Analysis of Software Artifacts

Mini Project 1

RAD Team

Suh, Il-Seok

Jung, Heejoon

Table of Contents

1. Introduction

1.1 MSIT-SE Practicum Project Introduction

The School of Computer Science (SCS) has operated the Master of Software Engineering (MSE) and the Master of Science in Information Technology – Software Engineering (MSIT-SE) programs for several years, and produced hundreds of graduates. At present, the MSE Program has three major databases which are related to the Admission database, Student Progress database, and Alumni database. The primary client, Jane Miller (SCS/ISRI), who is a MSE program manager, is generally satisfied with the Admission database, and Student Progress database. However, a major issue is the Alumni database. The client needs more fields to store Alumni information such as company, salary, and address. Second, the three databases are not cleanly integrated so that the client needs to do additional work when migrating data between the three databases. Third, the Alumni database should have security and shared-access privileges, which are granted by the primary client. Therefore, the project is to develop a database system which satisfies these three requirements.

1.2 Purpose of Evaluation

This team needs to use test data for the practicum project, because loading database with useful test data can be a difficult and time consuming task. To test functional requirements and quality attributes of the database such as performance, the database needs many records which are realistic. Therefore, the purpose of this evaluation is to make meaningful test data as soon as quickly. In addition, we expect that we can estimate how can use this tool for our practicum project by evaluating this tool.

1.3 Evaluation Approach

To evaluate DataFactory, we classified key functions, and made evaluation criteria regarding our practicum project. After making evaluation criteria, we designed a simple Alumni database with Microsoft Access. After execution, we evaluate the generated test data according to the evaluation criteria.

2. What is DataFactory?

2.1 Description and Features

DataFactory is a test data generation tool developed by Quest Software Inc. This tool allows the database testers to easily populate test databases with millions of rows of meaningful, syntactically correct test data with ease of user interface. DataFactory reads a database schema and displays database tables and fields. Then, users can enter several input condition for the test data. Following is the key functions which DataFactory provides [1]:

- Generate realistic data using pre-defined test data tables that includes thousands of names, zip codes, area codes, cities, states, and other database data
- Make the same script to be used on various databases
- Compatible with not only Oracle, DB2, MS SQL Server, and Sybase databases but also databases using an ODBC
- Keep data integrity between database tables
- Transfer outputs to the databases directly or to a text file [1]

Following is the steps to generate test data:

- Load a schema from database
- Display database tables and fields
- Produce meaningful test data
- Write the test data to output files or save into the database

2.2 Supporting Platforms

DataFactory 5.5 can be run on the following operating systems:

- Windows 2000
- Windows XP
- Windows 2003

2.3 Supporting Platforms

- Oracle: 7, 8, 8i, 9i, 9.2 and 10g
- SQL Server: 7 and 2000
- Sybase: 11.5 and 12
- DB2 UDB: 7 and 8

2.4 DataFactory User Interface

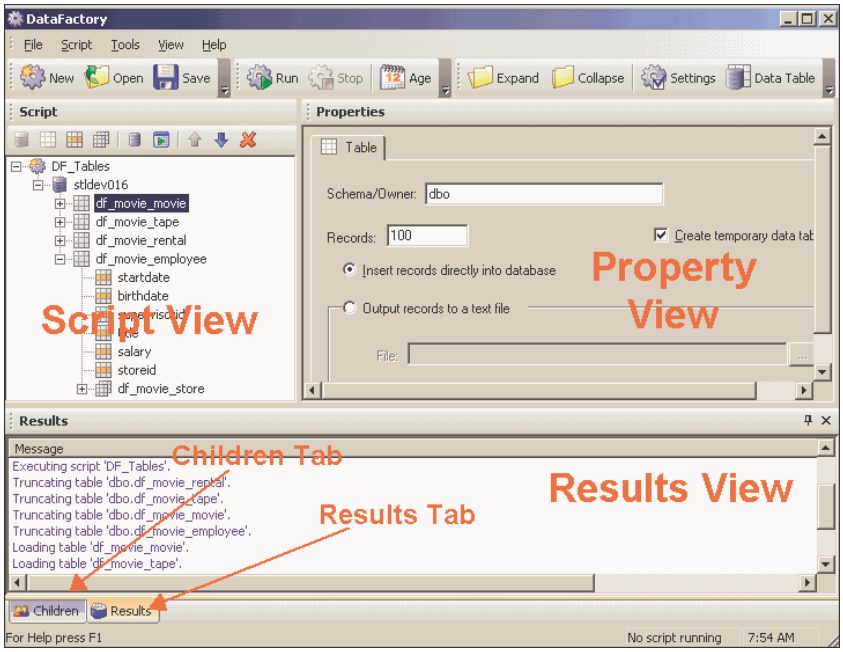

Figure 1 DataFactory User Interface [2]

- Scripts View: This view displays the details of the database that users are populating.
- Properties View: This view displays individual data structure characteristics of the node users have selected in the script view.
- Results View: This view displays messages detailing what took place during the running of a script.
- Children View: This view displays the properties of the children of the node selected in the scripts view.

3. Evaluation Criteria

To evaluate whether DataFactory is a valuable tool as a database test data generator, we made five evaluation criteria: validity of generated data, compatibility, usability, performance, and documentation. According to these criteria, we analyzed the result and evaluated the tool.

3.1 Validity of Generated Data

3.1.1 We evaluate whether the DataFactory generates test data that maintain referential integrity between database tables. To use generated data from the tool, it is necessary to verify that the data meet all the referential integrities. It is the most important and hardest evaluation criterion that validates the tool is useful.

3.1.2 We evaluate that generated data is flawless. When users set a rule or a range of generating values, data generated by the tool should satisfy those conditions to be used as test data.

3.1.3 We evaluate whether the tool generates realistic data or not. In other words, we evaluate generated data are meaningful, that is, it seems like existing person's name, address, phone number, or anything in real.

3.2 Compatibility

3.2.1 We evaluate whether the DataFactory is compatible with database management systems (DBMS) that are commonly used and databases using open database connectivity (ODBC). The 'compatibility' means database tables are well read (loaded) from databases and generated test data are well written (saved) to databases.

3.3 Usability

3.3.1 We evaluate whether the tool provides good user interface, which means users don't feel difficulties to use the tool. Also, we check whether the tool supports a

command line interface or a graphical user interface (GUI).

3.3.2 We evaluate whether the tool is easy to use. In other words, we test whether users can easily perform the tool's main functionalities.

3.3.3 We evaluate whether there is any restriction in the tool that causes inconvenience for users.

3.4 Performance

3.4.1 We evaluate how much time will be taken to generate all the test data that satisfy a certain condition that user set.

3.4.2 We evaluate whether there is any factor that affects the performance of the tool. If there exists, we will find correlation the factor and the performance.

3.5 Documentation

3.5.1 We will evaluate that the tool has well written documentations such as an installation guide, a user manual, or an error list.

4. Experiments

4.1 Experimental Setup

To make test data, the DataFactory tool should be connected with a certain Database which has several tables. Our team made a Alumni Database which has three tables and each table has a relationship. Following figure represents the tables, attributes, primary key, foreign key, and relationships.

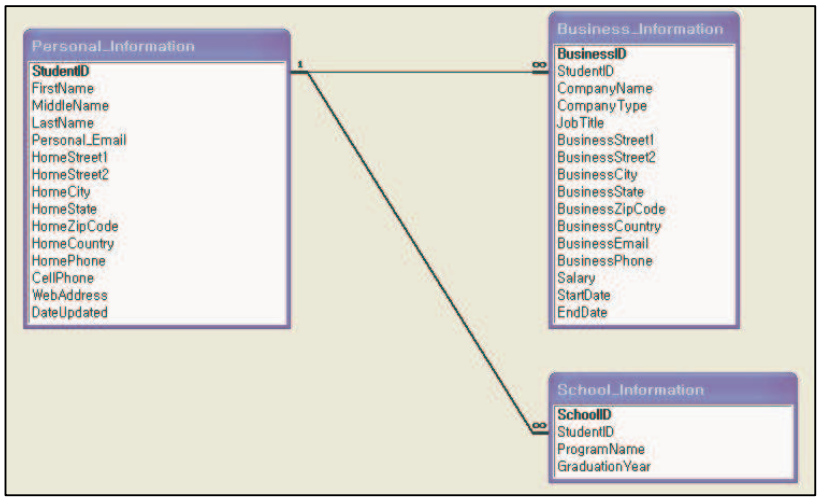

Figure 2 Alumni Database

Alumni Database has three tables, Personal_Information table, Business_Information table, and School_Information table. Each table has primary key, StudentID, BusinessID, and SchoolID respectively. Also, StudentID attribute in the Business_Information table and the School_Information table is a foreign key which refers to the primary key in the Personal_Information table.

To load the Database into the DataFactory, the Database should be registered in the ODBC Data Source. After registering, following steps are required in the DataFactory:

- Select the connection method such as DB2, Oracle, ODBC, SQL Server, or Sybase
- Enter Connection parameters for the data type such as data source, id, and password
- Select the tables which will be populated into DataFactory

- Select conditions per each table and attribute

4.1.1 Experiment by using Microsoft Access

DataFactory does not support the direct connection with Microsoft Access, so we have to make a connection with ODBC driver. After make the connection with Microsoft Access, we made conditions per each table and attribute to identify whether the generated data are realistic and maintain the data integrity such as primary key and foreign key. Following figure represents the conditions to the Personal_Information table and its attributes.

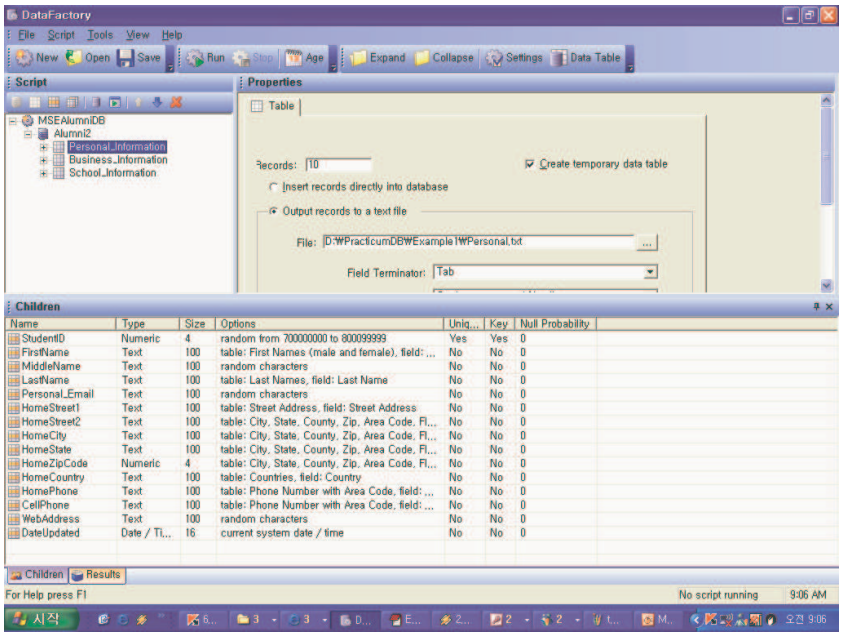

Figure 3 Conditions to the Personal_Information table and its attributes

Following tables represent the conditions:

Type	Pre-Condition
Primary Key	It should be unique and random number or sequential numbers within predefined boundary.
Foreign Key	It should refer to the primary key which exists in another table.
Text	DataFactory have pre-defined data tables that contain first names, last names, cities, states, countries, companies, zip codes, and phone numbers. If an attribute is not related to these

	data, then text attributes should be selected on random option and number of characters.
Date	The boundary of Date should be defined before running this tool.
Number	Sequential or random numbers should be selected.

In addition, the generated test data cannot be entered into the Database table directly, because the DataFactory does not support the function, so we made output files to store the generated test data per each table.

After making the conditions, we executed the tool, and it showed a success message and elapsed time.

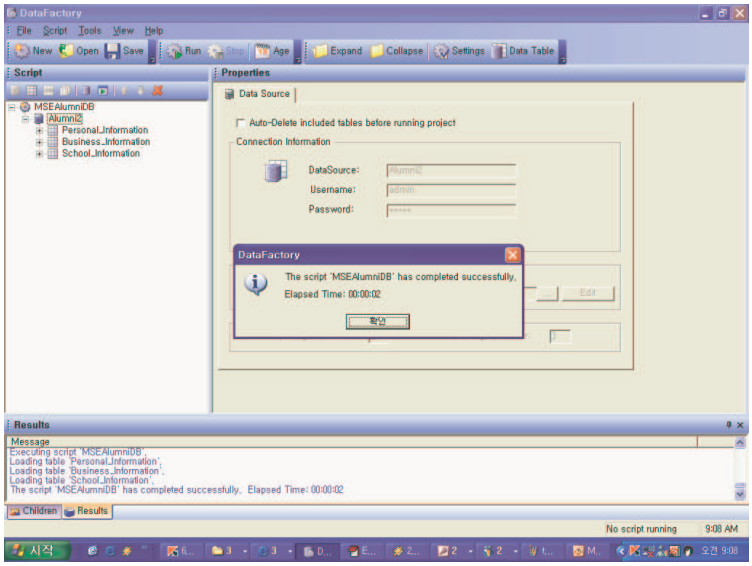

Figure 4 Result Message

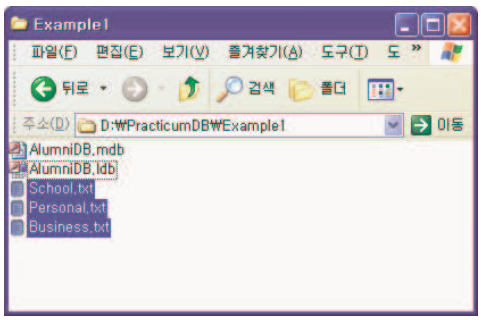

Figure 5 Generated Files

Also, it made three text files in which the generated test data are stored. Following figures represent the generated test data in each text file.

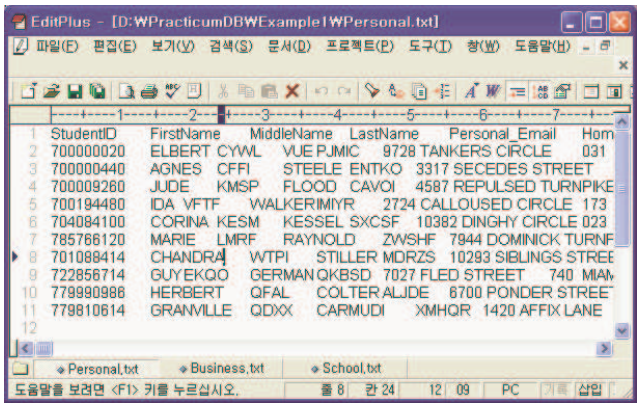

Figure 6 Generated Test Data for Personal_Information Table

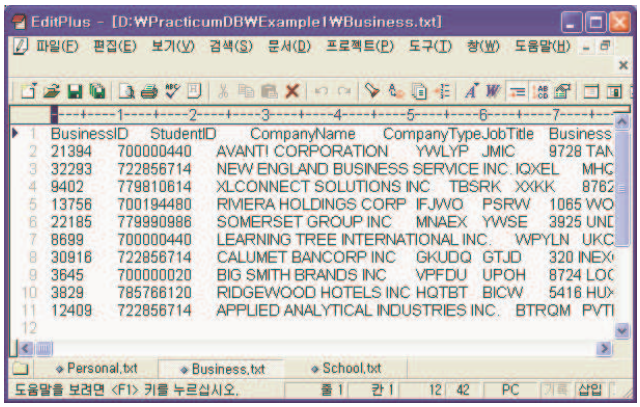

Figure 7 Generated Test Data for Business_Information Table

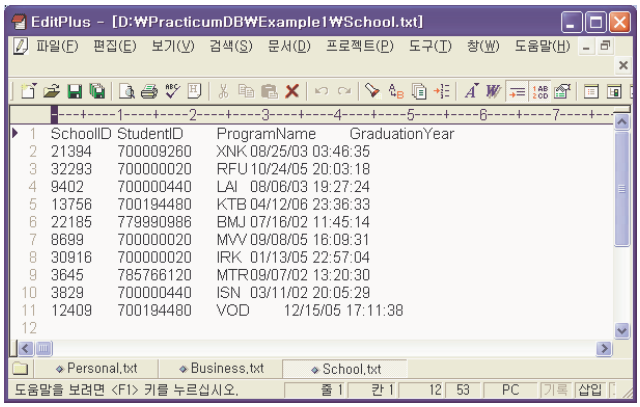

Figure 8 Generated Test Data for School_Information Table

4.2 Qualitative Data

This database has 3 tables and 35 fields. And we generated 10 records, 100 records, 1000 records, 5000 records and 10000 records.

Following table show the elapsed time per test:

Record Number	10	100	1000	5000	10000
Elapsed Time	0~1 seconds	0~1 seconds	2 seconds	16 seconds	61 seconds

4.3 Quantitative Data

DataFactory generated certain number of test data for each fields. As the pre-condition, it generated data. The primary key field, StudentID at Personal_Information table, maintained unique and random number, and the foreign key filed, StudentID at Business_Information and School_Information table, referred to the primary key which exists in Personal_Information table. Also, the pre-defined data such as name, address, and company name are generated with realistic data, but the data which are not pre-defined generated non-realistic data. It just combined characters randomly. In addition, the address data in the generated data does not provide consistency with other address fields. For example, there are four fields related to address such as street1, street2, city, and state. The street1 and street2 should be located in the city, and the city should be located in the state, but the generated data does not maintain this consistency.

5. Evaluation

Following evaluations are performed based on the evaluation criteria described in the Section 3.

5.1 Validity of Generated Data

5.1.1 Advantages

One of the most important evaluation criteria that we made was the validity of test data generated by the DataFactory. We want to have test data for our database tables, which is a part of the MSIT-SE practicum project, so the generated data should be realistic and should not have a defect. During our evaluation we couldn't find any defects from the generated data itself.

First of all, the DataFactory supports a function that checks referential integrity between database tables. In other words, the users of the tool do not need to care about detailed relationships between database tables. When people make test data for a database, it is virtually impossible not to care about referential integrities. Actually a database has not just one table but two or more tables and they have complicated relationships. It is hard to manually create test cases that are free from primary keys and foreign keys. However, during our evaluation the tool always maintained referential integrities between database tables. Also, we confirmed that the tool maintains the referential integrity among multiple foreign keys.

In addition, the tool supports various options to generate test data. It is possible to make unique values and generate sequential numbers automatically. These functions are very useful when users want to generate primary keys for database tables. Generally primary keys consist of a combination of numerical values and should be unique. We confirmed that the tool could generate flawless primary keys by using those functions.

Moreover, we could set a range of values that would be generated.

5.1.2 Disadvantages

Even though the generated test data itself is flawless, the tool still has several problems.

The DataFactory has its own database that stores a lot of realistic data. For example, the database has first names like John, David, Mary, and Mark, etc. and last names like Smith, Bush, Lee, and Klein, etc. The primary mechanism that the DataFactory generates realistic test data is to randomly retrieve data already stored in the database and combine them. For instance, when generating a name the tool randomly selects a first name such as Mark and a last name such as Klein from the tool's database and combines them – Mark Klein. It seems like the tool creates very realistic data, but it has a problem. The database of the tool has limited data. The number of a set of test cases, which are generated by this mechanism, would be restricted by the number of data stored in the tool's database. Therefore, it is necessary that sufficient data should be always stored in the tool's database.

Another problem is a relationship between generated data. The Datafactory merely generates test data and does not check relationship among generated data. In other words, the tool only generates data and puts them into fields in a table, and does care about their dependencies. The most problems that we found were addresses. When creating a home address it randomly retrieves a street address, a city, a state, a zip code, and a country from the tool's database without considering their relationship. For example, the tool can combine Craig St., Santa Barbara, Illinois, and 15213. In this case, it is possible that Craig St. does not exist in the Santa Barbara, and there is no city named Santa Barbara the Illinois state. Furthermore, the generated zip code, 15213, is not matched with the city. This problem seriously lowers the reality of the generated test data.

5.2 Compatibility

5.2.1 Advantages

The primary advantage of the DataFactory is that it enables users to access various popular databases management systems (DBMS) and open database connectivity (ODBC) compliant databases. It is possible to directly load database tables from Oracle, MS SQL Server, DB2, and Sysbase and reversely save generated test data into those databases. The tool supports direct access to those databases, so the only thing that users have to do is following several steps to make a connection with the tool and a database. For other databases such as FileMaker that aren't mentioned above, it is possible to access them by using an ODBC connection. It means various databases, which are

commonly used, are compatible with the DataFactoy, so the tool can read and write the tables in the databases. This advantage makes the DataFactory stronger than other tools. Most of other database test generator tools do not support direct access to database tables, that is, if over 1,000 or 10,000 test data are generated, it is virtually impossible to manually type all the data into databases. Considering that it is hard and time-consuming work to save generated test data into databases, compatibility is a remarkable strength of the DataFactory.

5.2.2 Disadvantages

Although the DataFactory has an advantage that it enables to access various kinds of database, it is still incomplete.

Especially the DataFactory is partly, not completely, compatible with databases using the ODBC connection. Before making a connection, it is necessary to install additional ODBC drivers. According to our test, the tool sometimes failed to even make a connection with a database using ODBC. In fact, most problems related to ODBC connections occurred when the tool tried to save generated test data into databases. This problem makes the tool lose its unique advantage. Obviously the generated test data are useless things if they are not in the original databases. Also, it does not support some major database management systems such as MySQL. It means the tool cannot make test data for databases using MySQL.

In addition, once tables are loaded from databases, relationships among tables that are existed in the original databases are no longer maintained. In other words, when loading tables from a database all of relationships maintained in the original database are broken up. To recover those relationships, additional work is required. If the original database contains more than one table, recovering relationships would be a troublesome task to users.

5.3 Usability

5.3.1 Advantages

Basically the DataFactory provides a simple and plain graphical user interface (GUI). The main frame consists of three major frames - Script frame, Properties frame, and

Results frame -, menus, and some icons. The tool is not decorated with fancy designs, but the application is understandable and scripts and messages are readable. For basic functions it seems like that users can intuitively get an idea how to use. Users may not feel inconvenient to use the tool.

In addition, the tool provides a wizard to create data tables. The “Create Data Table Wizard” makes data tables by a step-by-step instruction, so what users have to do is just following some steps and selecting what they want.

Also, the tool provides “Children View” that shows which values would be set to each field. For users, it is easy to recognize which field attributes are set up and what values they have.

5.3.2 Disadvantages

Even though the DataFactory is easy to use and provides a simple GUI, there are several problems that lower the usability of the tool.

First of all, the DataFactory does not have its own viewer to display output results. In the tool, there are only two ways to display outputs: saving into text files and saving into the original database. The latter one that saves generated test data into original database tables does not make a problem, but the former one causes a problem. When saving results into text files, the readability of the output is not good. Especially when the output contains a null value the result seems like tangled, and users may feel difficulties to read output data.

In addition, it is impossible to generate test cases without a database. Strictly speaking, it is hard to define a standalone application, because users cannot do anything only having the tool itself. It always needs an external database. In other words, the tool cannot even generate a data without a database. This problem seriously restricts the usability of the tool that generates test data.

Moreover, the tool needs an extra work to perform some functions. In other words, it is not a completely automatic test data generator. For example, users have to specify additional items to verify a referential integrity between tables. Also, users should check some items to recover relationships among tables. These problems make users feel

inconvenience to use the tool.

5.4 Performance

We measured the performance of the DataFactory by elapsed time. Generally the tool showed a satisfactory performance. We tested a database that we actually created for the MSIT-SE practicum project. The database had three tables and totally 35 fields. We generated 1,000, 5,000, and 10,000 test data for each table - total 3,000, 15,000, and 30,000 test data respectively. The elapsed times of the test were 2 seconds, 16 seconds, and 67 seconds. Even though the number of maximal test data that the trial version can generate for a table is restricted within 10,000, it is enough to test a general performance of the tool.

Also, we discovered that there is a correlation between performance and the number of tables. The performance of the tool was lowered if the number of tables in a database was increased when the number of generated test data is fixed. It means the elapsed time that the tool generates 10,000 test data for a table is faster than the elapsed time that the tool generates respective 5,000 test data for two tables (total 10,000 data). Obviously, also, there is a correlation between performance and the number of fields. The performance of the tool was lowered if the number of fields in a database was increased when the number of tables is fixed. It is because the number of generating test data would be increased.

5.5 Documentation

5.5.1 Advantages

The only strong point related to the documentation that the DataFactory has is providing a simple tutorials to users. The tutorial gives users general directions for use. The tutorial consists of step-by-step instructions that train users to use basic functions of the tool. The tutorial will be helpful to novice users.

5.5.2 Disadvantages

The DataFactory has little documentation. Even though the DataFactory is commercial

software and we used the trial version, documentation is definitely insufficient. There are very limited resources available.

In fact, there is no official documentation. We could not find any documents related to the tool on the Web site.

The tool does not provide an installation guide. Although software installation process is simple and easy, some users may have to install additional ODBC drivers to make a connection between the tool and the database. However, we failed to find a document that describes which ODBC drivers are required and how to install those drivers.

Also, there is no user manual. There is no documentation that describes how to use the tool. Fortunately the tool provides intuitive GUI, so we could familiarize ourselves with the basic functions of the tool. However, it was hard to know detailed system functionalities and how to use those. The best try we could do was to click all of the buttons and menus.

Furthermore, the tool does not provide error lists or exception lists. During the evaluation of the tool, we found some errors occurred. However, we couldn't find the detailed descriptions about those errors. Clearly, it is hard to find a cause of an error and fix it without a detailed description.

These problems - the lack of documentations – may cause a serious problem that users keep away from the tool.

6. Future Improvements

It is necessary to enhance validity of generated data. The present DataFactory is able to generate some meaningful test data. However, the number of generating data is restricted by the number of data stored in the tool's database. It is necessary to increase the number of sets by diversify candidates. One available method is that the tool uses databases on the Web instead of its local database. Another suggestion is to periodically update the local database. Also, the tool should care relations between generated data. To increase validity of data, the tool should check relations of fields such as street address and city, city and state, and state and zip code.

Also, the tool should support more direct access to DBMS. As shown in the Section 5.2.2, the tool is not fully compatible with databases using ODBC connection. There were several problems discovered during the evaluation. The tool is unstable when loading and saving using ODBC connection. Therefore, it is desirable that the tool supports direct access to databases instead of using ODBC.

In addition, the tool should reinforce its graphical user interface with a result viewer. The tool needs to include a result viewer to improve the usability. Also, the tool should provide a function that generates test data without a database for users, who don't have a database.

Finally, the tool should improve its documentation. It is necessary to create an installation guide, a user manual, and an error list.

7. Conclusion

The team evaluated DataFactory, the database test data generator tool, to obtain meaningful test data for the practicum project.

According to the experiment and evaluation results, the team verified that the tool is generally useful to create realistic test data. It showed good performance, that is, it does not take long time to generate over ten thousand of test cases.

However, it has some minor problems to be a market leading tool. It is necessary to improve its validity by updating data, enhance compatibility with major database management systems, and create some documentation such as a user manual.

The team has a plan to generate test cases for the practicum project by using the tool. We expect the tool will be improved and show better result.

8. References

[1] http://www.quest.com/datafactory/features_benefits.aspx

[2] DataFactory Help File