References

[1] AAAI (2011), *The Conceptual Frameworks and Organized Vocabularies forAI Programs*,http://www.aaai.org/AITopics/pmwiki/pmwiki.php/AITopics /ontologies, Retrieved on 02 November, 2010

[2] Chandrasekaran, B. and Jorn R. Josephson, V. Richard Benjamins. (1999), *What Are Ontologies, and Why Do We Need Them?* IEEE Intelligent Systems. 14 (1): pp. 20 - 26.

[3] DMSoftTechnologies (2011), *Smart Database Conversion Tools*, http://www.dbconvert.com, Retrieved on 10 September, 2010

[4] DMSoft Technologies (2011), *Database Synchronization*, http://dbconvert.com/synchronization.php, Retrieved on 25 September, 2010

[5] DMSoft Technologies (2011), *Data Filtering*, http://dbconvert.com/filtering.php, Retrieved on 28 September, 2010

[6] DMSoft Technologies (2011), *Optimal data types mapping with DB Convert tools*, http://dbconvert.com/field.php, Retrieved on 06 October, 2010

[7] eSurveysPro (2011), *Online survey software. Free web survey tool*, http://www.esurveyspro.com/, Retrieved on 25 November, 2010

[8] Erik T. Ray (2003), Learning XML 2nd Edition, O'Reilly Media, Inc., pp.78-103.

[9] Ileanadu (2010), *.NETFramework*, http://en.wikipedia.org/wiki/ .NET_Framework, Retrieved on 15 September, 2010

[10] Jan L. Harrington (2010), SQL Clearly Explained, 3rd Edition, Elsevier, pp.77-130

[11] Ken Bailey (2010), *Ontology (information science)*, http://en.wikipedia.org/wiki/Ontology_(information_science) , Retrieved on 18 October, 2010

[12] NarendraBute (2004), *Important Features of .Net*, http://www.codeproject.com/KB/vb/Features_of_Net.aspx, Retrieved on 15 October, 2010

[13] Pro Data Doctor Pvt Ltd (2010), *Need of database conversions solution?* http://www.db-convert.net, Retrieved on 10 September, 2010

[14] Stanford Center for Biomedical Informatics Research (2011), *The Protégé Ontology Editor and Knowledge Acquisition System*, http://protege.stanford.edu/, Retrieved on 20 November, 2010

[15] Stanford Center for Biomedical Informatics Research (2011), *What is protégé?*, http://protege.stanford.edu/overview/, Retrieved on 22 November, 2010

[16] Tiwebb Ltd (2007), *What is .NET?* ,http://www.devtopics.com/what-is-net/, Retrieved on 10 October, 2010

[17] Tom Gruber (1992), *What is an Ontology?*http://www-ksl.stanford.edu/kst/what-is-an-ontology.html, Retrieved on 20 October, 2010

[18] University of Maryland, Baltimore County (UMBC) (2002), *Ontology Tutorial*, http://ilit.umbc.edu/Ontology-tutorial-content.pdf,Retrieved on 18 September, 2010

Appendix A – User Guide for OnexDB Engine

A.1 Embedding the OnexDB- Multi-Dialect SQL Conversion Engine with C#/VB .NET applications

The OnexDB Engine multi-dialect SQL conversion engine can be embedded in your C# / VB .NET applications. The reason to embed OnexDB Engine into the application is to make the application independent of the relational database used for persisting the application data and at the same time having the ability to utilize the power of SQL and the underlying relational model inside the application business logic.

Prerequisite: Microsoft .NET Framework redistributable package should have been installed in the system before using OnexDB.

1. Directly use the OnexDB Engine and parse and convert the given SQL from one dialect to another. This option has the benefit of applications having the control of when an SQL needs to be parsed and converted. Also there is no performance penalty for this option as the applications need to convert the SQL only once and pass the converted SQL to the database specific driver layer. Before using the OnexDB Engine you need to have reference to the OnexDB.dll assembly in the bin directory in the application folder.

2. Use the database independent OnexDB ODBC wrapper which acts as a proxy between the application and the database specific ODBC driver.

This section will shows about how to use the embedded option from a C#/VB.NET application.

A.2 Using OnexDB API .NET Directly in C# Application

In this phrase explains the step by step procedure to create a simple c# application directly using the OnexDB to convert SQL from one dialect to another before passing the SQL to the underlying database through a database driver or by other means.

```
/* Import the required OnexDB */
      using OnexDBAPI;
/* End Import */

/* Variable Declarations */
 String GivenSQL = "select top 10 * from employee";
 String OracleSQL = null;
 String MSSQLServerSQL = null;
 String MySQL = null;
/* End Variable Declarations */

/* Instantiate OnexDB with the Given SQL and convert the SQL to Oracle */
 OnexDbSqlApi onexDbSqlApi = new OnexDbSqlApi(GivenSQL);
 OracleSQL = onexDbSqlApi.Convert(OnexDbSqlApi.ORACLE);

/* Set the GivenSQL to API .NET and convert the SQL to MSSQLServer */
 onexDbSqlApi.SQLString = GivenSQL;
 MSSQLServerSQL=
onexDbSqlApi.Convert(OnexDbSqlApi.MSSQLSERVER);

/* Set the GivenSQL to API .NET and convert the SQL to MySQL */
 onexDbSqlApi.SQLString = GivenSQL;
MySQL = onexDbSqlApi.Convert(OnexDbSqlApi. MYSQL);

/* Print the converted SQL */
 System.Console.Out.WriteLine("\nOracle SQL : " + OracleSQL);
```

```
System.Console.Out.WriteLine("MSSQLServer SQL : " + MSSQLServerSQL);
System.Console.Out.WriteLine("MySQL SQL : " + MySQL);
/* End */
```

- **Main Steps need to follow**

1. **Importing the OnexDB API namespaces/classes**

   You need to import OnexDB name spaces from your application

   - OnexDBAPI

   OnexDB API is the main API to be used in applications to parse and convert SQL from one dialect to another.

2. **Instantiating the OnexDbSqlAPI**

   While instantiating the OnexDbSqlAPI, you can pass the SQL String that needs to be converted to a different dialect, as an argument.

3. **Converting the SQL to a different Dialect**

   Use the API call OnexDbSqlAPI.Convert (int) to convert the SQL from one dialect to another. The Convert () function takes an integer as an argument. OnexDbSqlAPIdefines static public fields for each dialect to be passed as argument to the convert () function. The static variable names in OnexDbSqlAPI are given below.

0	OnexDbSqlAPI.GIVENSQL	No conversion. Return the Original SQL
1	OnexDbSqlAPI.ORACLE	Convert to Oracle SQL dialect
2	OnexDbSqlAPI.MSSQLSERVER	Convert to MSSQLServer SQL dialect
3	OnexDbSqlAPI.MYSQL	Convert to MySQL SQL dialect

A.3 Using OnexDB .NET API Directly in VB.NET Application

In this phrase explains the step by step procedure to create a simple vb.net application directly using the OnexDB to convert SQL from one dialect to another before passing the SQL to the underlying database through a database driver or by other means.

```
'Import the required OnexDB API classes
    Import OnexDBAPI
 ' End Import

'Variable Declarations
    Dim GivenSQL As String
    Dim OracleSQL As String
    Dim MSSQLServerSQL As String
    Dim MySQL As String
  ' End Variable Declarations

    GivenSQL = "select top 10 * from employee"

  'Instantiate OnexDB API with the Given SQL and convert the SQL to Oracle
    OnexDbSqlAPI onexDbSqlApi= new OnexDbSqlAPI (GivenSQL)
    OracleSQL = onexDbSqlApi.Convert(OnexDbSqlAPI.ORACLE)

  ' Set the GivenSQL to OnexDB API  and convert the SQL to MSSQLServer
    onexDbSqlApi.SQLString = GivenSQL
    MSSQLServerSQL =
    onexDbSqlApi.Convert(OnexDbSqlAPI.MSSQLSERVER)

  ' Set the GivenSQL to OnexDB API  and convert the SQL to MySQL
    onexDbSqlApi.SQLString = GivenSQL
    MySQL = onexDbSqlApi.Convert(OnexDbSqlApi.MYSQL)
```

```
'Print the converted SQL
  Console.WriteLine("\nOracle SQL : " + OracleSQL)
  Console.WriteLine("MSSQLServer SQL : " + MSSQLServerSQL)
  Console.WriteLine("MySQL SQL : " + MySQL)
'End
```

- **Main Steps need to follow**

1. **Importing the OnexDB API namespaces/classes**

   You need to import.NET name spaces from your application

   - OnexDBAPI

   OnexDBAPI is the main API to be used in applications to parse and convert SQL from one dialect to another.

2. **Instantiating the OnexDbSqlAPI**

   While instantiating the OnexDbSqlAPI, you can pass the SQL String that needs to be converted to a different dialect, as an argument.

3. **Converting the SQL to a different Dialect**

   Use the API call OnexDbSqlAPI. Convert (int) to convert the SQL from one dialect to another. The Convert () function takes an integer as an argument. OnexDbSqlAPIdefines static public fields for each dialect to be passed as argument to the convert () function. The static variable names in OnexDbSqlAPI are given below.

0	OnexDbSqlAPI.GIVENSQL	No conversion. Return the Original SQL
1	OnexDbSqlAPI.ORACLE	Convert to Oracle SQL dialect
2	OnexDbSqlAPI.MSSQLSERVER	Convert to MSSQLServer SQL dialect
3	OnexDbSqlAPI.MYSQL	Convert to MySQL SQL dialect

Appendix B – Protégé XML File

B.1 Protégé XML Mapping File

In the Figure B.1and B.2 shows the XML mapping file. It contains set of SQL syntaxes of different DBMS's.

```
<?xml version="1.0"?>
<!DOCTYPE Ontology>
- <Ontology>
<!-- Generated by the OWL API (version 3.1.0.1602) http://owlapi.sourceforge.net -->
    <Prefix IRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#" name="rdf"/>
    <Prefix IRI="http://www.w3.org/2002/07/owl#" name=""/>
    <Prefix IRI="http://www.w3.org/2001/XMLSchema#" name="xsd"/>
    <Prefix IRI="http://www.w3.org/2000/01/rdf-schema#" name="rdfs"/>
    <Prefix IRI="http://www.w3.org/2002/07/owl#" name="owl"/>
  - <Declaration>
      <Class IRI="#mssql_all"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#mssql_distinct"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#mssql_into"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#mssql_like"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#mysql_like"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#mysql_all"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#oracle_and"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#oracle_as"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#oracle_isnull"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#oracle_left_join"/>
    </Declaration>
  - <Declaration>
      <Class IRI="#oracle_like"/>
    </Declaration>
  - <SubClassOf>
      <Class IRI="#mysql_constraint"/>
      <Class IRI="#constraint"/>
    </SubClassOf>
  - <SubClassOf>
      <Class IRI="#mysql_delete"/>
      <Class IRI="#delete"/>
    </SubClassOf>
  - <SubClassOf>
      <Class IRI="#oracle_all"/>
      <Class IRI="#all"/>
    </SubClassOf>
  - <SubClassOf>
      <Class IRI="#oracle_and"/>
      <Class IRI="#and"/>
    </SubClassOf>
  - <SubClassOf>
      <Class IRI="#mssql_create"/>
      <Class IRI="#create"/>
    </SubClassOf>
  - <SubClassOf>
      <Class IRI="#mssql_database"/>
      <Class IRI="#database"/>
    </SubClassOf>
```


Figure B.1: Detail Protégé XML File

```
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#mssql_all</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">all</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#mssql_and</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">and</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#mysql_all</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">all</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#mysql_and</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">and</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#oracle_all</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">all</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#oracle_and</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">and</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#oracle_as</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">as</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#oracle_between</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">between</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#oracle_constraint</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">constraint</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#oracle_delete</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">delete</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#oracle_inner_join</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">,</Literal>
  </AnnotationAssertion>
- <AnnotationAssertion>
    <AnnotationProperty IRI="http://purl.org/dc/elements/1.1/description"/>
    <IRI>#oracle_insert</IRI>
    <Literal datatypeIRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#PlainLiteral">insert</Literal>
  </AnnotationAssertion>
</Ontology>
```


Figure B.2: Detail Protégé XML File

Appendix C – Test cases and Test Result

In Table C.1 shows the detail description of the test cases and test result.

Test Case ID	Test Data	Expected Results	Pass / Fail
Case 1	Import OnexDBAPI library on c#.net program	Successfully imported.	Pass
Case 2	Import OnexDBAPI library on vb.net program	Successfully imported.	Pass
Case 3	Creating an object on OnexDbSqlApi using c#.net	The object of OnexDbSqlApi should be created.	Pass
Case 4	Creating an object on OnexDbSqlApi using vb.net	The object of OnexDbSqlApi should be created.	Pass
Case 5	Call onexDbSqlApi Convert function by passing correct SQL query using c#.net	The Convert function should available for accept correct SQL queries.	Pass
Case 6	Call onexDbSqlApiConvert function by passing correct SQL query using vb.net	The Convert function should available for accept correct SQL queries.	Pass
Case 7	Call onexDbSqlApi Convert function by passing incorrect SQL query using c#.net	Return exceptions efficiently.	Pass
Case 8	Call onexDbSqlApi Convert function by passing incorrect SQL query using vb.net	Return exceptions efficiently.	Pass
Case 9	Assign Converted SQL query to a variable using c#.net.	Successfully assigned converted query.	Pass
Case 10	Assign Converted SQL query to a variable using vb.net.	Successfully assigned converted query.	Pass

Case 11	Call onexDbSqlApi Convert function by passing correct SQL query without creating object of onexDbSqlApi using c#.net.	The Convert function should available for accept correct SQL queries.	Pass
Case 12	Call onexDbSqlApi Convert function by passing correct SQL query without creating object of onexDbSqlApi using vb.net.	The Convert function should available for accept correct SQL queries.	Pass
Case 13	Call onexDbSqlApi Convert function by passing incorrect SQL query without creating object of onexDbSqlApi using c#.net.	Return exceptions efficiently.	Pass
Case 14	Call onexDbSqlApi Convert function by passing incorrect SQL query without creating object of onexDbSqlApi using vb.net.	Return exceptions efficiently.	Pass
Case 15	Call onexDbSqlApi Convert function without passing DBMS name of SQL Syntax using c#.net.	Return exceptions efficiently.	Pass
Case 16	Call onexDbSqlApi Convert function without passing DBMS name of SQL Syntax using vb.net.	Return exceptions efficiently.	Pass
Case 17	Call onexDbSqlApi Convert function without passing DBMS name of SQL Syntax should convert using c#.net.	Return exceptions efficiently.	Pass

Case 18	Call onexDbSqlApi Convert function without passing DBMS name of SQL Syntax should convert using vb.net.	Return exceptions efficiently.	Pass
Case 19	Call onexDbSqlApi Convert function without passing incorrect DBMS name of SQL Syntax using c#.net.	Return exceptions efficiently.	Pass
Case 20	Call onexDbSqlApi Convert function without passing incorrect DBMS name of SQL Syntax using vb.net.	Return exceptions efficiently.	Pass
Case 21	Convert valid SQL using OnexDB Console	Return converted SQL query.	Pass
Case 22	Convert invalid SQL using OnexDB Console	Return exceptions efficiently.	Pass
Case23	Clear text fields using clear button.	All the text fields should be cleared.	Pass
Case 24	Load user guide with menu command.	User guide opened with menu selection.	pass
Case 25	Exit from OnexDB console after selecting close menu.	OnexDB console has closed after selecting close menu.	Pass

Table C.1: Test cases and Test Result

Appendix D – Installation Guide for OnexDB Console

D.1 Installing OnexDB Console

1. Double-click on OnexDBAPI Setup.exe to start the installation.

2. Welcome: The Installer prompts you with information of the product to be installed in your machine. Shows in Figure D.1. Click Next to proceed.

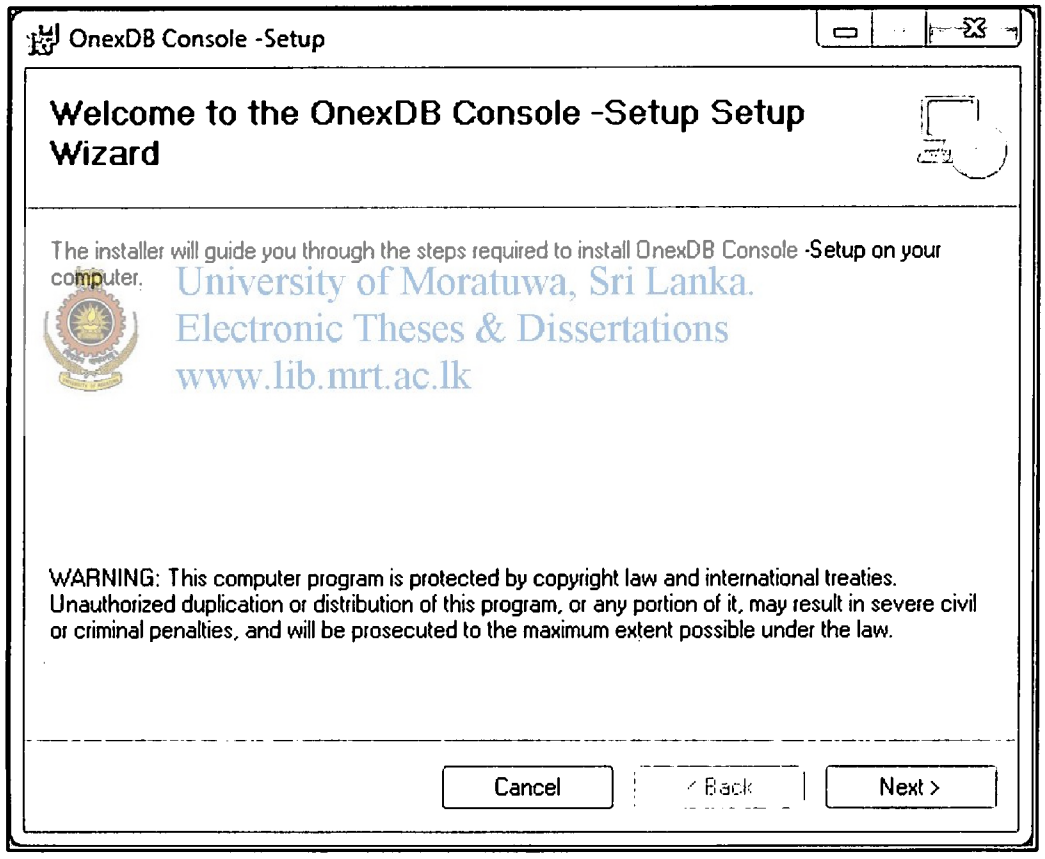

Figure D.1: OnexDB Console Setup Step 1

3. Choose the Destination Location: You can install the product in the default location (C:\Program Files\OnexDB Console -Setup\) or you can browse to specify the location of product installation. Shows the Figure D.2.

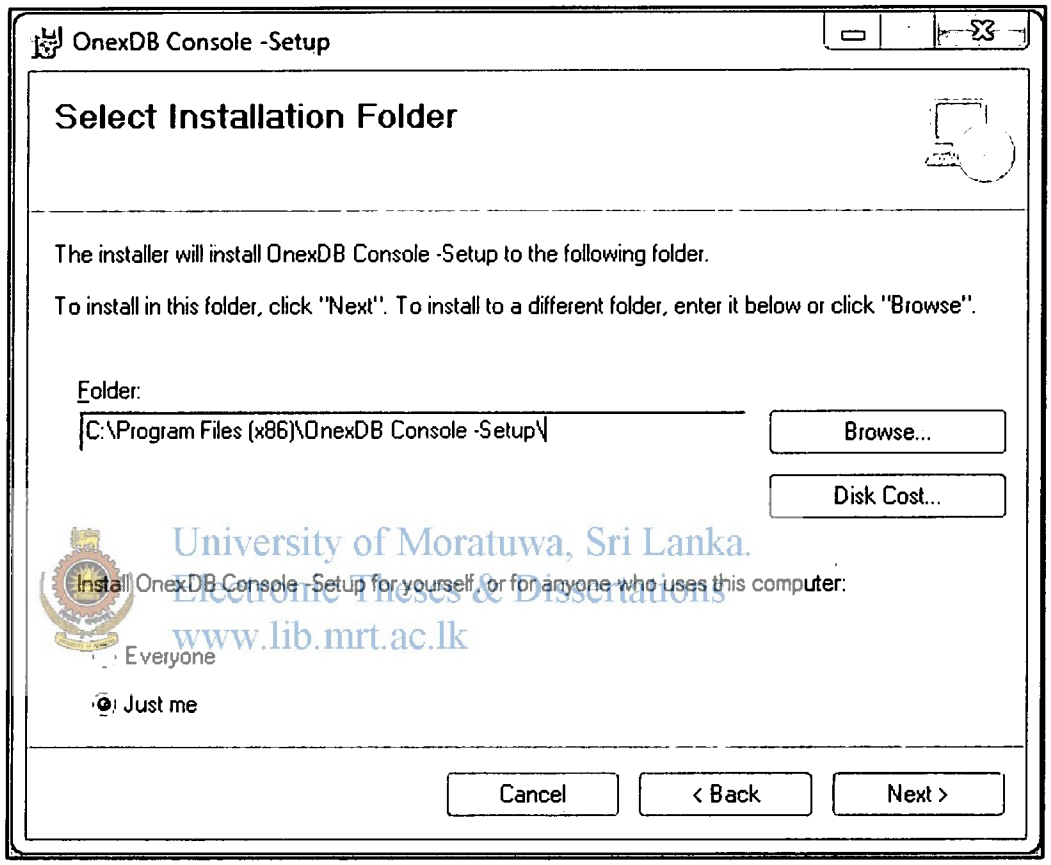

Figure D.2: OnexDB Console Setup Step 2

4. The Installer is ready to install the OnexDB Console to your local machine. You can click Back to make changes or click next to proceed with the installation to your local machine.Step 3 shows the Figure D.3.

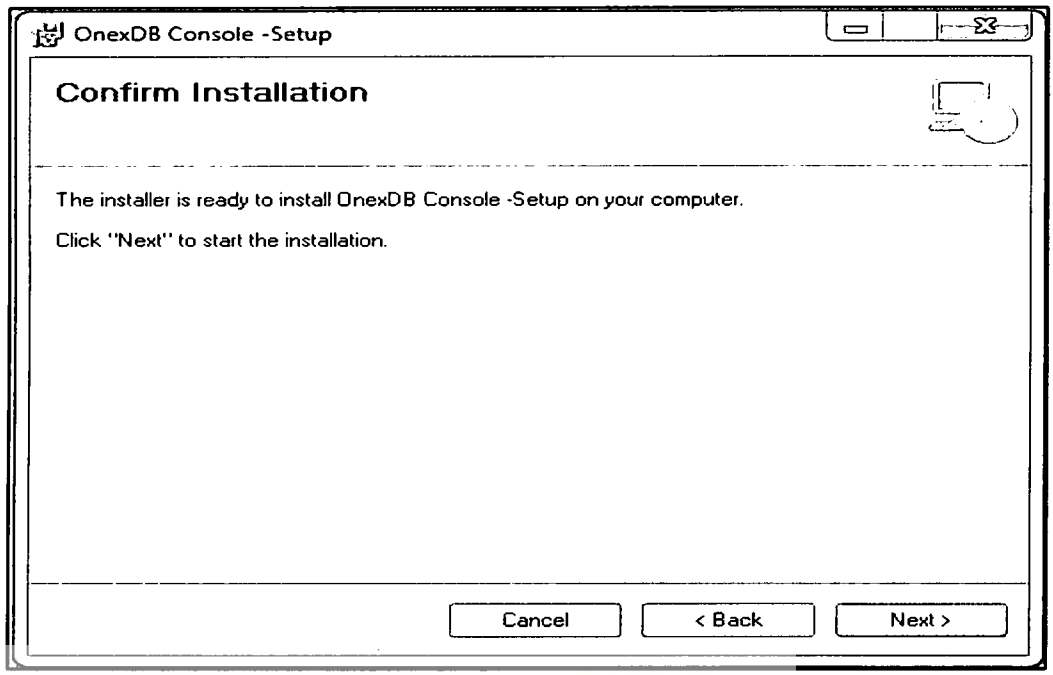

Figure D.3: OnexDB Console Setup Step 3

5. OnexDB Console – setup is being installed. Shows on Figure D.4

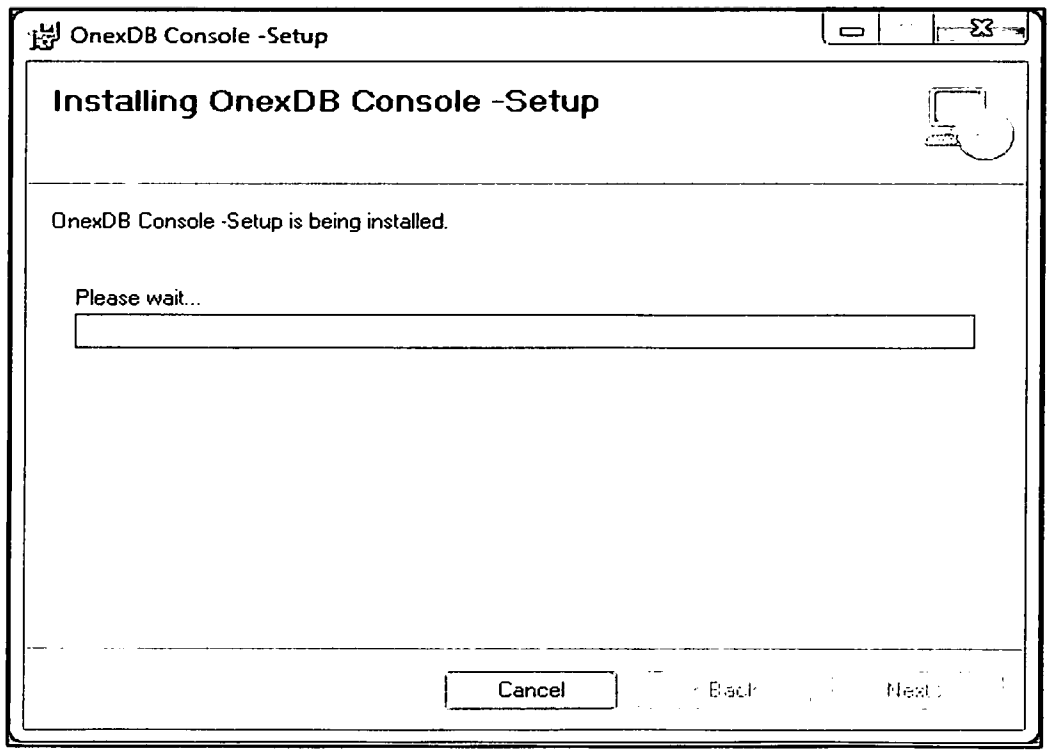

Figure D.4: OnexDB Console Setup Step 4

6. Install Shield Wizard Complete: A message appears stating that the installation was successful.It shows Figure D.5.

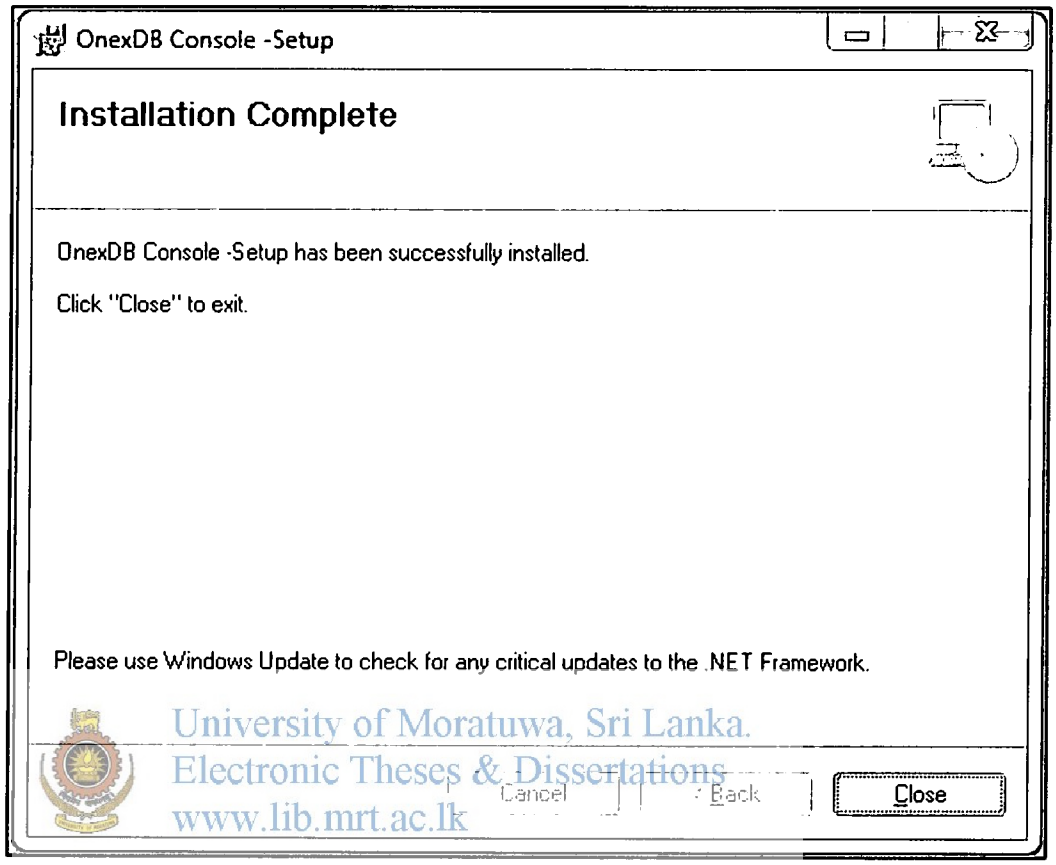

Figure D.5: OnexDB Console Setup Step 5

Appendix E – User Guide for OnexDB Console

E.1 Getting Started with OnexDB Console

You can start OnexDB Console in anyone of the following way:

- Click on the OnexDB Console link icon (You must have checked the option of having a quick link in the desktop in the final screen during installation)
- You can invoke the product from windows menu Start - > Programs
- Double click on the *OnexDBConsole.exe* file under *<InstallDir>/OnexDB/Console/* folder.
- Double click on the *run OnexDBConsole.exe* file under *<InstallDir>/OnexDB/Console/bin* folder.

Getting Familiar with the GUI

OnexDB Console spots a simple and effective UI. Figure E.1 shows the product snapshot given below briefs the various features and options available.

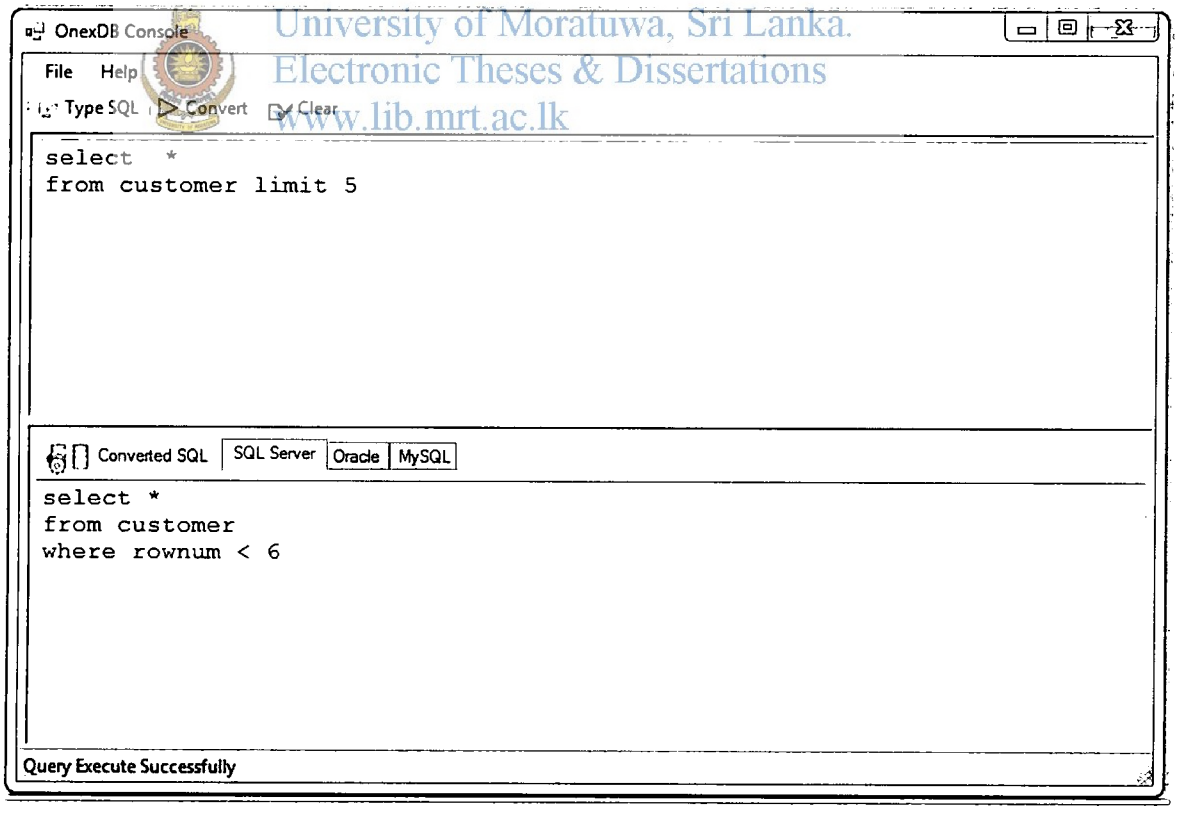

Figure E.1: OnexDB Console

OnexDB Console GUI is segmented in **2 parts** which are illustrated below:

- **Type SQL Area**: You need to type the SQL that needs to be converted. Once typed, you can use the Convert option (in the menu or toolbar) to convert the SQL into different database SQL dialects. The converted SQL will be displayed in the "Converted SQL" area. You could also load SQL script files for conversion.

- **Converted SQL Area**: All the converted SQLs will be displayed on the respective database Tab. For example clicking on Oracle tab will show the Oracle equivalent of the typed (or input) SQL.

Use the OnexDB Console GUI to type in a database specific SQL and convert the SQL into another relational database's SQL dialect. The converted SQL can be tested for accuracy from within the OnexDB Console tool itself. The OnexDB Console helps users who want to migrate their application from one database to another. It also would be immensely helpful to Database Administrators who are working with multiple databases and not familiar with all the database specific SQL dialects. The OnexDB Console application is also a good way to learn different SQL dialects.

E.2 Converting/Migrating SQLs

Select the corresponding SQL statement that has to be converted and choose any one of the following ways:

- Invoke *Convert* menu and choose the appropriate database menu item **(Or)**

- Invoke *Convert* tool bar icon (This will convert the SQL to the default target database selected)

The query will automatically be converted to the chosen database syntax and the converted SQLs will be displayed on the *Converted SQL* text area.

Appendix F – Web based Questionnaire used for evaluating the system

F.1 OnexDB Evaluation Questionnaires

Figure F.1, F.2, F.3, F.4 and F.5 shows the list of questionnaires which has given to selected users.

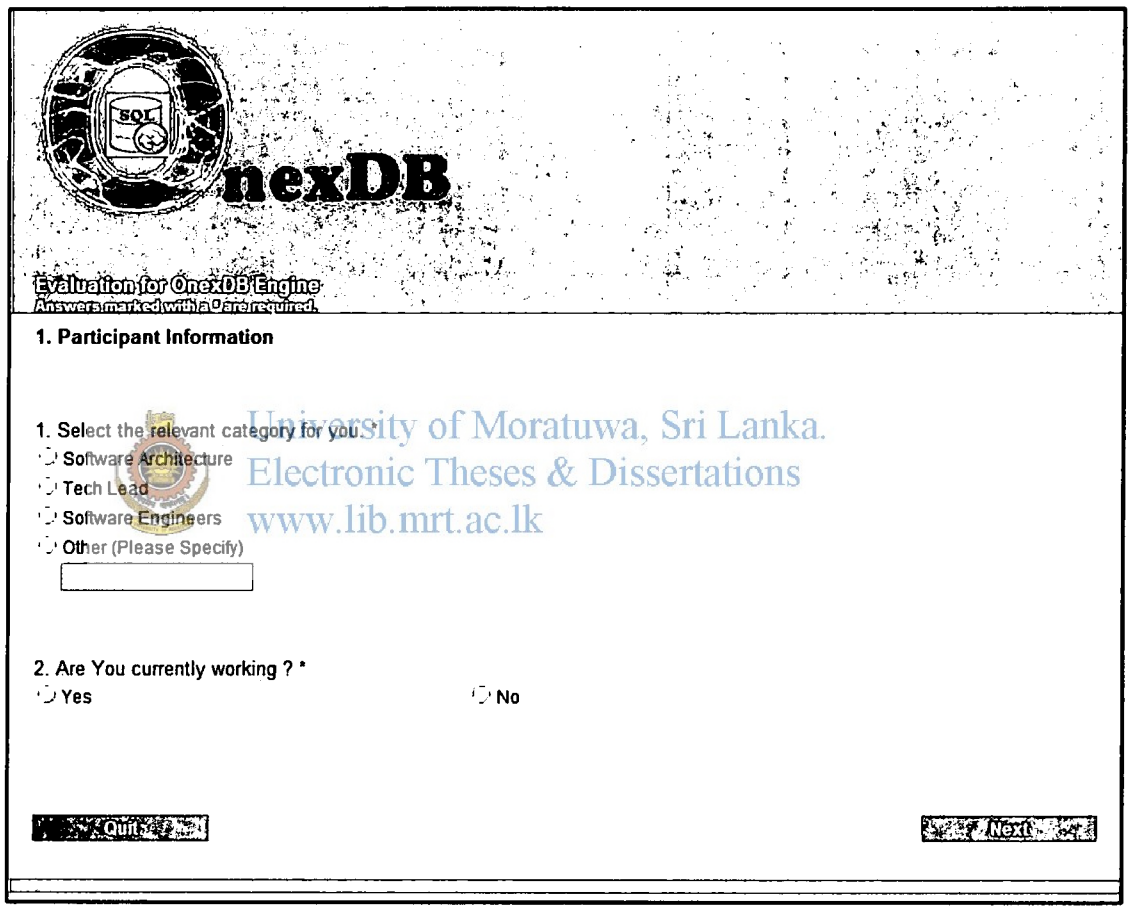

Figure F.1: Survey Questionnaire page 1

Evaluation for OnexDB Engine

Answers marked with a * are required.

2. Quality Attributes

3. Ease of Installation *
- Poor
- Below Average
- Average
- Good
- Excellent

4. Response Time *
- Poor
- Below Average
- Average
- Good
- Excellent

5. Accuracy of Result *
- Poor
- Below Average
- Average
- Good
- Excellent

Figure F.2: Survey Questionnaire page 2

6. Exception Handling *
- Poor
- Below Average
- Average
- Good
- Excellent

7. User Friendliness *
- Poor
- Below Average
- Average
- Good
- Excellent

8. Support for other language (c#.net, vb.net) *
- Poor
- Below Average
- Average
- Good
- Excellent

Figure F.3: Survey Questionnaire page 3

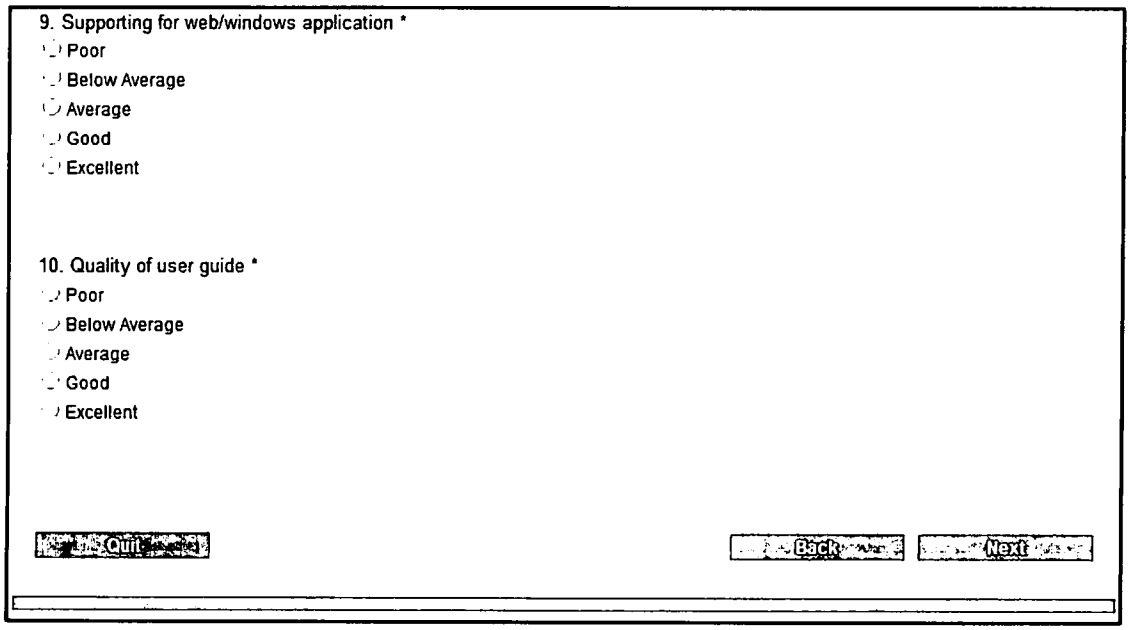

Figure F.4: Survey Questionnaire page 4

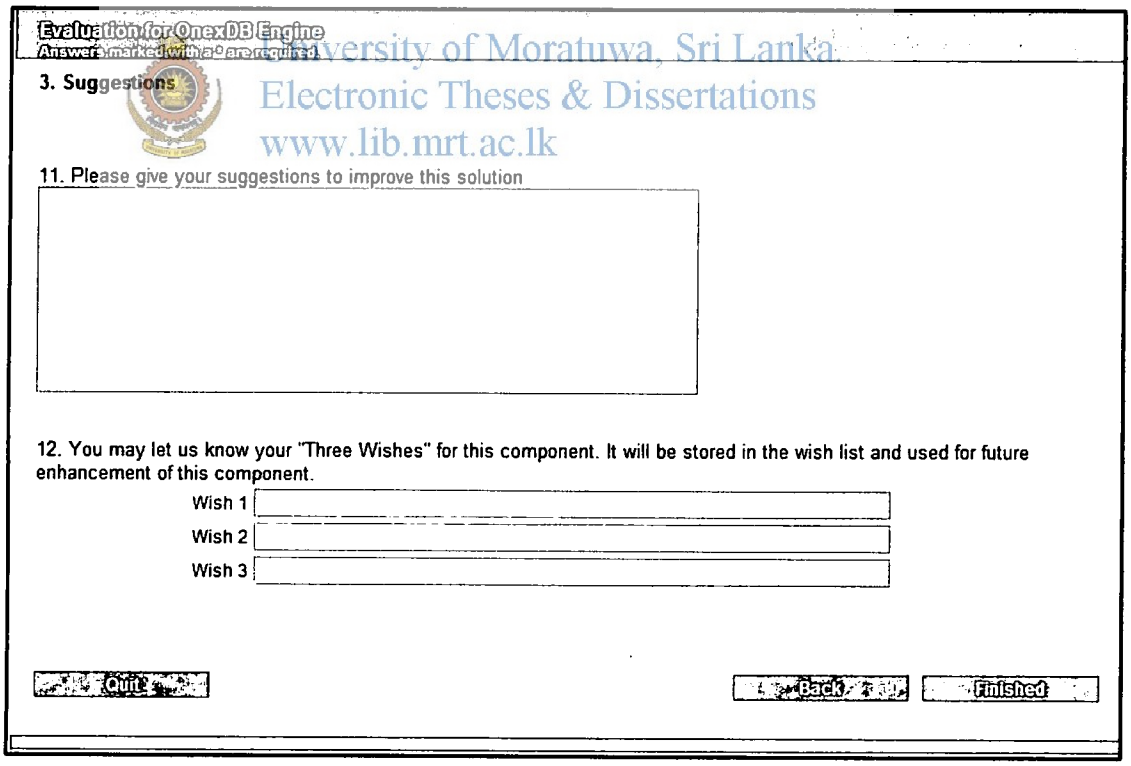

Figure F.5: Survey Questionnaire page 5