Effective Keyword Search in Relational Databases for Lyrics

Navin Kumar Trivedi
Assist. Professor, Department of Computer Science & Information Technology
MGM College of Engineering & Technology, Noida, Uttar Pradesh, India

Divya Singh
B.Tech (CSe) Scholar
MGM College of Engineering & Technology, Noida, Uttar Pradesh, India

Pooja Pandey
B.Tech (CSe) Scholar
MGM College of Engineering & Technology, Noida, Uttar Pradesh, India

Gaurav Tomar
B.Tech (CSe) Scholar
MGM College of Engineering & Technology, Noida, Uttar Pradesh, India

Abstract- The main objective of this paper is to enable the users to be able to search for the desired text information in relational databases which contain information of different data types such as date and time, number, text. Even though the major Relational Database management systems have provided full-text search capabilities, they still require the users to have knowledge of the database schemas and use a Structured query language to search for the desired text information. Such a search model becomes complicated for most ordinary users. The paper aims to provide a user-friendly GUI (Graphical User-Interface) to the user where the user can easily submit the query as a combination of one or more keywords. The paper implements the following functions: (1) Answer generation, (2) Ranking of Answers (Vector Space Model).

I. INTRODUCTION

A database is a means of storing information in such a way that information can be retrieved from it. In simplest terms, a relational database is one that presents information in tables with rows and columns. A table is referred to as a relation in the sense that it is a collection of objects of the same type (rows). Data in a table can be related according to common keys or concepts, and the ability to retrieve related data from a table is the basis for the term relational database.

SQL is a language designed to be used with relational databases. There is a set of basic SQL commands that is considered standard and is used by all RDBMSs.SQL commands are divided into categories, the two main ones being Data Manipulation Language (DML) commands and Data Definition Language (DDL) commands. DML commands deal with data, either retrieving it or modifying it to keep it up-to-date. DDL commands create or change tables and other database objects such as views and indexes.

Common DML Commands used: (1) SELECT : Used to query and display data from a database. It specifies which columns to include in the result set. (2) INSERT : Adds new rows to a table. It is used to populate a newly created table or to add a new row (or rows) to an already-existing table. (3) DELETE : Removes a specified row or set of rows from a table. (4) UPDATE : Changes an existing value in a column or group of columns in a table.

Common DDL Commands used: (1) CREATE TABLE :creates a table with the column names the user provides. The user also needs to specify a type for the data in each column. Data types vary from one RDBMS to another, so a user might need to use metadata to establish the data types used by a particular database. `CREATE TABLE` is normally used less often than the data manipulation commands because a table is created only once, whereas adding or deleting rows or changing individual values generally occurs more frequently. (2) DROP TABLE :deletes all rows and removes the table definition from the database. (3) ALTER TABLE :adds or removes a column from a table. It also adds or drops table constraints and alters column attributes.

A unique key or primary key is a key that uniquely defines the characteristics of each row. The primary key has to consist of characteristics that cannot collectively be duplicated by any other row. In an entity relationship diagram of a data model, one or more unique keys may be declared for each data entity. Each unique key is composed from one or more data attributes of that data entity. The set of unique keys declared for a data entity is often referred to as the candidate keys for that data entity. From the set of candidate keys, a single unique key is selected and declared the primary key for that data entity. In an entity relationship diagram, each entity relationship uses a unique key, most often the primary key, of one data entity and copies the unique key data attributes to another data entity to which it relates. This inheritance of the unique key data attributes is referred to as a foreign key and is used to provide data access paths between data entities.

This paper has been implemented using relational databases. A relational database is a database that has a collection of tables of data items, all of which is formally described and organised according to the relational model. Data in a single table represents a relation, from which the name of the database type comes.
The major difficulty with relational database is that it contains all types of data including text, image, etc and the paper aims at retrieving text information only.
Three tables have been created for effective search of lyrics namely artist, album and song. All the three tables have been logically connected using primary key to foreign key relationship.

*Primary Key:*A primary key or unique key is a key that uniquely defines the characteristics of each row. The primary key has to consist of characteristics that cannot collectively be duplicated by any other row. The primary key constraint uniquely identifies each record in a database table. The following SQL creates a PRIMARY KEY:

Syntax

```
Create table tablename
(s_id int NOT NULL,
PRIMARY KEY (s_id));
```

*Foreign Key :*In the context of relational databases, a foreign key is a field in one table that uniquely identifies a row of another table. In other word, a foreign key is a column or a combination of columns that is used to establish and enforces a link between two tables.
The table containing the foreign key is called the referencing or child table, and the table containing the candidate key is called the referenced or parent table.
A foreign key in one table points to a PRIMARY KEY in another table. The following SQL creates a FOREIGN KEY on the column:

Syntax

```
Create table tablename1
(t_id int NOT NULL,
PRIMARY KEY (t_id),
FOREIGN KEY (s_id) REFERENCES tablename(s_id));
```

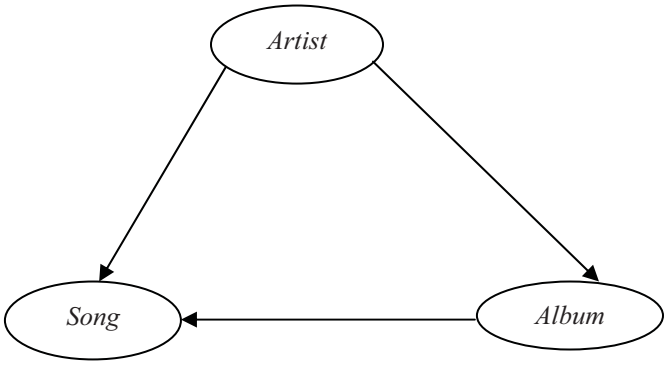

Figure 1 : The Lyrics Database Schema Graph

Relation Artist

FIELD	*KEY*	*NULL*	*DEFAULT*
Artist_id	PRIMARY KEY	No	NOT NULL
Artist_name	-	No	NOT NULL
Band	-	Yes	NULL
Type_of_song	-	Yes	NULL

Table 1 : Relation Artist

Relation Album

FIELD	*KEY*	*NULL*	*DEFAULT*
Album_id	PRIMARY KEY	No	NOT NULL
Album_name	-	No	NOT NULL
Title	-	Yes	NULL
Artist_id	FOREIGN KEY	Yes	NULL
Production_company	-	Yes	NULL

Table 2 : Relation Album

Relation Song

FIELD	*KEY*	*NULL*	*DEFAULT*
Song_id	PRIMARY KEY	No	NOT NULL
Title	-	Yes	NULL
Lyrics	-	No	NOT NULL
Album_id	FOREIGN KEY	Yes	NULL
Album_name	-	No	NOT NULL
Artist_id	FOREIGN KEY	Yes	NULL
Artist_name	-	No	NOT NULL

Table 3: Relation Song

The following SELECT query, in context of JSP has been implemented in this paper :

Select lyrics from song

where lyrics LIKE '%"+lyrics+"%' OR

artist_name LIKE '%"+lyrics+"%' OR

album_name LIKE '%"+lyrics+"%' OR

title LIKE '%"+lyrics+"%'";

Figure 2 : SQL Select Query

II. ANSWER GENERATION

In this section, we describe the framework for generation of Answers for a given user's keyword query. Section 2.1 gives a describes the JSP processing of the user's query. Section 2.2 describes an Algorithm to generate the Answers.

Processing User's Query - In this section, a set of answers is retrieved according to the given keyword query as entered by the user as an input to the system. The method employed for answer generation includes the classes of the collection framework of Java. The keyword query entered by the user is firstly stored as a string in a variable. The result for the user's query is stored in the ResultSet of the JDBC (Java Database Connectivity) API, which is then added to the reference variable ts of the type TreeSet which is a class of the collection framework of Java. TreeSet class provides an implementation of the Set interface that uses a tree for storage (objects are stored in an ascending order). The instantiated object ts of TreeSet uses a hash table for storing the data elements. Its main advantage is that it removes any duplicacy in the data, because it stores the data by using a mechanism called as hashing. In hashing, the information content of a key is used to determine a unique value, called as the hash code. The hash code is then used asan index at which the data associated with the key is stored.

Each element of TreeSet is traversed by using the iterator method. Each of the collection classes provide an iterator method that returns an iterator to the start of a collection. The data is then obtained in a string with the help of downcasting, and then stored in the PriorityQueue. The answer canbe retrieved byusing the poll method of PriorityQueue, and displayed to the user.

Algorithm for Answer Generation –

INPUT: Query Q

OUTPUT: Set of answers

1. Store the keyword query as a string.
   String lyrics = request.getParameter("search");
2. Create a reference variable of the type TreeSet.
3. Create a reference variable of the type PriorityQueue.
4. Register the Database Driver by using the class.forName() method.
5. Establish the Database connectivity by using Conection interface of the JDBC API.
6. Input the query in a String 'qry' and obtain the result in ResultSet.
   ResultSet rs=st.executeQuery(qry);
7. While(rs.next())

   {
   Ts.add(rs.getString(1));
   }
8. Traverse each element added to the TreeSet.
   Iterator itr=ts.iterator();
9. While(itr.hasNext())

```
{
        String s1=(String)itr.next();
        Pq.add(s1);
}
While(!pq.isEmpty())
{
        Out.println(pq.poll());
}
```

III. RANKING OF ANSWERS – VECTOR SPACE MODEL

Ranking Strategy -For the ranking of the retrieved answers, the information retrieval systems use a score for each answer document to show its relevance with respect to the given query. In this paper, the Vector Space Model has been used for the ranking of answers. Both answer documents and queries, are represented as a vector of terms and each item in the vector has a non-negative weight, which measures the importance of the corresponding term in the text. Thus, a similarity value between document vector D and a query vector Q is computed.

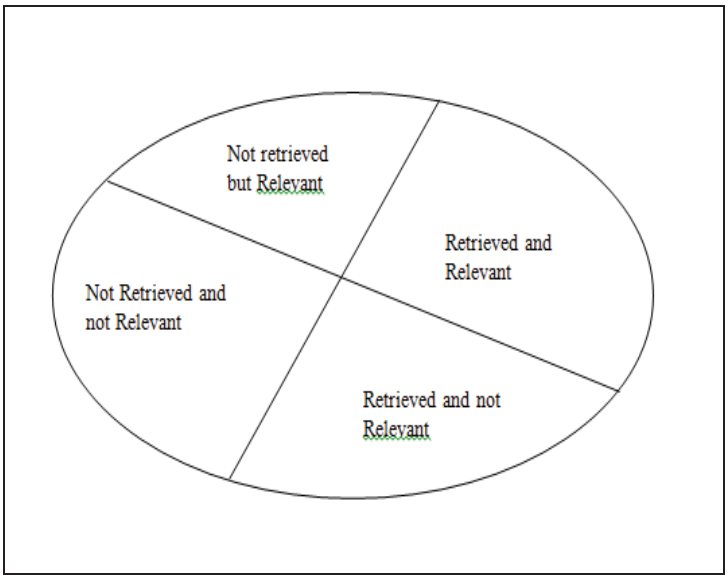

Figure 2 : Document space viewed as four quadrants

Algorithm for Ranking of Answers -

INPUT: Retrieved Answers (or documents)

OUTPUT: Similarity values for ranking of Answers

1. Initialize String str=null
2. Add the retrieved answers to PriorityQueue.
3. Iterate elements of PriorityQueue
   Iterator itr=pq.iterator();
4. Obtain the count of the number of generated answers (or documents) N and number of distinct terms M.
5. Calculate Inverse Document Frequency for M distinct terms:
   $$Idf_i = \log_2(N/df)$$
   (df is the document frequency i.e. the number of documents that the $term_i$ occurs in)
6. Calculate the term frequency for M distinct terms in each document:
   $$Tf_{ij} = \text{number of occurrences of } term_i \text{ in } document_j$$

7. Compute (tf * idf) matrix T1 for N documents:

$$T1_{ij} = idf_i * tf_{ij}$$

8. Compute (tf * idf) matrix T2 for Query Q:
   $T2_i$ = (number of occurrences of $term_i$ in Q/Maximum frequency in Q) * idf_i

9. Calculate the normalized length of each document and Query Q:

$$|D|len = \sqrt{\sum_{i}^{M} \sum_{j}^{N} (T1ij)^2}$$

$$Qlen = \sqrt{\sum_{i}^{M} (T2i)^2}$$

10. Compute similarity values for each document:

$$Sim(Dj, Q) = \sum_{i}^{M} \sum_{j=1}^{N} [T1ij * T2i] / [|D|len * Qlen]$$

11. If ($Sim(D_j,Q) > Sim(D_k,Q)$)
    Then document D_j is ranked higher than document D_k.

IV. EXPERIMENTATION

Data Set -

Database:We use a Lyrics database in this experimental setup. We collected the data from a various Lyrics websites and converted that data into a relational database schema, as it has been described in the previous sections.

Query Set -

A set of sample queries have been used to verify the output of the experimental setup. The user's keyword query can contain the artist name, album name, or the song title, for which the lyrics have to be retrieved. The Experimental setup successfully displays the result as the desired Lyrics to the user, on a simply designed Graphical User Interface.

V. CONCLUSION AND FUTURE WORK

It can be thus concluded that the system functions effectively, with theimplementation of theVector space model used for the ranking of answers. An effective lyrics database is developed by using the Oracle database 10g, and a simple, yet an attractive Graphical user interface is designed by the use of HTML and JSP technology. With all the inherent advantages of these technologies being used in the paper, the system will be able to deliver its satisfactory services to a wide range of users, ranging from an ordinary user to any field to which a user belongs.

The system has a vast future scope, as when deployed to the users, it can also include the features such as providing the users with the audio version of the songs for which the lyrics have been searched. This is possible with the use of the Oracle Database 10g that has been used in the project. One of the interesting features of the 10g version is that it not only stores the text data but also other types of data in different formats such as spreadsheets, power point presentations, multimedia (audio, video, pictures, images, graphics). Also, the user can be provided to create his login id and password for a personalized account so that he can save his searches as desired, and also post lyrics to a song that will be stored in the database, so that other users can search for it as and when required.

REFERENCES

[1] Bei Yu, Guoliang Li, Karen Sollins: Effective Keyword-based Selection of Relational Databases. SIGMOD'07, June 12-14, 2007, Beijing, China. Copyright 2007 ACM 978-1-59593-686-8/07/0006.
[2] D. Grossman and O. Frieder, Information Retrieval: Algorithms and Heuristics, Springer Publishers, 2nd Edition 2004
[3] Fang Liu, Clement Yu, WeiyiMeng, Abdur Chowdhury: Effective Keyword Search in Relational Databases. SIGMOD 2006, June 27-29, 2006, Chicago, Illinois, USA Copyright 2006 ACM 1-59593-256-9/06/0006
[4] G.Salton and M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, New York, 1983.

[5] Hulgeri, A., Bhalotia, G., Nakhe, C., Chakrabarti, S., Sudarshan, S. Keyword search in databases. IEEE Data Engineering Bulletin 24(3): 22–32, 2001.
[6] Jeffrey Xu Yu, Lu Qin, Lijun Chang: Keyword Search in Relational Databases: A Survey. Copyright 2010 IEEE. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering
[7] SinaFakhraree, FarshadFotouhi: Effective Keyword Search over Relational Databases Considering keywords proximity and keywords N-grams. 1529-4188/11 2011 IEEE DOI 10.1109/DEXA.2011.28
[8] S Agrawal, S Chaudhuri, G Das: DBXplorer: A system for keyword-based search over relational databases. ICDE 2002
[9] Hulgeri, A., Bhalotia, G., Nakhe, C.,Chakrabarti, S., Sudarshan, S. Keyword search in databases. IEEE Data Engineering Bulletin 24(3): 22–32, 2001.
[10] UtharnBuranasaksee, KriengkraiPorkaew, UmapornSupasitthimethee: Answer Aggregation for Keyword Searchover Relational Databases. 978-0-7695-4042-9/10 2010 IEEE DOI 10.1109/ICCNT.2010.130
[11] V. Hristidis, L. Gravano, Y. Papakonstantinou: Efficient IR-Style Keyword Search over Relational Databases. VLDB 2003
[12] V.Hristidis and Y. Papakonstantinou. DISCOVER: Keyword search in relational databases. VLDB 2002.
[13] HyperText Markup Language http://en.wikipedia.org/wiki/HTML
[14] Primary Key and Foreign Key http://en.wikipedia.org/wiki/Unique_key
[15] Relational Databases http://docs.oracle.com/javase/tutorial/jdbc/overview/database.html
[16] HTML and CSS http://www.w3.org/standards/webdesign/htmlcss
[17] http://www.tutorialspoint.com/jsp/jsp_overview.htm
[18] JSP Architecture http://www.tutorialspoint.com/jsp/jsp_architecture.htm
[19] JSP Form Processing http://www.tutorialspoint.com/jsp/jsp_form_processing.htm
[20] User Front End http://www.tutorialspoint.com/jsp/jsp_form_processing.htm