Research Catalogue
CS315- Database Management System

Group-26:

- Vaibhav Nagar (14785)
- Pranay Borkar (14189)

April 15, 2017

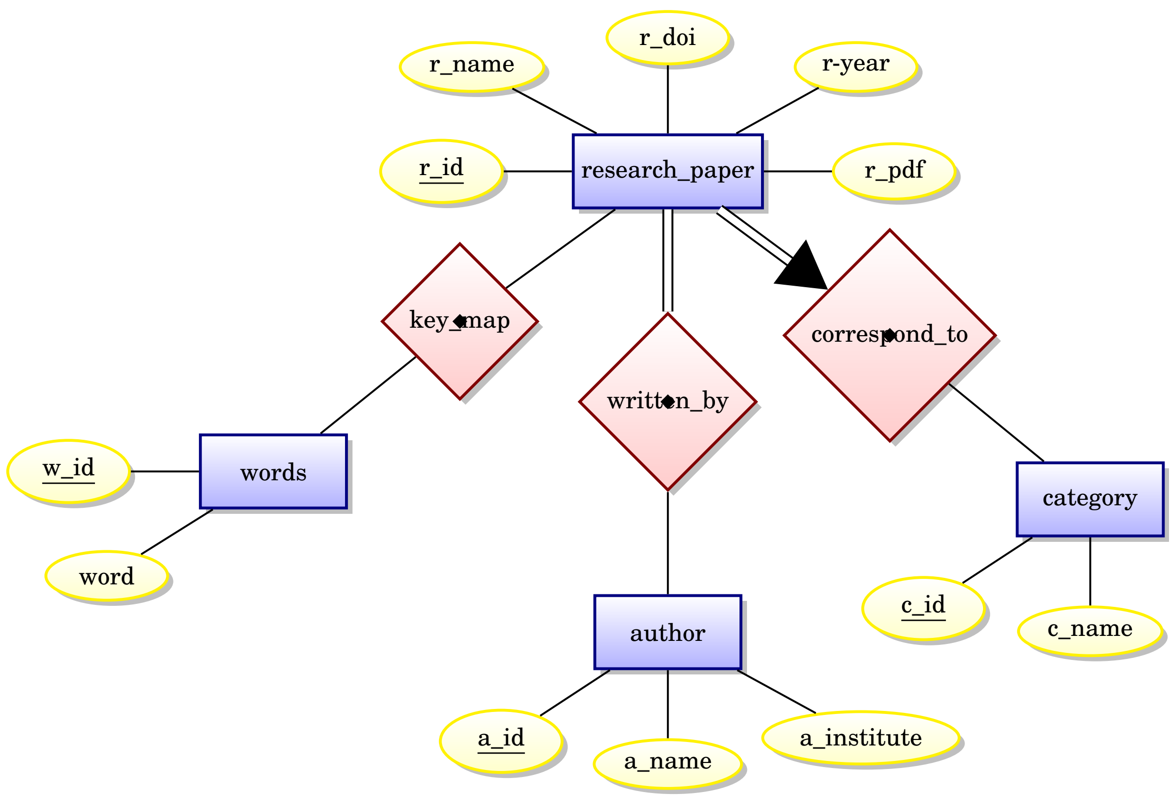
Abstract

Catalogue is a database of bibliographic records, an organized digital collection of references to published literature, including journal and newspaper articles, conference proceedings, reports, government and legal publications, patents, books, etc. Students who are new to many research fields but interested in exploring different fields face difficulties in getting reading material to start their research study. Thus, a user friendly interface that enables accurate search results and easy browsing of search results by providing helpful links and suggested search results is desired.

Technologies used

- **Database Management System:** MySQL database stored as InnoDB storage engine.
- **Front-End:** HTML5, CSS (Bootstrap), Javascript
- **Back-End:** PHP 7.0.15
- **Server:** Apache2

Entity-Relationship Diagram



Tables (Database) Flow chart with constraints

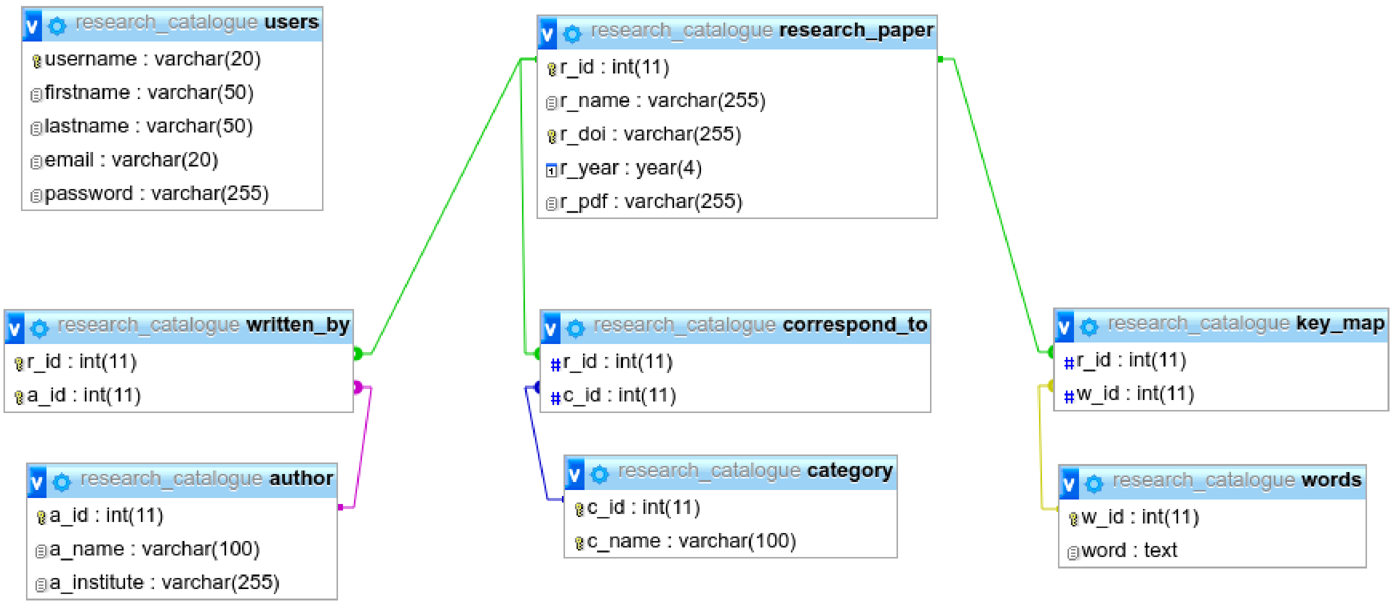


Figure 1: Flow chart of tables and arrows connected between tables indicate the foreign key constraint.

Implementation

User Interface:

The main steps to find any research paper in database is as follows:

Login

It allows only the registered students to access the research papers. It requires the username and password of the user. On login a session for that particular user is created which is destroyed on logout. One can register by clicking on Sign-up and providing basic information. The password provided during sign-up is stored in database after it is hashed.

Normal Search

In this Search the user provides a keyword that is to be searched for. The given keyword is then searched in all the fields of the database. The research paper in which the keyword appears is given as result.

Advanced Search

In this search the user provides the specific information about the research paper that he needs. The information that user can provide include the name of paper, author of paper, category to which the paper corresponds to and the year in which it was published.

Result

All the research paper that are qualified by the keywords and the filters given by the user are listed. Clicking on the name of research paper, redirects to the link containing information about the research paper(http://dx.doi.org/). For every research paper the list of authors, the category of paper and the year in which it was published is shown. If the institute to which the author belongs is known, it is stated in brackets in front of name of the author. The result also has options for viewing the summary and viewing the pdf the research paper.

Populating the Database:

Database is populated by using "arXiv " API hosted at arXiv.org. API calls are made via an HTTP GET method by passing a category name in the request and retrieve research papers in bulk in XML format. This data is then converted to python dictionary and inserted into database using appropriate SQL queries. Python script is written for this purpose which can be run online (using web interface) and offline (manually) to add research papers' data in the database.

Indexes Created:

To optimize the queries several indexes which are supported in InnoDB storage engine are created. Research papers are inserted only once in the database in the bulk, but more selection queries will be executed by the user. So indexes are made on such columns that optimizes the selection queries.

- Table::users: BTREE index on 'username'
- Table::research_paper: BTREE index on 'r_id', 'r_name', 'r_doi'
- Table::author: BTREE index on 'a_id', 'a_name'
- Table::category: BTREE index on 'c_id', 'c_name'
- Table::words: BTREE index on 'w_id'

SQL Queries and their Plan Tree

SQL syntax and its plan tree of the most frequently used query.

- SELECT r_id, r_name, r_year, r_doi, r_pdf, c_name
  FROM research_paper NATURAL JOIN written_by NATURAL JOIN author NATURAL JOIN correspond_to NATURAL JOIN category NATURAL JOIN key_map NATURAL JOIN words
  WHERE r_name LIKE '%xyz%' OR a_name LIKE '%xyz%' OR c_name LIKE '%xyz%' OR r_year = 'xyz' OR word LIKE '%xyz%'
  GROUP BY r_name, r_year, r_doi, r_pdf, c_name, r_id
  ORDER BY r_year DESC, r_name ;

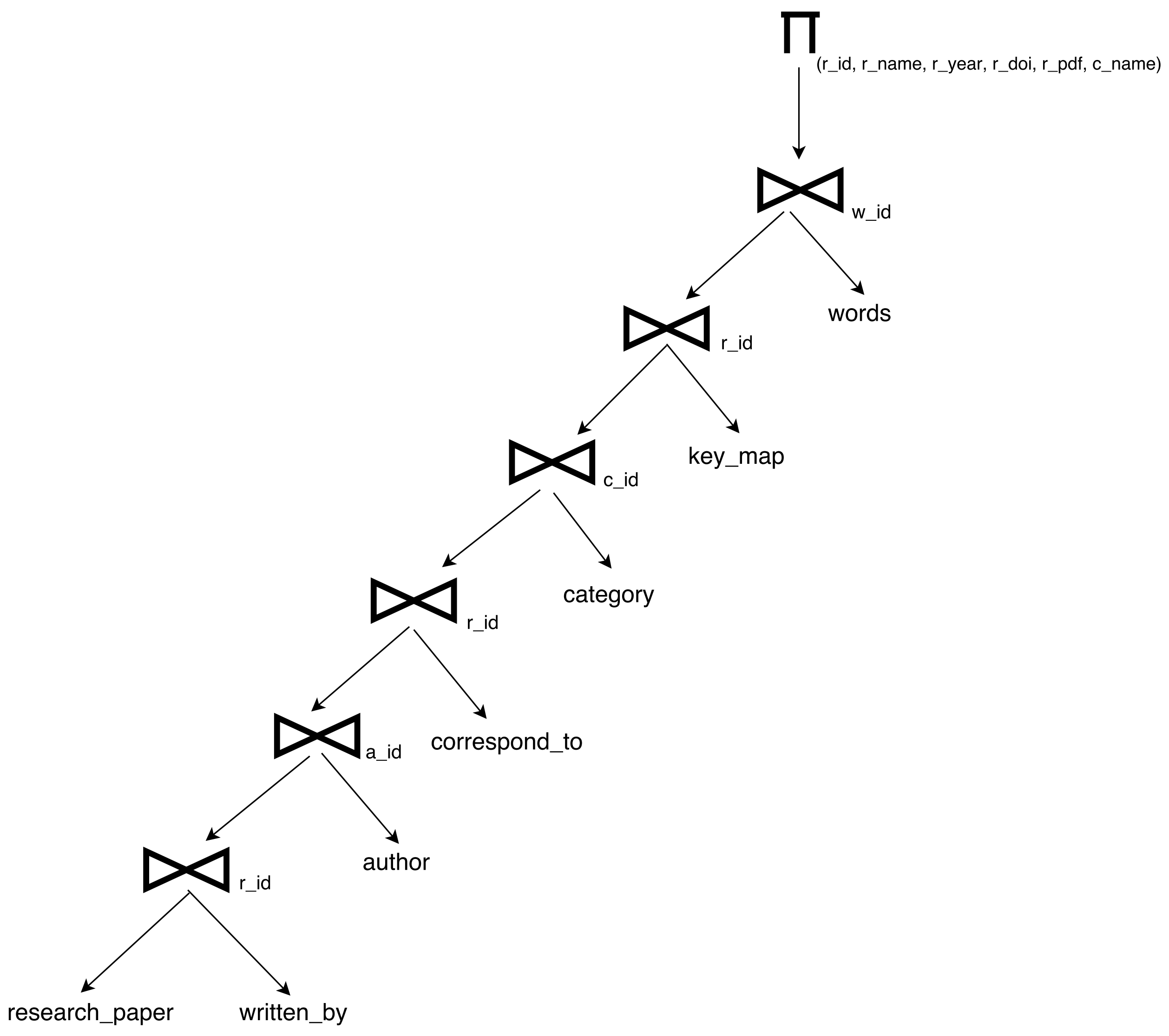


Figure 2: Query Plan Tree

In the above plan tree in order to optimize natural join, selection condition (column_name LIKE '%xyz%') can be pushed down to join operator in their corresponding table. Keeping in mind this plan tree, several indexes are made to optimize selection statements.

References

- Link to the data of research papers: https://arxiv.org/help/api/index
- Link to find the research paper from the doi: http://dx.doi.org/