

Applications of SQL for Informetric Data Processing

Abstract: Many informetric phenomena lend themselves to ready adaptation to relational DBMS environments. SQL, the standard language used for constructing and querying relational databases, provides useful tools for processing of informetric data. The author demonstrates the applications and some limitations of SQL for efficient organization and tabulation of raw informetric data.

Résumé : Plusieurs phénomènes infométriques se prêtent à une adaptation rapide aux environnements relationnels DBMS. SQL, le langage standard utilisé pour la construction et la recherche des bases de données relationnelles, offre des outils utiles pour le traitement des données infométriques. L'auteur démontre les applications et certaines restrictions de SQL pour l'organisation efficace et la tabulation des données infométriques brutes.

1. OVERVIEW

Informetrics, the quantitative study of recorded discourse, has long been a core area of investigation within information science. Informetrics research attempts to identify, generalize, understand, and ultimately apply regularities in the way information and their containers are produced and used. As a research area that relies on the collection and analysis of large data sets, effective means for the storage, representation and processing of data are needed. In earlier times, this was handled manually through painstaking collection of data from print sources. As more resources have become available in electronic form, this process has been streamlined through the development of software that identifies, collects and manipulates the data as needed.

Structured Query Language (SQL) is an ANSI standard for accessing and manipulating relational database content (Burlesan et al., 2003). It is widely used in industry and is supported by major database management systems (DBMSs). DBMSs are primarily used to house structured textual and numeric data that have been compartmentalized into records and fields. This compartmentalization facilitates access and retrieval of data contents. Through the use of the SQL data manipulation language, one is able to readily summarize and process content. The power associated with these capabilities makes relational DBMSs and, specifically, SQL well suited for the storage and manipulation of informetric data. Although XML and object-oriented database querying languages are under development, the relational paradigm will continue to be the primary environment for most database users at least for the near future.

There have been several research studies that have examined relational database construction for the housing of informetric data, but the applications have been for a specific purpose or have focused on the storage of data and not their manipulation. A smaller number of studies have examined the scope of relational database modelling and how the querying abilities afforded by SQL may be used to extract data and frequency distributions meaningful for informetric analysis. This study presents applications of the relational model and SQL for informetric data processing. Working with data in raw form

and aggregating data as needed through querying instead of aggregating the data outright has its advantages. The ability to view data in different ways through SQL data grouping makes it possible to analyze patterns from different perspectives. Examples are provided for how raw data may be isolated, grouped and tallied through the use of SQL statements, which ultimately saves researchers' time. In the author's previous research examining large data sets representing query data from a search engine (Ajiferuke & Wolfram, 2004) and resource usage in a public access full text database environment (Ajiferuke, Wolfram, & Xie, 2004) the storage and querying of the raw data sets in a relational DBMS environment have been quite useful. Examples are taken from these environments to demonstrate the utility of the DBMS approach. These examples are easily extended to other informetric data environments involving classic relationships, citation/sitations, or information object content. Limitations inherent in relational database modelling and SQL querying, however, may require that additional processing steps be used to achieve the final consolidated data.

2. SOFTWARE TOOLS FOR DATA ANALYSIS

There have been a number of examples of in-house and proprietary software developed since the late 1980s that have been designed to tabulate informetric characteristics of primarily bibliographic data sets. These have included: the Bibliometrics Toolbox (available as freeware from <ftp://ftp.u.washington.edu/public/tabrooks/toolbox/>), which contains routines for Bradford analysis and indices of concentration; DataView (<http://crrm.u-3mrs.fr/commercial/software/dataview/dataview.html>), which automatically analyzes bibliometric characteristics of tagged bibliographic data, and; Bibexcel (available as freeware from: <http://www.umu.se/inforsk/Bibexcel/>), which contains various routines for the analysis of bibliographic data. Sotolongo-Aguilar, Suárez-Balseiro, & Guzmán-Sánchez (2000) cite further examples of software for informetric analysis and present a five-step methodology called MOBIS-ProSoft that relies on proprietary and in-house software to collect, convert, and analyze data. Finally, additional examples of software for informetric research may be found on the *Cybermetrics* Website (<http://www.cindoc.csic.es/cybermetrics/links18.html>). Many of the software examples developed are application specific--that is, they were developed for specific types of analysis such as author productivity or citation analysis. Although limited, for those applications they can be quite helpful.

Data mining software has become popular for analyzing patterns in large data sets. Tools provided include routines for predictive modelling, neural network and decision trees development, clustering and visualization, in addition to mainstream statistical methods. Although developed primarily for business analysis or operations research and not specifically for informetric data analysis, there are common techniques available through data mining software such as clustering and visualization that are of interest to researchers in informetrics.

A third category of software used for transaction log analysis (particularly of websites) offers an additional set of tools for data collection and analysis. Reported summaries of data, however, are often not in formats that are useful for informetric analysis due to the aggregate nature of the data presentation.

Whether one uses raw data or relies on aggregate data will depend on the type of analysis to be performed. In most cases, data modelling routines ultimately rely on aggregated data, whether the data relate to frequency distributions or co-occurrences. The advantage of leaving collected data in their raw form is that connections between data items are not lost. Each data observation retains its “identity” and is not lumped together with the other observations until needed. Ultimately, data will be aggregated to develop distributions based on frequencies or ranks. Structured storage methods, such as those available through spreadsheet or database software, are useful for retaining data characteristics for flexible data manipulation and aggregation on demand.

Spreadsheet software has evolved to include many useful mathematical and statistical operations for preprocessing of data (at least), if not for modelling purposes as well. The user friendly plotting capabilities with logarithmic scaling alone make this software invaluable for some types of modelling techniques such as least squares estimation. Built-in tools for data consolidation and sorting, along with tools for the calculation of descriptive and inferential statistics allow data analysis to be undertaken with minimal effort. Statistical software such as SPSS, SAS, or Minitab will provide a broader array of statistical options than is usually available in commercial spreadsheet software. One drawback of spreadsheet software for large, raw data sets is that there is at present a limit of 64K rows of observations in most packages. This is not a problem for aggregate data already in distribution form where the number of unique data types usually consists of fewer than several hundred rows even for large data sets. For large data sets that exceed the limits of spreadsheet software, such as index contents or user log data records, which may consist of millions of observations, database management system software provides a better alternative (Wolfram, 2003, Chapter 4).

Aside from spreadsheet software, no structured storage mechanism other than a DBMS environment provides a comparable level of flexibility for the representation and aggregation of raw data. The ability to create tables to house data and establish links between tables permits easy access and maintenance of data observations and their relationships. In addition to numeric data manipulation, DBMSs provide advantages for large textual data sets, where parsing of textual strings for distinct words (*types*) and different occurrences of words (*tokens*) may be carried out using the extensive programming capabilities built into many DBMSs. Once these data are stored, tabulation of distributions can be easily performed using SQL queries.

3. PREVIOUS STUDIES

Several researchers have investigated selected aspects of the applicability of DBMSs for informetric data storage and analysis. Among the earliest, Fernández, Cabrero, Zulueta, and Gómez (1993), outline the construction of a relational database for storing bibliographic and demographic data for informetric analysis. The focus of the article is on the structuring of the tables to allow relationships to be built between tables from which bibliometric indicators may be tabulated. Expanding on this idea, Small (1995), provides details of relational database modelling for bibliographic data and appears to be the first to demonstrate how SQL may be applied to perform citation network analysis within a relational DBMS environment where links are represented as incoming (cited) or outgoing (cites) links to and from a given bibliographic record. This study presents a useful demonstration of the power of SQL for tabulating data relationships for

informetric analysis, but focuses only on citation analysis.

Recently, more sophisticated database designs that include complete document representation within the database have been explored. Järvelin, Ingwersen, and Niemi (2000) outline a database model for storing bibliographic references and extracting data for informetric analysis. Their representation of bibliographic references as complex objects permits the study of author, journal, article, citation, and co-citation-based analyses using a specially developed querying approach. Querying of data sets is expanded in Niemi, Hirvonen, and Järvelin (2003), where the authors outline how an online analytic processing (OLAP) approach may be used to model multidimensional data. The authors develop a declarative query language to manipulate the data. As a proposed query language, it awaits further implementation, but is promising for informetric analysis. Given the limited literature at present on the topic, the intention of the present paper is to provide a similar demonstration of the power inherent in database modelling of informetric data, but to rely on a widely available approach (i.e., SQL) for this purpose.

4. INFORMATION PRODUCTION PROCESSES AND RELATIONAL DATABASE ENVIRONMENTS

Informetric phenomena can be described in terms of information production processes (IPPs). An IPP, defined by Egghe (1990), consists of *source* entities that produce *items*. Strict tallies of an informetric process such as the growth in the total number of documents in a collection over time may be thought of as a one-dimensional IPP. Identification of the individual characteristics in this case is not important; the focus is on the characteristic of the overall set of entities. Two-dimensional IPPs involve source and item relationships as one-to-many relationships (e.g., the number sources that produce x items), represented by classic informetric laws such as Lotka's law (author productivity) and Bradford's law (journal productivity), or in the case of query analysis (Ajiferuke & Wolfram, 2004), the production of query and term occurrences (tokens):

<i>Source</i>		<i>Item</i>
Users	→	Queries
Queries	→	Terms
Terms	→	Tokens

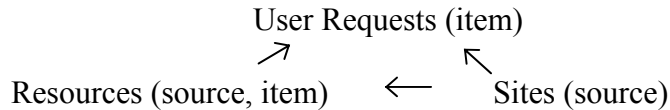
It is possible to have higher dimensional IPPs, where multiple source and item relationships exist or may be derived, or where items produced by a source in turn serve as sources to produce a new set of items. Egghe (1994, 2004) describes three-dimensional informetrics in a linear and non-linear forms. In the area of user query processes, queries (source) produce terms (item), which in turn produce tokens (item):

Queries (source) → Terms (item, source) → Tokens (item)

Taking this further, where users are responsible for the creation of queries, an additional dimension can be added:

Users (source) → Queries (item, source) → Terms (item, source) → Tokens (item)

In some environments complex data relationships may be viewed from a non-linear perspective. In the case of Ajiferuke, Wolfram, and Xie (2004), who examine the characteristics online resource requests from an online site (usage) and site visitation data to a site, the IPPs may be viewed such that resources (source) generate requests by users (item), or sites (source) generate visits by users (item). The site (source) in turn also produces resources (item):



From a data modelling perspective, these can be thought of as many-to-many relationships (i.e., where a user can visit many sites and a site can attract many different users), which for simplicity may be decomposed to two or more one-to-many relationships. These types of modelling relationships translate readily to the relational DBMS environment.

As an example of multi-dimensional IPP database design, consider the IPP of IR system query session data. As outlined above, the environment represents a multi-dimensional IPP, where users represent sources of queries, queries represent sources of terms and terms represent sources of tokens. From these data relationships, a number of data distributions can be tabulated. In preparation, these data are stored as a set of three tables within a relational DBMS (see Figure 1). The primary source file consists of raw queries and user identifiers documents. Each query is parsed for identifiable terms. This requires some programming, but with the array of string functions available with many DBMSs, this is not problematic. Identified terms are then stored in a separate query term table. The ordinality of the relationship between terms and queries is clearly many-to-many (i.e., one query may have many terms and one term may be associated with many queries). By reducing this relationship to two one-to-many relationships required by relational DBMSs, the difficulty of dealing with many-to-many relationships is removed. A third table is created, representing query term tokens, to connect query terms to queries. What results is similar to an inverted file structure used in indexing documents in IR systems. In essence, we have built a simple storage component for an IR system using the DBMS. This type of structure may be applied to many types of informetric data. The decomposition of the data fields into component parts such as individual authors, title words, source publication, and subject headings could be dealt with in a similar way. Commercial textbase software such as Inmagic's DBTextWorks could also be used to isolate terms and tabulate term distributions, but manipulation of the indexes is limited to exporting them as text files and there is limited control over the definition of term delimiters. Also, without the availability of SQL querying capabilities, additional processing or programming is needed.

[illegible]

Figure 1 – Query Analysis Tables

Familiarity with the data description language of SQL or a user-friendly graphical interface for table construction is needed to establish the tables and relationships between data components. The remaining focus here, however, will be on the data manipulation language for querying database contents.

5. EXTRACTING IPP DATA USING SQL

In addition to fundamental operations such as SELECT and JOIN, which allow table fields and different tables to be combined, SQL has a number of useful functions such as Count() that can perform computational tasks on the extracted data. When combined with the feature of aliasing, which allows operations to be enacted on multiple copies of a table, as well as grouping and ordering features on specified fields, output can be presented as frequency distributions ready for modelling without any additional programming.

One-dimensional IPPs that report totals are rudimentary, so examples provided begin with two-dimensional IPPs. Although the examples relate to the term and query data, they are analogous to more traditional informetric data such as author and journal productivity, where field names need only be changed. The examples provided are based on MS Access SQL statements (with some modifications) and may vary somewhat from one SQL implementation to another.

Once the raw data have been processed as outlined above, SQL queries may be constructed to tabulate the indexing exhaustivity distributions (or distribution of terms used per query) by tallying and grouping the number of terms used in each query from the Queries table.

Example 1

```
SELECT Size, Count(Size)
FROM Queries
GROUP BY Size
ORDER BY Size;
```

The resulting output will be a frequency distribution for the number of terms occurring once, twice, etc. and the distribution for queries containing one term, two terms, etc. A term frequency distribution may be calculated the same way using the Terms table.

Example 2

```
SELECT Freq, Count(Freq)
FROM Terms
GROUP BY Freq
ORDER BY Freq;
```

The generation of the distribution of queries submitted per user becomes a two-step procedure, where intermediate calculations are stored in a temporary table. In the case of MS Access, two queries can be used, where the output of the first query serves as the input for the second query. Using a temporary table, which is a more standard approach for different dialects of SQL, the first SELECT statement tallies occurrences for each user, which are stored in QueryTemp. The second statement then calculates the frequency distribution of the values in QueryTemp.

Example 3

```
CREATE TABLE QueryTemp
(SELECT User_id, Count(User_id) AS Id_count
FROM Queries
GROUP BY User_id);

SELECT Id_count, Count(Id_count)
FROM QueryTemp
GROUP BY Id_count
ORDER BY Id_count;
```

A variation of this approach can be used to calculate the distribution of query frequencies across users.

A more elaborate calculation is the determination of the frequency distribution of co-occurring pairs of terms within queries. This can also be extracted from the existing structure. It is a multi-step process that involves performing a sub-query with a JOIN operation on the Tokens table with an alias of itself (Tokens1), then ordering the term pairs so that co-occurrences of Term A followed by Term B are not counted as distinct from co-occurrences Term B followed by Term A within and across queries. The result of these steps, stored in a temporary table (Cooccur), can then be tallied.

Example 4

```

CREATE TABLE Cooccur
(SELECT Count(ThePair)
FROM
  (SELECT Tokens.Term_id & " " & Tokens1.Term_id AS ThePair
   FROM Tokens INNER JOIN Tokens1
   ON Tokens.Query_id = Tokens1.Query_id and
       Tokens.Term_id < Tokens1.Term_id)
GROUP BY ThePair);

SELECT PairCount, Count(PairCount)
FROM Cooccur
GROUP BY PairCount
ORDER BY PairCount;

```

The examples provided so far represent classic two-dimensional IPP relationships. With more detailed querying, indirect relationships representing three-dimensional and higher IPP distributions may be extracted from the relational tables. To demonstrate, consider the frequency distribution of distinct terms for users. This represents the relationship of the vocabulary size generated by a user over the course of a search session consisting of one or more queries. These data are not directly available by a single query, but may be extracted through additional selection and joining of data. In the pursuit of two-dimensional IPP data in aggregated form, these data could easily be ignored and would be unobtainable once the more frequently studied data are tabulated and the raw data are discarded. By performing a JOIN operation between the Tokens and Queries tables, selecting only those that are distinct for each user identifier (stored in temporary table UserVocabularySize), and then tallying the occurrences of each vocabulary size across user identifiers in the same way as earlier, the frequency distribution of user vocabulary size may be determined.

Example 5

```

CREATE TABLE UserVocabularySize
(SELECT Count(UserTerms) AS UserWordCount
FROM (SELECT DISTINCT Tokens.Term_id, Queries.User_id
      AS UserTerms
      FROM Tokens INNER JOIN Queries ON
      Tokens.Query_id = Queries.Query_id)
GROUP BY UserTerms);

SELECT UserWordCount, Count(UserWordCount)
FROM UserVocabularySize
GROUP BY UserWordCount
ORDER BY UserWordCount

```

As stated earlier, the methods used for the above examples of query data can be applied to more traditional informetric environments involving authors, journals, articles, references/citations and time, from which author productivity, journal productivity, citation/co-citation and obsolescence data may be obtained. The field names may change,

but the relationships between the different data types remain the same. Classic citation analysis, which involves cites/cited relationship tables, can also be performed. This approach is already outlined in Small (1995). Essentially, the method is similar to Example 4, where aliasing of a citation table allows co-citations and cites/cited relationships to be mapped between the citation table and its alias. Small's examples translate readily to the newer hyperlink analysis, which examines inter-document linkage structure in hypermedia environments. However, caution must be exercised in the hypermedia environment. Although the concept of citation trails that provide chains of links to earlier works is feasible in citation analysis, the non-temporal nature of hyperlinks, where circular sets of links are commonplace, must be considered.

6. DISCUSSION & CONCLUSIONS

This paper has demonstrated by example how IPP data and relationships can translate readily to a relational DBMS environment for efficient processing of frequency distribution relationships. The DBMS environment is particularly useful for large datasets that cannot be readily accommodated in other structured environments such as spreadsheet software. Although counts of different processes are the desired outcome in many cases, a DBMS offers the ability to summarize data in different formats as they are needed while leaving raw data and their relationships intact, which provides much more flexibility for analysis.

There are still inherent limitations in SQL's ability to handle complex data manipulation tasks. Preprocessing of non-numeric data may be needed to identify sources, items, and occurrences. In the above examples, queries were first parsed for individual terms using a different programming environment. This may not be necessary for some textual environments where the field definitions and raw data isolate each type (for example, authors stored in separate fields for author-based analyses). More recent versions of relational DBMS environments that support SQL such as MySQL, Oracle, and PostgreSQL provide string functions that are useful for parsing textual content, but do not provide the same functionality or control as full application development languages. Still, when used in conjunction with these languages where SQL statements are embedded in program routines, fewer procedural steps need to be specified.

A second issue of relevance to informetrics research is the data format. Examples provided here use a Lotkaian (size-frequency) approach to data representation. Some types of informetric studies, such as word frequency distribution investigations that use a classic Zipf approach, will organize data in a rank-frequency format. Data within DBMSs are easily sorted by frequency, but calculation of ranks may be more problematic, particularly if average ranks are to be used for data observations of the same frequency. Again, newer versions of SQL support some ranking features, but with several ways of dealing with tied ranks, the desired ranking method may not be supported without additional programming.

Researchers are developing more sophisticated ways to deal with multidimensional data. As research develops along the lines of a type-token/taken approach (Egghe, 2003), which incorporates usage aspects of traditional type-token relationships, more elaborate analytical approaches will be needed. Ultimately, SQL on its own does not provide all the functionality needed to perform these tasks. Newer generations of SQL are including

OLAP features, which extend the ability to work with complex data sets, but these are still in the early stages of development. Implementation of frequency distribution calculations is also now available for some implementations of SQL. Other approaches, such as the development of database and querying models proposed by Järvelin, Ingwersen, and Niemi (2000) and Niemi, Hirvonen, and Järvelin (2003) will offer additional powerful tools for storage and analysis of complex data, but the general application of these approaches has not yet seen wide use. At present, SQL remains a viable and readily accessible tool for processing many types of informetric data.

REFERENCES

Ajiferuke, Isola, and Dietmar Wolfram. 2004. Informetric modelling of Internet search and browsing characteristics. *Canadian Journal of Information and Library Science* 28: 1-16.

Ajiferuke, Isola, Dietmar Wolfram, and Hong Xie. 2004. Modelling Website Visitation and Resource Usage Characteristics by IP Address Data. In *Proceedings of the 32nd Annual Conference of the Canadian Association for Information Science*. Available: http://www.cais-acsi.ca/proceedings/2004/ajiferuke_2004.pdf [2005, March 10th].

Burlesan, Donald K., Joe Celko, John Paul Cook, and Peter Gultzan. 2003. *Advanced SQL Database Programmer Handbook*. BMC Software and DBAZine.

Egghe, Leo. 1990. The duality of informetric systems with applications to empirical laws. *Journal of Information Science* 16: 17-27.

Egghe, Leo. 1994. Bridging the gaps: Conceptual discussions on informetrics. *Scientometrics* 30: 35-47.

Egghe, Leo. 2003. Type/token-taken informetrics. *Journal of the American Society for Information Science and Technology* 54: 603-610.

Egghe, Leo. 2004. Positive reinforcement and 3-dimensional informetrics. *Scientometrics* 60: 497-509.

Fernández, M.T., A. Cabrero, M.A. Zulueta, and I. Gómez, 1993. Constructing a relational database for bibliometric analysis. *Research Evaluation* 3: 55-62.

Järvelin, Kalervo, Peter Ingwersen, and Tiemo Niemi. 2000. A user-oriented interface for generalised informetric analysis based on applying advanced data modelling techniques. *Journal of Documentation* 56: 250-278.

Niemi, Tiemo, Lasse Hirvonen, and Kalervo Järvelin. 2003. Multidimensional data model and query language for informetrics. *Journal of the American Society for Information Science and Technology* 54: 939-951.

Small, Henry. 1995. Relational bibliometrics. In *Proceedings of the Fifth Biennial Conference of the International Society for Scientometrics and Informetrics*, eds. M.E. D.

Koenig and A. Bookstein (pp. 525-532). Medford, NJ: Learned Information, Inc.

Sotolongo-Aguilar, G. R., Suárez-Balseiro, C. A., & Guzmán-Sánchez, M. V. (2000). Modular bibliometric information system with proprietary software (MOBIS-ProSoft): A versatile approach to bibliometric research tools. *LIBRES: Library and Information Science Research*, 10(2). Available: <http://libres.curtain.edu.au/libres10n2/modular.htm> [2005, March 15th].

Wolfram, Dietmar. 2003. *Applied informetrics for information retrieval research*. Westport, CT: Libraries Unlimited.