ISSN 2278-3091
Volume 7, No.2, March - April 2018
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse01722018.pdf
https://doi.org/10.30534/ijatcse/2018/01722018

SpreadDB: Spreadsheet-Based User Interface for Querying and Updating Data of External Databases

Somchai Chatvichienchai[1], Yu Kawasaki[2]
[1]Dept. of Information Security, Faculty of Information System, University of Nagasaki, Japan, somchaic@sun.ac.jp
[2]Dept. of Information and Media Studies, Faculty of Global Communication, University of Nagasaki, Japan, b2214016@sun.ac.jp

ABSTRACT

Spreadsheet software is a software application capable of organizing, storing and analyzing data in tabular form. However, it has many limitations such as poor performance on large data sets, and lack of efficient and secured data sharing. In order to overcome these limitations, many organizations promote their users to store their data into databases. However, databases lack in the ease of use since they request database users to use SQL which is a standard language for querying and editing information stored in the databases. Since SQL is difficult for many end users to learn, database utilization of some parts of organizations does not progress. In order to relief users' burden of learning SQL, we propose SpreadDB which is a spreadsheet-based user interface for querying and updating database data. SpreadDB enables users to design spreadsheet templates used to perform data query and data update. We also present security measures of SpreadDB that prevent unauthorized persons from accessing and modifying database data.

Keywords: Database, Query, Security, Spreadsheet, User Interface.

1. INTRODUCTION

Spreadsheet software is a software application capable of organizing, storing and analyzing data in tabular form. However, it has many limitations such as poor performance on large data sets, and lack of efficient and secured data sharing. In order to overcome these limitations, many organizations promote their users to store their data into relational databases (RDBs, for short) [1]. However, RDB lack in the ease of use since they request database users to use SQL (Structured Query Language) [2] which is a standard language for querying and editing information stored in the RDBs. When an organization needs a new database, it typically hires a contractor to build it or buys a heavily supported product customized to its industry sector. Usually, the organization already owns all the data it wants to put in the database. But few organizations have in-house database experts who write complex queries in SQL or some other database scripting language to pull data from many different sources; to filter, sort, combine, and otherwise manipulate it; and to display it in an easy-to-read format. In companies that have not enough budgets to hire a contractor to develop database application programs, many users employ spreadsheet software to retrieve data of external databases such as MySQL, PostgreSQL, SQL Server, etc. However, almost all spreadsheet software have the following two drawbacks. The first is that query condition definition is troublesome task since users need to specify database tables, the fields of selected table that are outputted and a query condition that is based on field values. If users want to narrow the scope of query output, they need to redefine the query condition again. The second is that current spreadsheet software don't provide data update function to external databases. Then, it is necessary to develop a program to realize that function. However, the cost of program development for data updating is proportional to the number of tables of the database.

The objective of this paper is to propose SpreadDB which is a spreadsheet-based user interface for typical users who don't know SQL in order to query and update data of RDBs. As Microsoft Excel [3] is one of popular spreadsheet software, we develop SpreadDB to operate on Microsoft Excel so that many users get used to its operations. In addition, SpreadDB allows users to create spreadsheet templates (templates, for short) for querying and updating data of RDBs. Therefore, Excel spreadsheet can be always up to date with the current data from the back end database.

2. SYSTEM REQUIREMENT AND ARCHITECTURE

In order to achieve the research objective, the proposed system should have following properties.

- **High Database Connectivity:** The proposed system should be able to connect to any external RDBs.
- **Easy to Use:** The proposed system should allow users to query and update data of database data without the need to know SQL.
- **High Scalability:** The proposed system should be able to handle new database tables without additional program development.

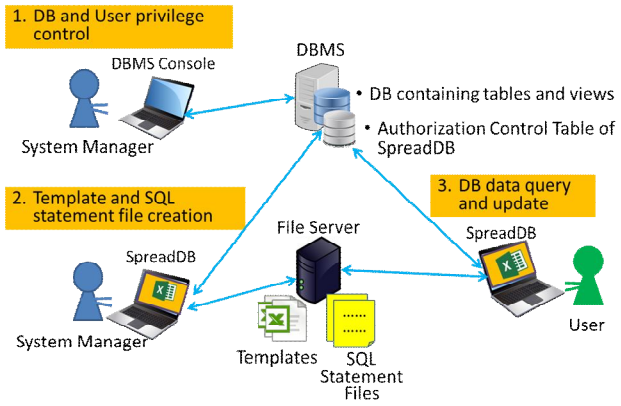


Figure 1: Architecture of SpreadDB

- **Secure Data Sharing:** The proposed system should have security countermeasure for preventing unauthorized data retrieval and modification.

We design SpreadDB that has the above properties. Its architecture is shown in Fig.1. We classify users into the following two types: a system manager and typical users (users, for short). The main tasks of a system manager are classified as follows.

(1) Maintaining tables and views of RDBs via RDBMS (Relational Database Management System) console.

(2) Maintaining database user accounts and user access rights via DBMS console.

(3) Generating Templates and corresponding SQL statement files used for querying and updating DB data. The templates and SQL statement files are stored in a file server so that they can be shared by users.

A template allows users to perform the following functions.

- Search function which allows users to perform search whose condition is defined by entering values and comparison operators at table items that are beside table header.
- Record insertion function that allows users to add a new record into the database table.
- Update function that allows users to rewrite data of highlighted rows to the database table.
- Delete function that allows users to delete highlighted rows from the database table.

3. ISSUES IN DEVELOPING SPREADDB

3.1 How to enable SpreadDB to have high database connectivity?

In order to enable SpreadDB connecting to any external databases, SpreadDB is connected with the databases thru

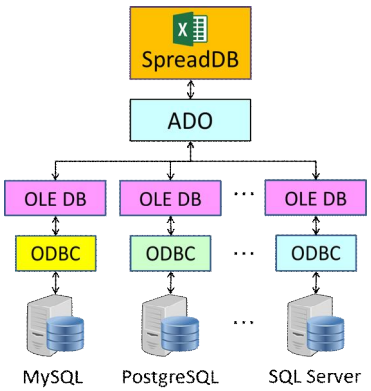
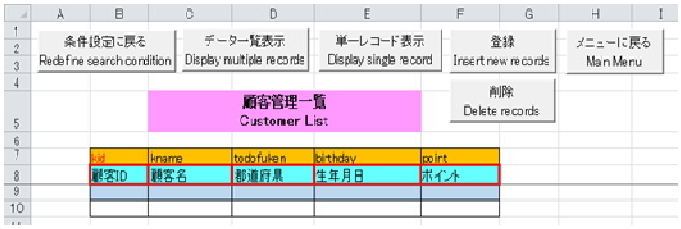
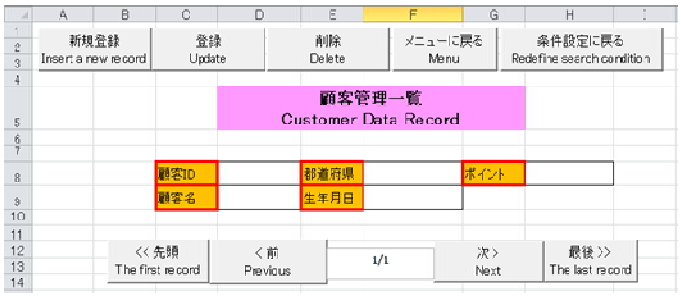


Figure 2: The connections between SpreadDB and external databases.

ADO (ActiveX Data Objects) [4]. Figure 2 shows the connections between SpreadDB and databases via components of Windows database technologies. ADO, which is built on top of the OLE DB (Object Linking and Embedding Database) [5]. OLE DB is one of several database interface technologies integrated into Microsoft Windows operating systems. ADO allows us to develop programs of SpreadDB that access data without knowing how the database is implemented. The OLE DB interfaces to databases directly through a specific ODBC (Open Database Connectivity) driver for the underlying RDBMS. System manager needs not to install ADO in computers of users because ADO is already installed in Excel VBA [6] which is the running environment of SpreadDB. However, system managers have to install ODBC and OLD DB of the target database into the computers of the users.



(a)

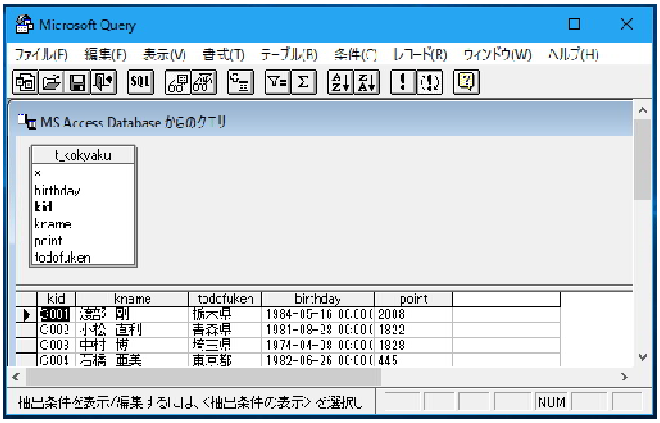


(b)

Figure 3: (a) An example of a multiple-record template for displaying customer list from database table *t_kokyaku* and (b) An example of a single-record template for displaying a customer record from database table *t_kokyaku*.

```
SELECT t_kokyaku.kid, t_kokyaku.kname,
       t_kokyaku.todofuken, t_kokyaku.birthday,
       t_kokyaku.point
FROM t_kokyaku t_kokyaku
ORDER BY t_kokyaku.kid
```

(a) An example of content of a SQL statement file used to query all data of database table t_kokyaku.



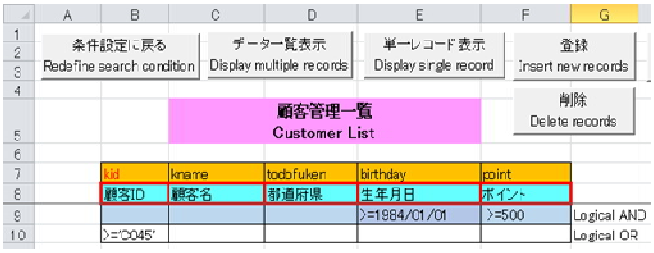
(b) An example of a screenshot of Microsoft Query Wizard that defines a SQL statement file of (a).

Figure 4: An example of a SQL statement file generated by Microsoft Query.

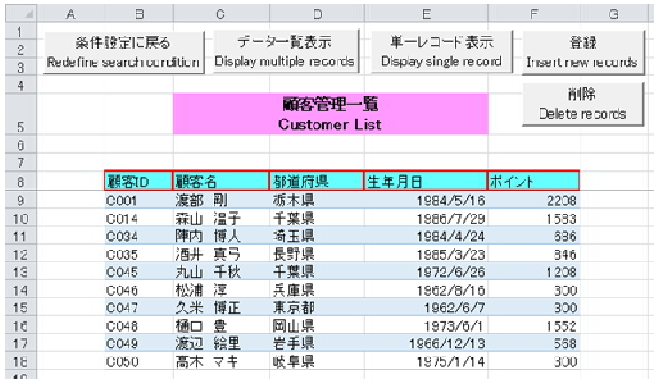
3.2 How to enable SpreadDB to have high scalability?

The main idea is to pass SQL statement to SpreadDB and to guide SpreadDB to output query results to the specified cells of Excel worksheet. This idea is realized by defining a pair of a template and a basic SQL statement. In order to guide SpreadDB to output the query result to target worksheet cells, system manager needs to define a mapping from attributes of the SQL statement to the target cells of the template. This mapping will be explained in the next section.

We propose two types of templates: a multiple-record template for displaying multiple records that meet query search condition (see Fig.3(a)), and a single-record template for displaying only one record (see Fig.3(b)). Note that a single-record template is an optional feature which allows users to move to previous/next record that meets query search condition. System manager has to generate a basic SQL statement for each template. A basic SQL statement is a SQL statement which does not contain condition to select some records of a database table. In order to save time of investigating database tables and table fields, we recommend system manager to use Microsoft query which is a function of Microsoft Excel for generation a basic SQL statement. Figure 4(a) depicts an example of content of a basic SQL statement file used to query all data of database table *t_kokyaku*. This SQL statement is paired with template of Fig.3(a) or template of Fig.3(b). Figure 4(b) shows an example of a screen-shot of Microsoft query where all fields of database table *t_kokyaku* is selected.



(a) An example of a template which search condition is defined for querying records of table *t_kokyaku*.



(b) An example of a worksheet showing query result of the above template.

Figure 5: An example of query records of table *t_kokyaku* by template.

3.3 How to enable users defining search condition without the knowledge of SQL?

The main idea is that users are allowed to define additional search condition on multiple-record templates. As shown in Fig.5(a), a new search condition "*where (birthday >=1984/01/01 and point>=500) or kid >='C045'* " is defined on the template for querying records of table *t_kokyaku*. Note that search conditions defined on the same row (here range(B9:F9)) are joined by the logical AND operator. Search conditions of range(B9:F9) and range(B10:F10) are joined by the logical OR operator. SpreadDB adds this condition as where clause of the corresponding SQL statement shown in Fig.4(a). Figure 5(b) depicts an example of a worksheet showing the result of the query performed by the template of Fig.5(a).

4. TEMPLATE DESIGN PROCEDURE

Template design procedure consists of the following three steps.

Step 1: System manager uses Microsoft Query (see Fig.3(a)) to define a basic SQL statement for querying a database table or for querying the result of joining

database tables. Microsoft Query is a visual method of creating database queries using examples based on a text string, the name of a database table or a list of database tables. The Query by Example system converts the user input into a formal database query using SQL on the backend, allowing the user to perform powerful searches without having to explicitly compose them in SQL, and without even needing to know SQL. The result of this step is a SQL statement file whose file extension is .dqy. Fig.3(a) shows an example of content of a SQL query file for querying database table *t_kokyaku* which stores customer information.

Step 2: System manager uses Microsoft Excel to create a new multiple-record template from a given sample multiple-record template which is stored as an Excel worksheet. As shown in Fig.5(a), system manager enters a list of attribute names, which appear in the SQL statement file of Step (1), into cell B7 thru cell F7. Range(B8:F8) denotes table headers of corresponding attributes. For example, *kid* which denotes customer-id is defined at cell B7. Therefore the value of attribute *kid* of this first record that matches with search condition is outputted to cell B9. The value of attribute *kname* of the first record is outputted to cell C9. In case that record update function is enabled, system manager must define the first field of the table as the attribute which is the primary key of the underlying table so that SpreadDB can uniquely identify the record to be updated. She enters table headers under corresponding attribute names.

Step 3: System manager uses Microsoft Excel to create a new single-record template from a given sample single-record template. This template is activated by clicking "Display single record" button of multiple-record template of step 2. In order to have SpreadDB outputted attribute values to the right cells of the single-record template, system manager must enter table headers that match with those of multiple-record template. As shown in Fig 4(b), table header "顧客ID" is defined at cell C8 then the value of attribute *kid* is outputted at cell C9.

5. COUNTERMEASURE FOR UNAUTHORIZED DATABASE DATA RETRIEVAL AND MODIFICATION

We propose two level protections of unauthorized database data retrieval and modification in SpreadDB.

- The first level of protection is realized at RDBMS by assigning each user with a database user-id and by defining database operations that are authorized to user who login with the user-id. In order to decrease the need to managing user-id of user side, we propose to integrate computer user-id with database user-id. However, password requested when signing on user's computer should be different from the password requested when connecting database in order to decrease the risk of impersonation at user's computer.
- The second level of protection is realized at the file server which stores templates and SQL statement files of SpreadDB by utilizing authorization control system of the file server. This protection reduces the risk of accessing irrelevant templates and corresponding SQL statement files.

Based on the above protection mechanism, SpreadDB will retrieve user-id from operating system of user's computer and will use that user-id when downloading templates and correspond SQL statement files and when connecting external relational databases.

6. RELATED WORK

The most similar to our work reported here are the following. The paper [7] demonstrates that a spreadsheet can implement all data transformations definable in SQL, merely by utilizing spreadsheet formulas. That work provides a query compiler, which translates any given SQL query into a worksheet of the same semantics, including NULL values. Thereby, database operations become available to the users who do not want to migrate to a database The paper [8] proposes an extension of the set of spreadsheet functions by carefully designed database function, whereby the user can specify (and later execute) SQL queries in a spreadsheet-like style, one step at a time. Two papers [9, 10] describe a project, later named Query by Excel to extend SQL by spreadsheet inspired functionality, allowing the user to treat database tables as if they were located in a spreadsheet and define calculations over rows and columns by formulas resembling those found in spreadsheets. In the final paper [11] a spreadsheet interface is offered for specifying these calculations, which had to be specified in an SQL-like code in the earlier papers.

7. CONCLUSION AND FUTURE WORK

We have presented SpreadDB which is a spreadsheet-based user interface for typical users who don't know SQL in order to query and update data of external databases. SpreadDB allows system manager to design a template which users use to perform data query and data insertion/update/delete upon a table or a view of a RDB. Furthermore, a template can also be design to display the result of joining database tables. Users can define additional query condition at the template. We have proposed two level protections of unauthorized database

data retrieval and modification.
The following questions or issues seem worth exploring.

- Develop a methodology that allows users to create their own query template without requesting system manager.
- Extend SpreadDB to connect other models of databases like NoSQL, and XML.
- Some RDBMS, such as MySQL (community version), etc., don't provide small granularity of authorization control such as data protection at table level and record level. Can SpreadDB provide another data protection level to compensate this drawback?

REFERENCES

1. Margo Seltzer. **Beyond Relational Databases**, in *Queue 3, 3* (April 2005), p. 50-58. DOI=http://dx.doi.org/10.1145/ 1059791.1059807.
2. A. Eisenberg and J. Melton. 2000. **SQL standardization: the next steps**, in *SIGMOD Rec. 29*, 1, pp.63-67. DOI=http://dx.doi.org/10.1145/344788.344819, Mar 2000.
3. W. Fischer. ***Excel: QuickStart Guide - From Beginner to Expert (Excel, Microsoft Office)***. CreateSpace Independent Publishing Platform (May 7, 2016).
4. J.T. Roff. ***ADO: ActiveX Data Objects: Creating Data-Driven Solutions***. O'Reilly Media; 1 edition (Jun 30, 2001).
5. C.Wood. ***OLE DB and ODBC Developer's Guide 1st Edition***. Wiley; 1 edition (Sep 24, 1999).
6. M. Alexander, R. Kusleika. ***Excel 2016 Power Programming with VBA (Mr. Spreadsheet's Bookshelf) 1st Edition***. Wiley; 1 edition (Feb 8, 2016).
7. J. Sroka, A. Panasiuk, K. Stencel, J. Tyszkiewicz. **Translating Relational Queries into Spreadsheets**, *IEEE Transactions on Knowledge and Data Engineering*, Vol.7, Issue.8, pp. 2291-2303, 2015. https://doi.org/10.1109/TKDE.2015.2397440
8. B. Liu and H. V. Jagadish. **A spreadsheet algebra for a direct data manipulation query interface**, *Proceeding of ICDE'09*, pp.417–428, USA. https://doi.org/10.1109/ICDE.2009.34
9. A. Witkowski, S. Bellamkonda, T. Bozkaya, G. Dorman, N. Folkert, A. Gupta, L. Shen, and S. Subramanian. **Spreadsheets in RDBMS for OLAP**, *Proceeding of SIGMOD'03*, pp. 52–63. https://doi.org/10.1145/872757.872767
10. A. Witkowski, S. Bellamkonda, T. Bozkaya, N. Folkert, A. Gupta, L. Sheng, and S. Subramanian. **Business modeling using SQL spreadsheets**, *Proceeding of VLDB'03*, pp.1117-1120. https://doi.org/10.1016/B978-012722442-8/50119-1
11. A. Witkowski, S. Bellamkonda, T. Bozkaya, A. Naimat, L. Sheng, S. Subramanian, and A. Waingold. **Query by Excel**, *Proceeding of VLDB'05*, pp.1204–1215.