String, String>> allPublishers;	boolean addPublisher (String identifier, Map<String, String> publisherInformation)

Method description

boolean addPublication (String identifier, Map<String, String> publicationInformation)

all Publications		
Identifier – Identifies a publication uniquely	publicationInformation	
	Key	value
“1”	author Title Journal Page Volume Issue Month Year Conference Locations	AuthorName1, Authorname2 titleName journlName pagenumber volumenumber issueNumber monthNumber yearNumber conferencename pageNo.
“2”	Same as above	Same as above
“3”	Same as above	Same as above

Above structure is given to us in the form of method parameters. Now all we have to do is map identifier as the key for which value will be the publication information and store it in the allPublications map.

e.g. allPublications.put(Identifier, publicationInformation); - If this is successful, then set return value true, or set return value as false.

boolean addReferences (String identifier, Set<String > references)

allReferences	
Identifier – Identifies a publication uniquely	Set of references

“1”	Set: {reference1, reference 2, reference 3} – this should be unique. No reference should get repeated.
“2”	Set: {reference1, reference2, reference 3} – No reference is repeated.
“3”	Set: {reference1, reference2, reference 3} – No reference is repeated

Bind the identifier and the set of references as map’s allReferences key and value pairs and store it into the allReferences map. If the storage is successful, then return true or else return false.

It will help to identify which reference is in which publication with the help of publicationIdentifier.

boolean addVenue (String venueName, Map<String, String> venueInformation, Set<String> researchAreas)

	venueInformation		
venueName	typeofInformation	actualInformation	researchAreas
Journal/Conference	Publisher	Publisher 1	{“computational geometry”, “isc”}
	Editor	Editor 1	
	Editor_contact	Contact number of the editor	
	Location	Location of the venue	
	Conference_year	Which year conference was held	

Bind the venueName with venueInformation and store these mapping in a separate class called as publicationVenue.

boolean addPublisher (String identifier, Map<String, String> publisherInformation)

allPublishers		
	publisherInformation	
Identifier	Type of information (key)	Actual Information (value)
“1”	Contact_name	Publisher 1
	Contact_email	publisher1@gmail.com
	Location	Location of the publisher

“2”	Contact_name	Publisher 2
	Contact_email	publisher2@gmail.com
	Location	Location of the publisher 2

boolean addArea (String researchArea, Set<String> parentArea)

add the research area to the set of parentAreas.

Map<String, String> getPublications (String key)

This first returns a map of all the information that’s stored on the specific publication.

Key that is passed as a parameter value is the publication key of a specific publication.

That publication must be referencing many articles.

Parameter value key	Key	Value
“1”	Reference	{“publicationidentifie1”,” publicationidentifier2”, publicationidentifier3”}

int authorCitations (String author)

Return the no. of publications that have directly cited the author.

Set<String> seminalPapers (String area, int paperCitation, int otherCitations)

Here, paperCitation refers to the number of citations referenced in the paper.

otherCitations refers to the number of papers that have referred the citaiton.

It has few references of its own in the current research area, but has many other papers in the research area referencing it.

seminalPapers = Returns the set of publication identifiers.

otherCitations <= count(seminalPaperOtherCitation)

count(seminalPapersinPapercitation) <= paperCitation

< paperCitation > Other Citation

Set<String> collaborators(String author, int distance)

Distance = shortest length that links one author to another author through co-authorship

Let's assume that each author is a node in the graph and the link between the authors is established through the co-authorship. Then, we use a djikstra to find the shortest path.

A – A = 0

A – B = 1

A – C = 1

B – D = 1

A – D = A – B – D =2

Paper1 - {A,B,C}

Paper 2 - {B,D}

Paper 3 – {B,E}

Paper 4 – {C,F}

Paper 5 – { F, E}

Store it in a 2D Matrix and then apply djikstra to find the shortest path and fill up the empty spaces in the matrix.

	A	B	C	D	E	F
A	0	1	1			
B	1	0	1	1	1	
C	1	1	0			1
D						
E						
F					1	

Set authorResearchAreas (String author, int threshold)

Return the researchareas in which the given author has published >= "threshold" of papers (passed from the parameter value)

Paper conversion

It's the starter class described above in the first table.

It takes in two files – inputFile and outputFile.

inputFile consists of regular text with entries in the form of \cite{xxx}

outputFile will consist of the converted entries.

inputFormat() – It reads in the word that starts with \cite{xxx} and all such entries.

outputFormat() – It converts the \cite{xxx} to [1] and stores that in the outputFile and then will append the text of the wholePaper with IEEE formatted reference entry.

Store the result ([1] appended with IEEE formatted reference entry, [2] appended with IEEE formatted reference entry) into the outputFile.

Test Cases

boolean addPublication (String identifier, Map<String, String> publicationInformation)

Input validation

Identifier is null, return false.

Identifier is empty, return false.

Map publication information is empty for the given identifier, return false.

Map publication information is null, return false.

publicationInformation is empty, return false.

Identifier should be unique, return true.

Boundary case

Map publication information contains all expected keyvaluepairs in valid format, return true.

Map publication information contains only some expected keyvaluepairs in valid format, return false.

Control flow

If the identifier value already exists in the library, return false.

If the year attribute in the publication information is future-related or invalid, then return false.

Multiple authors in the publication information are allowed to be added, then return true.

If the month attribute in the publication information is future-related or invalid, then return false.

If a mandatory information is missing in the publication information and trying to add it to the library, return false.

If more than one mandatory information is missing in the publication information, return false.

When any of the fields in the publication information is there, but the value is empty, return false.

Dataflow

Before call

- Store all the publication related information in publicationInformation.
- Store a unique identifier for each publication.

During call

- Ensure that identifier is binded to the information in publicationInformation and returns true if success and false if not success.

boolean addReferences (String identifier, Set<String > references)

Input validation

Identifier is null, return false.

Identifier is empty, return false.

There are no references for the given identifier, return false.

Identifier and references both are valid, return true.

Identifier and references both are empty, return false.

Reference is a alphanumeric set of strings, return true.

Control flow

If for a set of references, identifier already exists, return false.

If reference is repeated for a specific identifier, return false.

Data Flow

Before call

- Store all the reference related information in the set of references.
- Store a unique identifier for each publication.

During call

- Ensure that unique identifier is binded to the set of references and returns true if added and false if not added.

boolean addVenue (String venueName, Map<String, String> venueInformation, Set<String> researchAreas)

Input Validation

venueName is not null, return true.

venueName is not empty, return true.

venueInformation is not empty, return true.

researchAreas set is not empty, return true.

researchAreas set is not null, return true.

researchAreas set has unique values and no duplicate elements, return true.

BoundaryCase

Year in the Conference_year variable in the venueInformation is in valid format, return true.

Year in the Conference_year variable in the venueInformation is not a future year, return true.

Editor_contact in the conference_year variable in the venueInformation is in valid format, return true.

Data Flow

Before call

- Store all the venue related information in the venueInformation map.
- Store all the researchareas related information in the set of researchAreas.

During call

- Ensure that venue is added in the proper format, return true.

Control Flow

All the mandatory information in the venueInformation is there, return true.

venueName already exists in the library, return false.

Some of the mandatory information in the venueInformation is not there, return false.

All mandatory information in the venueInformation is missing, return true.

boolean addPublisher (String identifier, Map<String, String> publisherInformation)

Input Validation

Identifier is empty, return false.

Identifier is null, return false.

publisherInformation is not empty, return true.

publisherInformation is not null, return true.

Boundary Case

Identifier already exists, return false.

Control Flow

publisherInformation already exists for the given identifier, return false.

Contact_name in the publisherInformation is valid and is in proper format, return true.

Contact_email in the publisherInformation is valid and is in proper format, return true.

boolean addArea (String researchArea, Set<String> parentArea)

Input validation

researchArea is not empty

researchArea is not null.

Boundary case

Research area is added with no parent areas, return true.

Research area is added with one parent area, return true.

Research area is added with more than one parent area, return true.

Research area is unique, return true.

Map<String, String> getPublications (String key)

Input validation

Key is null, return false.

Key is empty, return false.

Key is not present in the publicationLibrary, return false.

Key is in the library, return true.

int authorCitations (String author)

Input validation

author is null, return false.

Author is empty, return false.

Author given doesn’t have any citations associated, return true.

Author’s name is misspelled or incorrect, return false.

Author doesn’t exist in the publicationLibrary, return false.

Set<String> seminalPapers (String area, int paperCitation, int otherCitations)

Input validation

Area is not null, return true.

Area is not empty, return true.

Area is valid, but no corresponding paper exists, return false.

paperCitation value is 0 or negative, return false.

otherCitation value is 0 or negative, return false.

paperCitation value is not null, return true.

paperCitation value is not empty, return true.

otherCitations value is not null, return true

otherCitations value is not empty, return true

Set<String> collaborators(String author, int distance)

- Author is not null, return true.
- Author is not empty, return true.
- Distance is not negative, return true.
- If distance = 0, then author is pointing to itself.
- If distance = 1, then author is pointing to some other author which has contributed to the same paper.

- Author has contributed alone to the paper – it's alone and disjoint, return true (ie. Empty)
- If distance = 2, then author is pointing to some other author which has contributed to the same paper and also to an author of another paper, return true.

Set<String> authorResearchAreas (String author, int threshold)

- Author is null, return true.
- Author is empty, return true.
- Threshold is empty and has no value, return true.
- Author is not there in the library, return false.
- Author has no research which meets the threshold, return empty set.
- Author has research areas that meet the threshold, return true.
- Author has more than one research areas, that meets the threshold, return all research areas

Paper conversion

- Citation string in Input file is in proper format, return true.
- Input file is not empty, return true.
- Citation string in input file is valid and exists in the library, return true.
- More than one Citation strings in input file is in proper format, return true.
- Output is in IEEE reference proper format, return true.
- Output for more than one citation strings is in proper format, return true.
- Output for one citation string is in proper format, return true.
- Every citation string has associated output, return true.

Documentation dated 20-04-2023 – Final One

Final files

1. Main.java – calls the method and supplies the information
2. PublicationLibrary.java – executes the method as called upon

Final call order in main.java

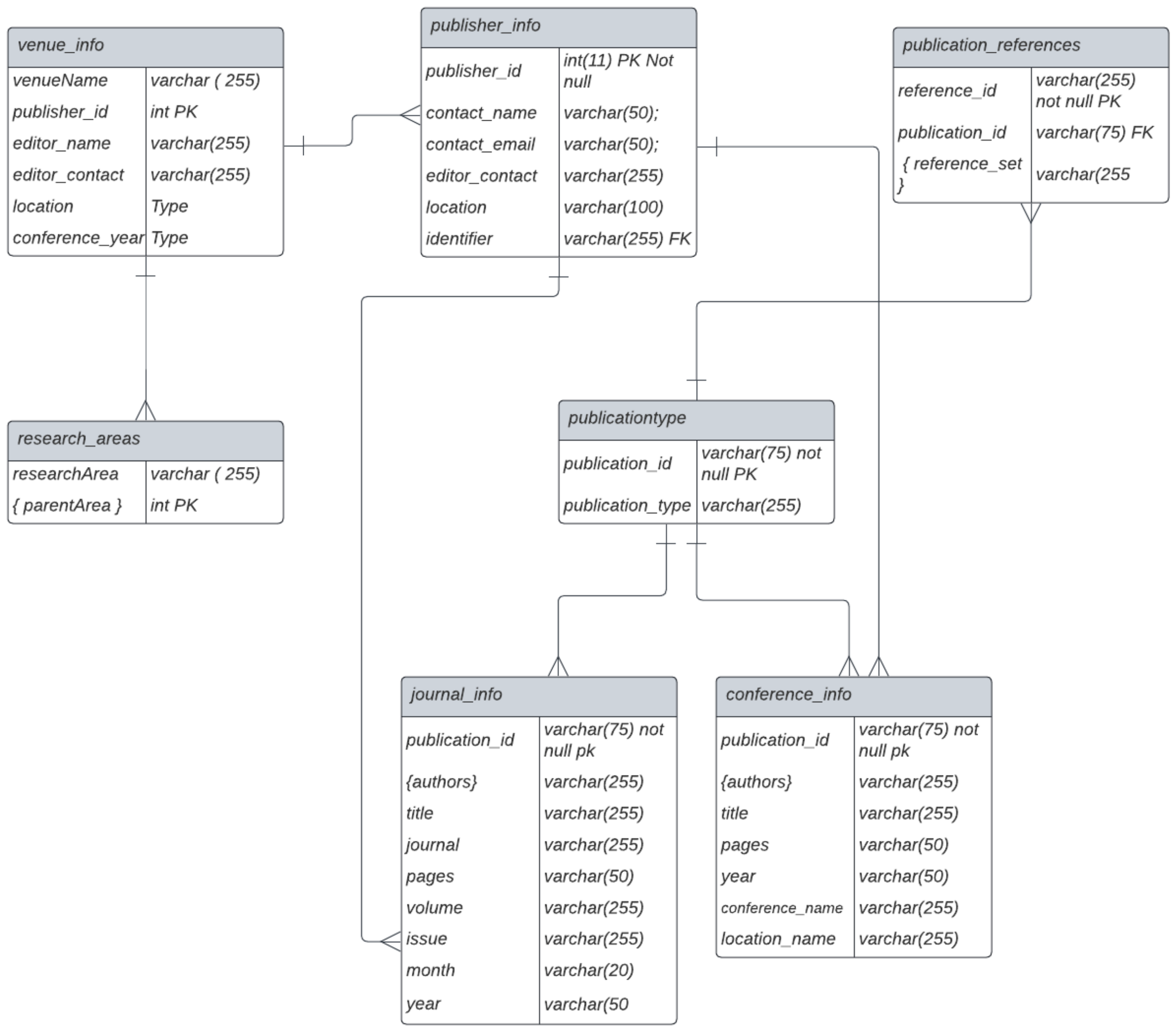
1. addArea()
2. addPublisher()
3. addVenue()
4. addPublication()
5. addReferences()

Exception handling is done using try catch statements.
Try – block of code

Catch Exceptions are as follows

1. SQL Exception
2. ClassNotFoundException

Database design


venue_info
venueName varchar (255)
publisher_id int PK
editor_name varchar(255)
editor_contact varchar(255)
location Type
conference_year Type
publisher_info
publisher_id int(11) PK Not null
contact_name varchar(50);
contact_email varchar(50);
editor_contact varchar(255)
location varchar(100)
identifier varchar(255) FK
publication_references
reference_id varchar(255) not null PK
publication_id varchar(75) FK
{ reference_set } varchar(255
research_areas
researchArea varchar (255)
{ parentArea } int PK
publicationtype
publication_id varchar(75) not null PK
publication_type varchar(255)
journal_info
publication_id varchar(75) not null pk
{authors} varchar(255)
title varchar(255)
journal varchar(255)
pages varchar(50)
volume varchar(255)
issue varchar(255)
month varchar(20)
year varchar(50
conference_info
publication_id varchar(75) not null pk
{authors} varchar(255)
title varchar(255)
pages varchar(50)
year varchar(50)
conference_name varchar(255)
location_name varchar(255)

1. addArea(): stores the parameter values into research_areas table.
2. addPublication(): stores the parameter values into journal_info, publication_type, conference_info tables. If the parameter passed has the key journal in it, then it will be stored in the journal_info table. If the parameter passed has the key conference in it, then it will be stored in the conference_info table.
3. addReferences(): stores the parameter values into publication_references table.
4. addVenue(): stores the parameter values into venue_info table. It calls venue_info basis if the venuetype is journal or the venuetype is conference.
5. addPublisher(): stores the parameter values into publisher_info table.
6. getPublications(String key)
   retrieves the corresponding values from journal_info, conference_info and

publication_references