README

Steps for creating and running the project:

1. Execute FINAL SQL Schema dump accumguojohns.sql to create the schema of 'accumguojohns' which includes all the data and tables used within the project.
2. Execute create_guest.sql to be able to log into the database as a guest, using the username: guest and the password: guest_user, otherwise use root user and root user password to login as DBA.
3. Run db.jar file within command prompt to access front end of the project.
    a. Enter username and password when prompted.
        i. Guest Account:
            1. Username: guest
            2. Password: guest_account
        ii. DBA
            1. Username: root
            2. The same password as used to login as a root user
4. As DBA enter one of the following:
    a. Q -- to enter a SQL statement
    b. P -- to view pending approvals
        i. E -- to show guest_user requests to edit a paper
        ii. A -- to show guest_user requests to add an article
        iii. K -- to show guest_user requests to add a keyword
        iv. P -- to show guest_user requests to add a paper
        v. D -- to show guest_user requests to delete a paper
    c. A -- to perform admin functions
        i. A -- add an article
            1. Follow steps on screen to add an article to the database.
        ii. P -- add a paper
            1. Follow steps on screen to add a paper to the database.
        iii. D -- delete a paper
            1. Follow steps on screen to delete a paper from the database.
                a. Triggers will be activated here to delete corresponding journals, articles, and authors.
        iv. K -- add a keyword
            1. Follow the steps on the screen to add a keyword with a corresponding paper_id to the database.
        v. C -- add a citation
            1. Follow the steps on the screen to add a citation with a corresponding paper_id to the database.
    d. Z -- to quit
5. As a guest_user:
    a. Q -- to enter a query
        i. P -- to query for a paper
            1. T -- to query a paper by the paper's title

            a. Follow steps on screen to continue
        2. A -- to query a paper by the paper's author
            a. Follow steps on screen to continue
        3. K -- to query a paper by a keyword
            a. Follow steps on screen to continue
        4. J -- to query a paper by a journal
            a. Follow steps on screen to continue
    ii. Au -- to query for an author
        1. Enter author's name to get information about them
    iii. Art -- to query for an article
        1. T -- query an article by it's title
            a. Follow steps on screen to continue
        2. W -- query an article by it's writer
            a. Follow steps on screen to continue
        3. P -- query an article by the paper(s) it cites
            a. Follow steps on screen to continue
b. R -- to request an update
    i. D -- request to delete a paper
        1. Enter the paper_id of which paper is to be looked at for deletion
        2. Enter the reason for deletion of said paper
    ii. I -- request to insert a paper or an article
        1. A -- request to add an article
            a. Follow steps on screen to complete suggestion
        2. P -- request to add a paper
            a. Follow steps on screen to complete suggestion
    iii. K -- request adding a keyword to a paper
        1. Enter the paper_id of which paper you want to add a keyword to
        2. Enter keyword to be added
    iv. E -- request to edit a paper
        1. Enter the paper_id of the paper to be edited
        2. Describe the changes to be made
c. Z -- to quit

Libraries and Software:

- MySQL -- to run .sql scripts and house the data and schema
- Java -- to run the jar file within the command line
- IntelliJ - IDE used to create the project
    - The program can also be run by opening up the "db" folder in IntelliJ which contains an IntelliJ project.

Technology Download Pages:

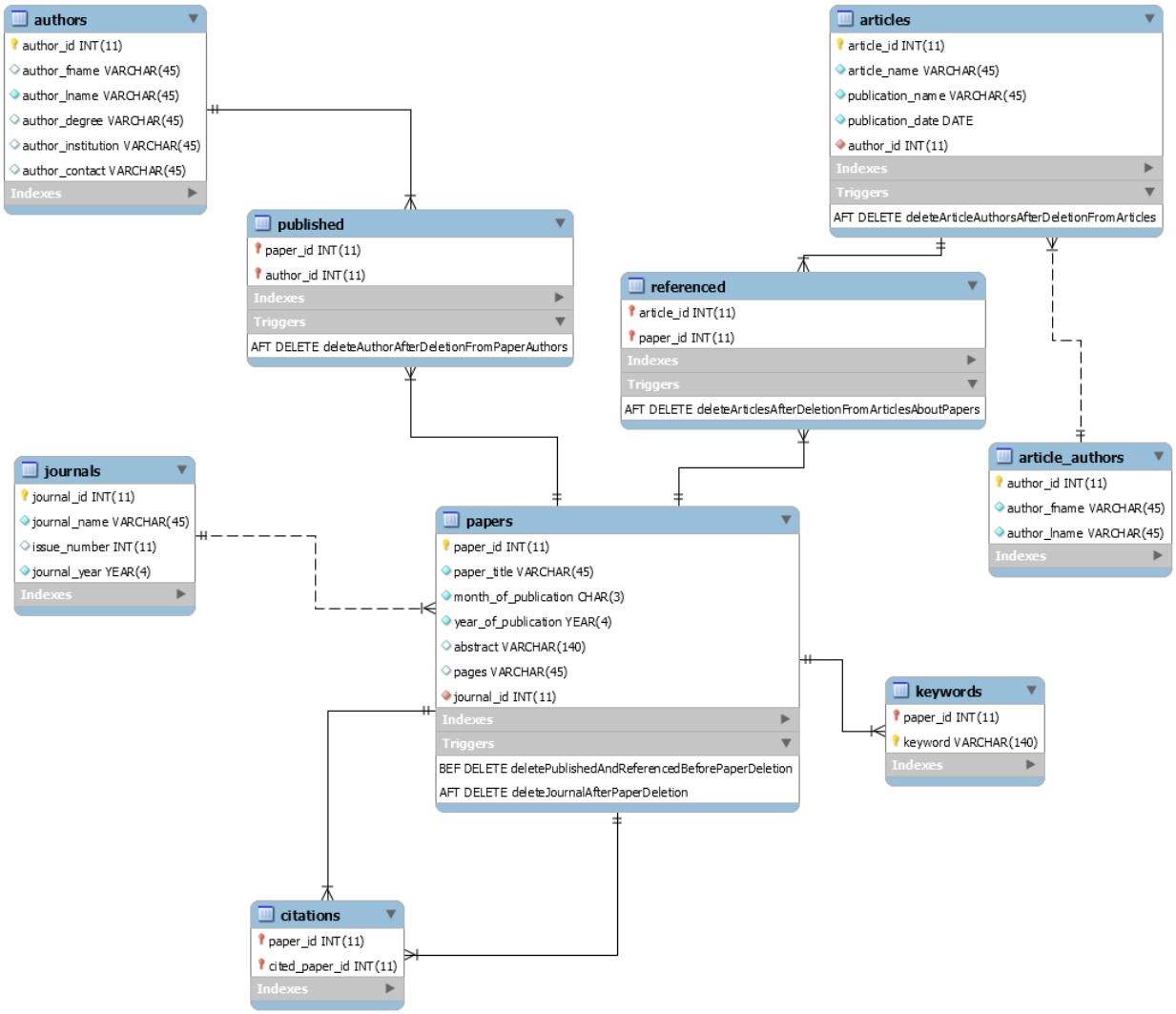
- https://dev.mysql.com/downloads/connector/j/

- Used to connect Java to MySQL

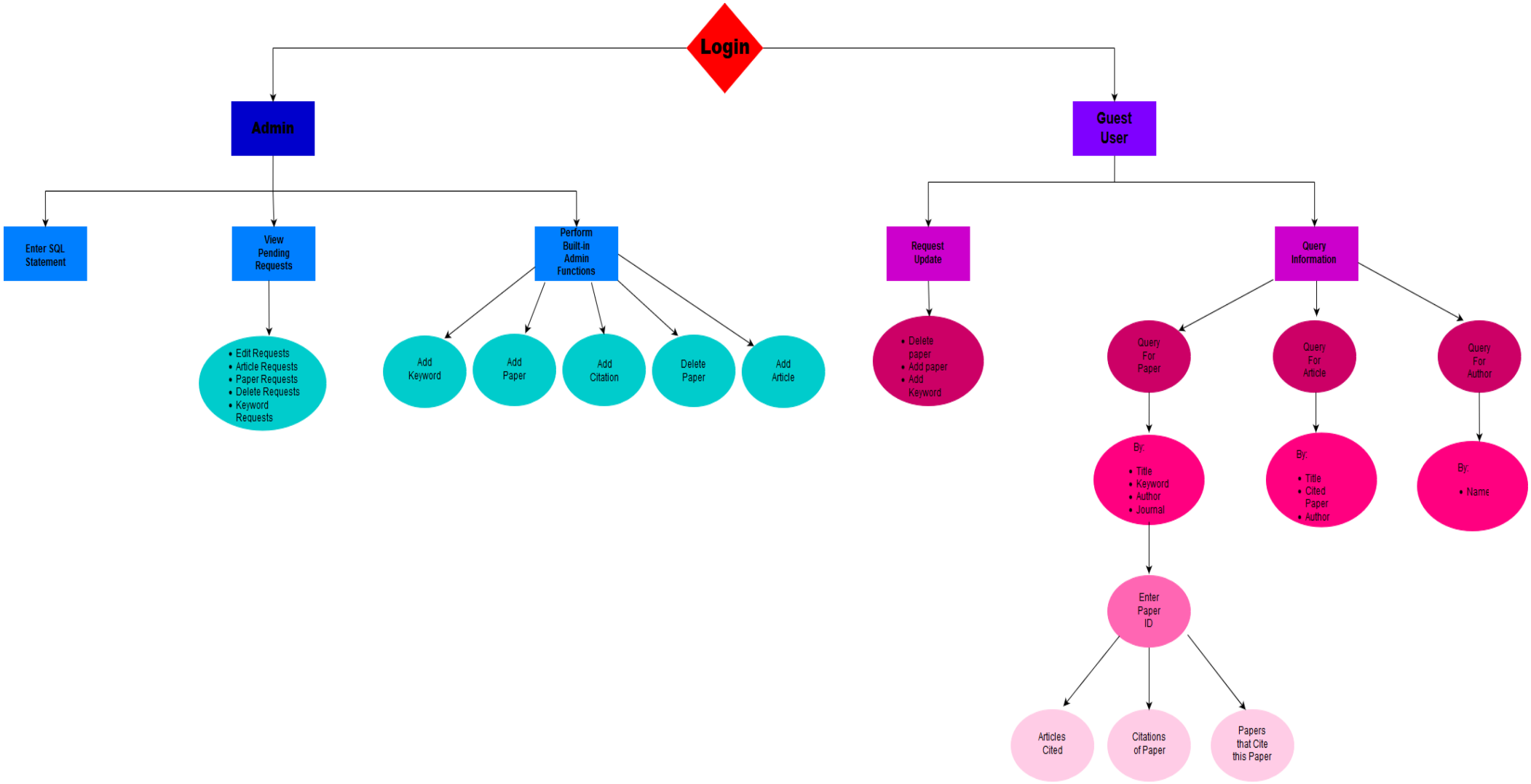
TECHNICAL SPECIFICATIONS

The database is managed by and uses MySQL to perform queries and Java is used to interact with the user and act as the mediator between the user and database.

EER DIAGRAM


authors
author_id INT(11)
author_fname VARCHAR(45)
author_lname VARCHAR(45)
author_degree VARCHAR(45)
author_institution VARCHAR(45)
author_contact VARCHAR(45)
Indexes
articles
article_id INT(11)
article_name VARCHAR(45)
publication_name VARCHAR(45)
publication_date DATE
author_id INT(11)
Indexes
Triggers
AFT DELETE deleteArticleAuthorsAfterDeletionFromArticles
published
paper_id INT(11)
author_id INT(11)
Indexes
Triggers
AFT DELETE deleteAuthorAfterDeletionFromPaperAuthors
referenced
article_id INT(11)
paper_id INT(11)
Indexes
Triggers
AFT DELETE deleteArticlesAfterDeletionFromArticlesAboutPapers
journals
journal_id INT(11)
journal_name VARCHAR(45)
issue_number INT(11)
journal_year YEAR(4)
Indexes
papers
paper_id INT(11)
paper_title VARCHAR(45)
month_of_publication CHAR(3)
year_of_publication YEAR(4)
abstract VARCHAR(140)
pages VARCHAR(45)
journal_id INT(11)
Indexes
Triggers
BEF DELETE deletePublishedAndReferencedBeforePaperDeletion
AFT DELETE deleteJournalAfterPaperDeletion
article_authors
author_id INT(11)
author_fname VARCHAR(45)
author_lname VARCHAR(45)
Indexes
keywords
paper_id INT(11)
keyword VARCHAR(140)
Indexes
citations
paper_id INT(11)
cited_paper_id INT(11)
Indexes
keyword_requests
keyword_requests_id INT(11)
paper_id INT(11)
keyword VARCHAR(140)
Indexes
paper_requests
paper_requests_id INT(11)
paper_title VARCHAR(45)
paper_abstract VARCHAR(140)
paper_date VARCHAR(45)
author_name VARCHAR(45)
journal_name VARCHAR(45)
journal_year VARCHAR(45)
journal_issue_number VARCHAR(45)
Indexes
delete_requests
delete_requests_id INT(11)
paper_id INT(11)
reason VARCHAR(140)
Indexes
article_requests
article_request_id INT(11)
title VARCHAR(45)
author_name VARCHAR(45)
publication_date VARCHAR(45)
publication_name VARCHAR(45)
Indexes
edit_requests
edit_id INT(11)
edit_paper_id INT(11)
edits VARCHAR(5000)
Indexes

FINAL USER FLOW DIAGRAM



PROCEDURE/TRIGGERS/ERROR HANDLING

- Procedures
    - Admin Procedures
        - showEditRequests
            - Shows all requests by guest user to update/edit a paper
        - showArticleRequests
            - Shows all requests by guest user to add an article
        - showKeywordRequests
            - Shows all requests by guest user to add a keyword to a paper
        - showPaperRequests
            - Shows all requests by guest user to add a paper
        - showDeleteRequests
            - Shows all requests by guest user to delete a paper
        - add_article_with_known_author
            - Add an article to the database with an article author that's already in the database
        - add_article_with_new_author
            - Add an article to the database with a new article author that isn't in the database
        - add_paper_with_known_author_and_known_journal
            - Add a paper to the database with an author and a journal that are already in the database
        - add_paper_with_new_author_and_known_journal
            - Add a paper to the database with an author that's not yet in the database, but has a journal that is in the database
        - add_paper_with_new_author_and_new_journal
            - Add a paper to the database with a new author and a new journal that are not yet in the database
        - add_paper_with_known_author_and_new_journal
            - Add a paper to the database with an author that is already in the database, but a journal that is not
        - delete_paper
            - Delete a paper from the database
        - add_keyword
            - Add a keyword to a paper in the database
        - add_citation
            - Add a citation to a paper in the database

- Guest Procedures
  - getPaperByAuthor
    - Search for paper(s) by a certain author
  - getPaperByTitle
    - Search for paper(s) with a certain title
  - getPaperByKeyword
    - Search for a paper by keyword
  - getPaperByJournal
    - Search for paper(s) that appear in a certain journal
  - getPapersCitedByThisOne
    - Search for paper(s) that are cited by a given paper
  - getPapersThatCiteThisOne
    - Search for paper(s) that cite a given paper
  - getArticleByTitle
    - Search for article(s) by the article title
  - getArticleByAuthor
    - Search for article(s) by their author names
  - getArticleByPaperCitation
    - Search article(s) that are cite a certain paper
  - getAuthorByName
    - Search authors of papers by the author name
  - getPapersCitedByArticle
    - Return paper(s) cited by an article
- Triggers
  - deletePublishedAndReferencedBeforePaperDeletion
    - Before a paper is deleted, delete the relevant tuples from the referenced and published tables.
  - deleteJournalAfterPaperDeletion
    - After a paper is deleted from the papers table, delete its corresponding journal only if the journal has no other papers that reference it.
  - deleteAuthorAfterDeletionFromPaperAuthors
    - After a paper-author tuple is deleted from the published table, delete the corresponding author from the authors table only if that author has not published any other papers in the database.
  - deleteArticlesAfterDeletionFromArticlesAboutPapers
    - After an article-paper tuple has been deleted from referenced, delete the corresponding article only if it does not cite any other papers in the database.

    - deleteArticleAuthorsAfterDeletionFromArticles
        - After an article had been deleted, delete its author from the article_authors table only if that author has not written any other articles in the database.
- Error handling
    - All transactions within the code catch an sql error and have a continue handler for any sqlexceptions which determine whether the transaction is either committed or rolled back.
    - add_article_with_new_author
        - Uses a transaction to make sure the author and article were correctly added and the referenced table was updated to show that change.
    - add_article_with_known_author
        - Uses a transaction to make sure the article was correctly added to the database and the referenced table was updated to reflect that change.
    - add_paper_with_known_author_and_known_journal
        - Uses a transaction to make sure the paper was correctly added to the database and the published table was updated to reflect that change.
    - add_paper_with_new_author_and_known_journal
        - Uses a transaction to make sure the author and paper were successfully added to the database while updating the published table to show that change.
    - add_paper_with_new_author_and_new_journal
        - Uses a transaction to make sure the author, paper, and journal were successfully added to the database and updates the published table to reflect all changes.
    - add_paper_woth_known_author_and_new_journal
        - Uses a transaction to make sure the paper and journal were successfully added to the database while updating the published table to reflect the changes.

LESSONS LEARNED

While designing our database, we learned how much work goes into ensuring the data integrity of even a relatively simple database schema. Beyond the basic design of the schema, we learned how to ensure that the deletion of one tuple would properly cascade through the database, by using a combination of foreign key constraints and triggers, which are called both before and after a deletion from the papers table.

Our group was able to effectively split up the workload for the project, by assigning certain triggers and procedures to different members of the group. While we had previous experience with web design, we underestimated the time it would take to learn how to hook up a website to the code on the back end, so we ultimately ended up using a Java program from the command line.

When reflecting on the design of our project, we realize we should have focused more on the ideal design for our front end, because the use cases of the end user were very important in determining the functionality that we needed from our procedures and triggers on the back end. For example, by using a command line program instead of a website, we needed to provide the user with the ID numbers of papers in order for the user to be able to effectively perform queries on those papers. If we had been able to hook up a website to our Java code, we would have designed this use case differently, because the user could have clicked on the tuple of a paper instead of entering a paper ID number.

FUTURE WORK

Building a Website

For future work on the database, we would like to build a website on the front end that we could publish online. The website would have a smoother UI, so that users could more easily request changes to the database and perform queries through an interactive menu. By using a website, we would also be able to hide some of the details of the database from the end user, such as the paper ID numbers, which are currently needed by the user to perform queries on specific papers. We weren't able to hide this information from the user in a command line program, because many of the research papers have duplicate titles, which means the user needs an ID number in order quickly identify a unique paper. In a website, the user could easily click on an entry to see more information about that specific paper, so there would be no need to provide the user with ID numbers form the back end.

Improved Error Catching and Protection from Injections

Along with building a website, we would also like to further improve the error catching in our program. Currently, we do not filter or scan the user input for errors, aside from the errors that are caught by MySQL. In order for a website to be publishable and readily useable by the end user, we would like to implement protections from injections, which would involve scanning the user input to ensure there is no extraneous or malicious SQL code. We would also like to develop

a more robust system on the front end for ensuring that the user can easily enter a valid input without needing to know what the data looks like in the database.