COMP 3311 Database Management Systems Spring 2015

Lab 7. A brief introduction to the ODBC interface

Objectives of the Lab

- You have seen how to connect to the Oracle server through the SQLPlus client.
- This lab will teach you doing the same connection using another interface, the ODBC interface:
- After this lab, you will be able to
  - connect to the Oracle server of COMP3311 through the ODBC interface using the C-language,
  - issue SQL queries using the ODBC interface.

In case you need to set up the ODBC environment for Visual Studio, you may refer to the appendix 1 at the end of this lab.

Brief introduction to ODBC 1

- ODBC stands for Open DataBase Connectivity.
- It provides a standardized database accessing interface that is independent of the actual DBMS being accessed.
- It supports high-level programming languages like C, C++, C#, VB, PHP etc…

Brief introduction to ODBC 2

- The Key components of the ODBC interface are:
  - Driver manager: that loads the proper DBMS driver on the behalf of an application
  - Driver: that processes the ODBC function calls and submit SQL requests to the specified data source
  - Data source: the DBMS

Brief introduction to ODBC 3

Application program
Driver Manager : loads the (Oracle) driver for the corresponding DBMS
DBMS driver (of Oracle in our case)
Data communication stack (i.e. TCP/IP)

Network

Data communication stack (i.e. TCP/IP)
DBMS driver (of Oracle in our case)
DBMS
Operating System

Connecting to a data source 1

- To connect to a the Oracle through the ODBC interface, one needs to:
  - Include the proper headers (<sql.h>, <sqlext.h>) to the C program,
  - Initialize ODBC environment,
  - Allocate a connection handle,
  - Connect to the data source corresponds to the Oracle server.

Connecting to a data source 2

- To initialize ODBC environment, one needs to
  - Declare a variable of the type HENV,
  - Call the SQLAllocEnv() function and pass it the address of the variable:

```
HENV   henv;

/* Allocate environment handle */
SQLAllocEnv( &henv);
```

Connecting to a data source 3

- To allocate a connection handle one needs to
  - Declare a variable of the type HDBC
  - Call the SQLAllocConnect() function and pass it the address of the variable. The ODBC driver would allocate memory for storing the connection information:

```
HDBC   hdbc;

/* Allocate connection handle */
SQLAllocConnect(henv, &hdbc);
```

Connecting to a data source 4

- To connect to the Oracle data source, one needs to call the SQLConnect() function and provide it with:
  - Data source name (in our case comp3311.cse.ust.hk)
  - Oracle account (your comp3311stuxxx account)
  - Oracle account password (your password)

Connecting to a data source 5

- The following is an example call to the function SQLConnect()

```
SQLConnectA(hdbc, (SQLCHAR*)
"comp3311.cse.ust.hk", SQL_NTS, (SQLCHAR*)
"comp3311stu120", SQL_NTS, (SQLCHAR*) "123456",
SQL_NTS);
```

  - "comp3311.cse.ust.hk" is the data source name (refer to appendix 1 on how setting up a data source)
  - "comp3311stu120" is an Oracle account name
  - "123456" is the password of the account
  - SQLCHAR is a char type of SQL
  - SQL_NTS denotes that the previous argument in the function is a Null Terminated String

Performing Query 1

- To perform the SQL operations using the ODBC interface, you need to:
  - allocate a statement handle
  - submit an SQL statement for execution
  - retrieve the results

Performing Query 2

- To allocate a statement handle, you need to:
  - declare a variable of the type HSTMT,
  - call SQLAllocStmt() function and pass the connection handle (hdbc) and the address of the HSTMT-typed variable.

```
HSTMT hstmt;

/*allocate the statement handle*/
SQLAllocStmt(hdbc, &hstmt);
```

Performing Query 3

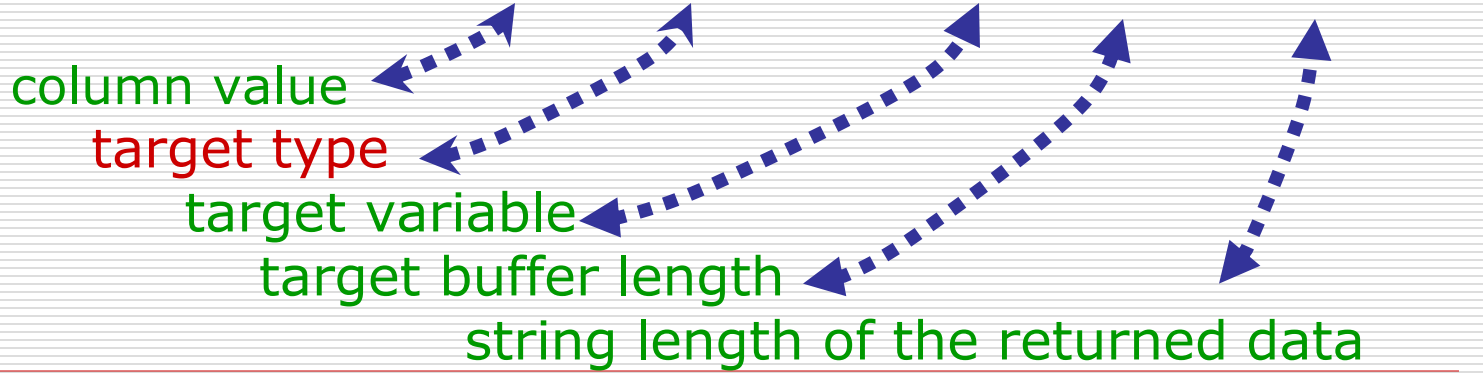
- To submit an SQL statement for direct execution (prepared statement will be covered next time), you need to:
  - Call SQLExecDirect() function and pass it with the SQL statement as well as the statement handle (hstmt)

```
SQLExecDirectA(hstmt, (SQLCHAR *) "SELECT
department_ID FROM departments",SQL_NTS);
```

Performing Query 4

- To retrieve the results you need to:
  - Bind variables to the attributes of the query results using SQLBindCol() function:

```
/* bind the char string variable "deptid" to get the
result (column 1) from the query*/

SQLBindCol(hstmt,1,SQL_C_CHAR,deptid,50,&deptid_n);
```



Performing Query 5

- Fetch the results using the SQLFetch() function:

```
/* fetch the results into the variable "deptid" and then
    display*/
     while (TRUE){
            retcode=SQLFetch(hstmt);
            if (retcode==SQL_SUCCESS || retcode \
                  ==SQL_SUCCESS_WITH_INFO)
                                    {printf("%s\n",deptid);}
            else break;
     }
```

Terminating the program

- To gracefully terminate, you need to
  - Disconnect from the Oracle data source
  - Free the environment variable and the handle:

```
SQLFreeStmt(hstmt,SQL_CLOSE);
SQLDisconnect(hdbc);
SQLFreeConnect(hdbc);
SQLFreeEnv(henv);
```

- A script file for building the Database is available at:
  http://course.cse.ust.hk/comp3311/labs/lab7.sql
  Remember to issue **"commit;"** command at the SQLPLUS client, otherwise your ODBC program will not see the records inserted.

- The complete code for connecting to the Oracle server of COMP3311 is available at:
  http://course.cse.ust.hk/comp3311/labs/odbc1.cpp

Putting everything together 1

```
#include "stdafx.h"
#include <windows.h>
#include <sql.h>
#include <sqlext.h>
#include <sqltypes.h>

int _tmain(int argc, _TCHAR* argv[])
{

 HENV   henv;
 HDBC   hdbc;
 HSTMT  hstmt;
 RETCODE retcode;
 SQLINTEGER sqlcode,deptid_n;
 SQLSMALLINT len;
 SQLCHAR deptid[50];

 /* Allocate environment handle */
 retcode = SQLAllocEnv( &henv);

 /* Allocate connection handle */
 retcode = SQLAllocConnect(henv, &hdbc);

 /* Connect to the service */
 retcode = SQLConnectA(hdbc, (SQLCHAR*) "comp3311.cse.ust.hk", SQL_NTS, (SQLCHAR*) "comp3311stu120",
         SQL_NTS, (SQLCHAR*) “123456", SQL_NTS);
 if (retcode == SQL_SUCCESS || retcode == SQL_SUCCESS_WITH_INFO){
        printf ("Connected to Oracle.\n");}
```

Putting everything together 2

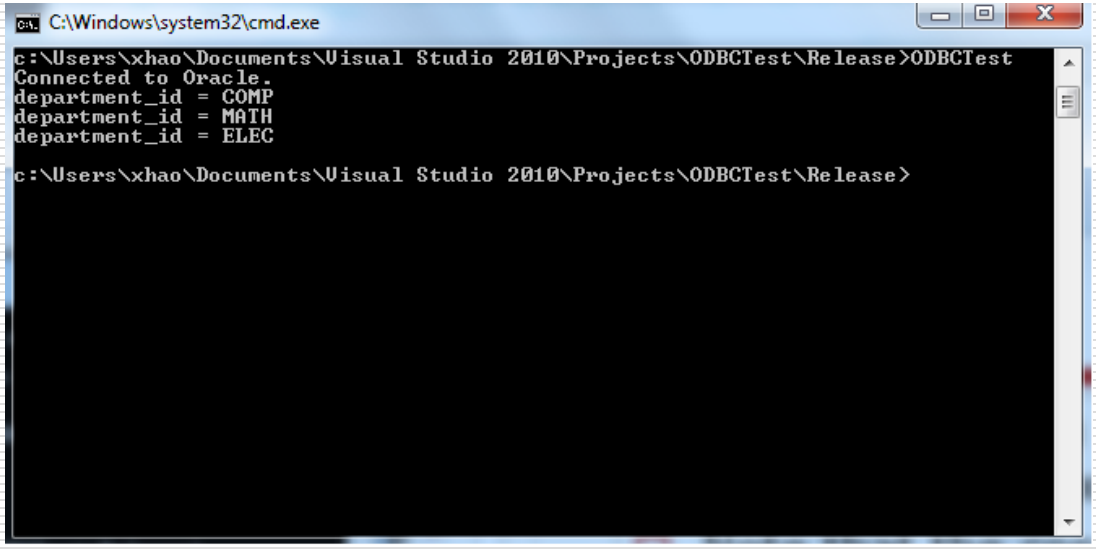
```
/* execute a SELECT statement*/
 SQLAllocStmt(hdbc, &hstmt);
 SQLExecDirectA(hstmt, (SQLCHAR *) "select department_ID from departments",SQL_NTS);

 /* bind the char string variable "deptid" to get the result from the query*/
 SQLBindCol(hstmt,1,SQL_C_CHAR,deptid,50,&deptid_n);

/* fetch the results into the variable "deptid" and the display*/
 while (TRUE){
        retcode=SQLFetch(hstmt);
        if (retcode==SQL_SUCCESS || retcode ==SQL_SUCCESS_WITH_INFO)
              {printf("department_id = %s\n",deptid);}
        else break;
 }

 /* free resources */
 SQLFreeStmt(hstmt,SQL_CLOSE);
 SQLDisconnect(hdbc);
 SQLFreeConnect(hdbc);
 SQLFreeEnv(henv);

 return 0;

}
```

Putting everything together 3

- You **must** set up the ODBC data source before your program will run successfully, please refer to appendix 1 on steps to set up the ODBC data source.
- You may refer to appendix 2 on compiling and running your first ODBC program using the Visual Studio 2010 package.

The result

- Compile and run the program according to the steps of appendix 2.
- Note that the connection to the Oracle server is successful, and results are returned.



Conclusions

- We covered the following topics in this lab:
  - connect to the Oracle server of COMP3311 through the ODBC interface,
  - issue simple a SQL query using the ODBC interface.

Appendix 1: Steps to set up ODBC environment for visual studio 1

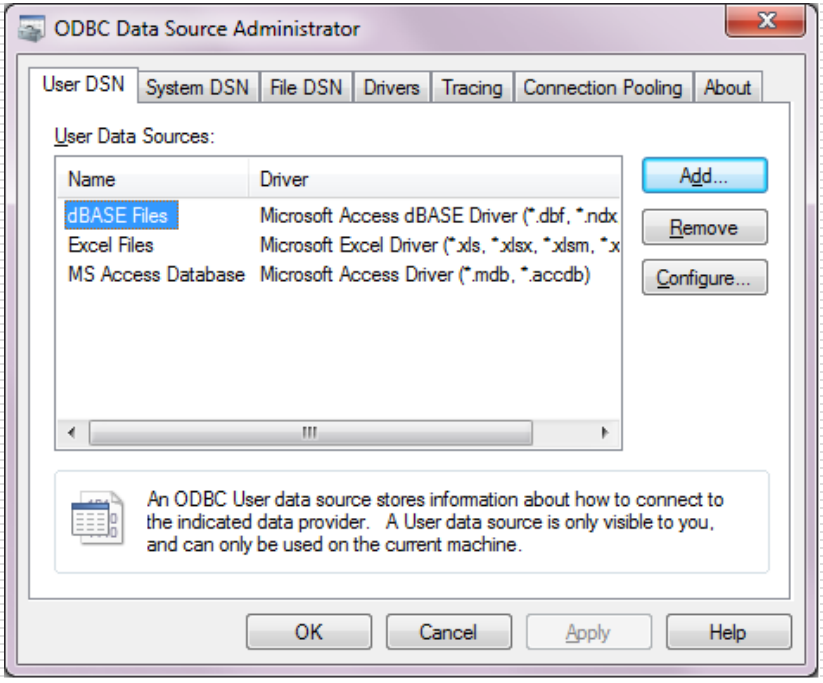
- In the CS Lab, we have already installed the ODBC driver at:
  - c:\windows\syswow64\odbcad32.exe
- At your home, you can download and install the Oracle ODBC driver from:
  - http://www.oracle.com/technetwork/topics/dotnet/utilsoft-086879.html

Appendix 1: Steps to set up ODBC environment for visual studio 2

- 1. You can setup the Oracle ODBC source in MS windows (the ***most important*** steps!)
  - Click the “start” button , then key in “c:\windows\syswow64\odbcad32.exe”

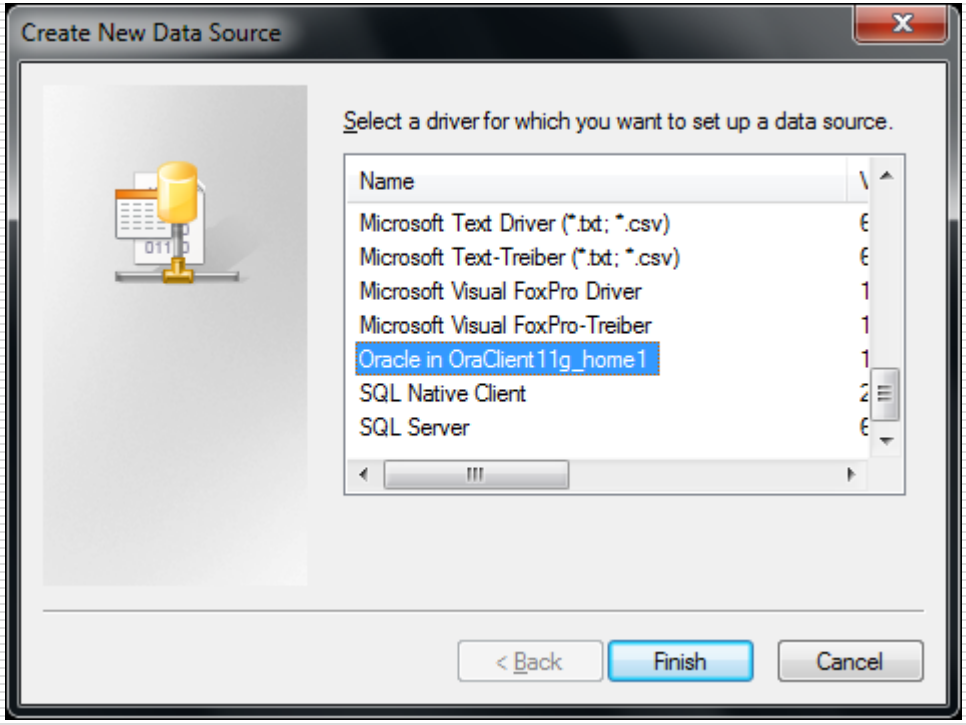
Appendix 1: Steps to set up ODBC environment for visual studio 3

- 2. You should see the below window and click the “Add” button



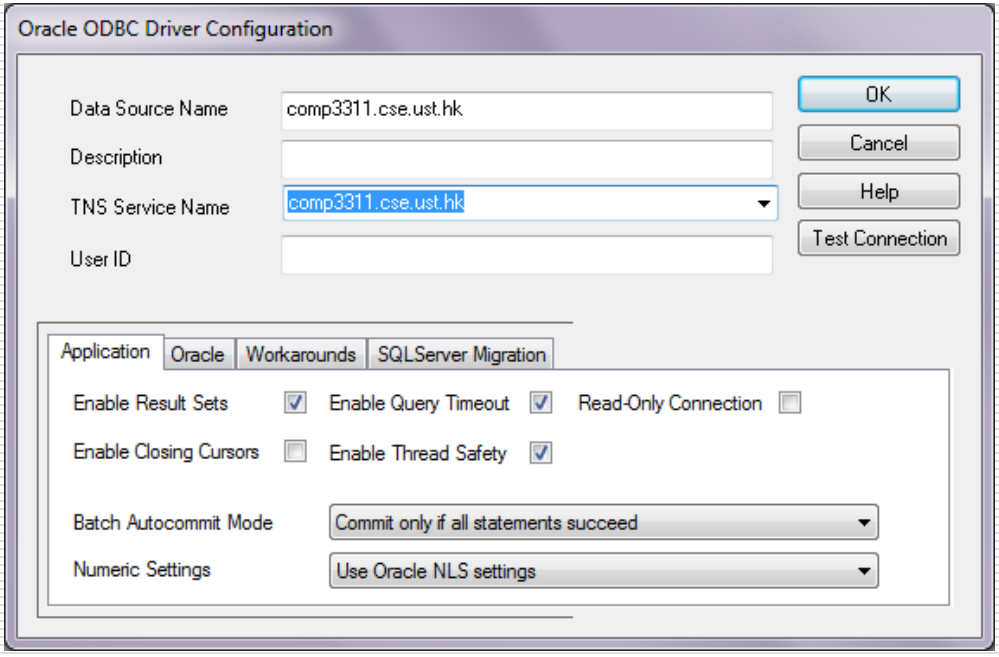
Appendix 1: Steps to set up ODBC environment for visual studio 4

- 3. Add the Oracle data source.



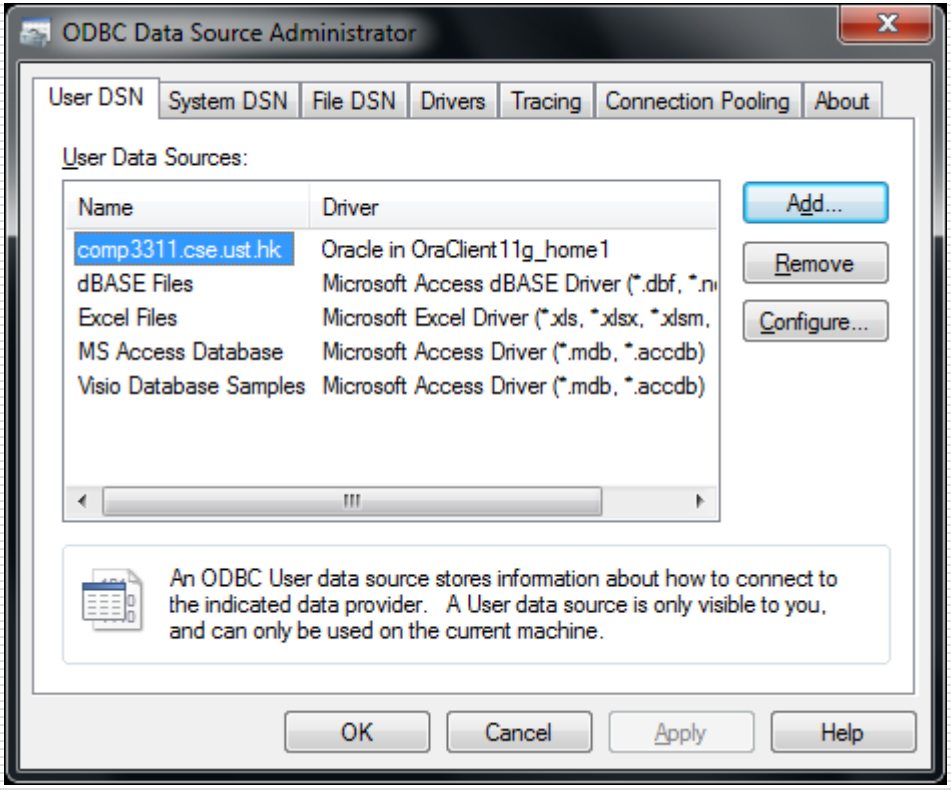
Appendix 1: Steps to set up ODBC environment for visual studio 5

- 4. Give a name to the Data Source (you need this name in the SQLConnect() function, the name here is “comp3311.cse.ust.hk”)



Appendix 1: Steps to set up ODBC environment for visual studio 6

- 5. The name "comp3311.cse.ust.hk" appears in the data source administrator window. You are now ready to write ODBC codes!



Appendix 1: Steps to set up ODBC environment for visual studio 7

- (Optional) Only laptops connected to the UST network can access the CSE Oracle server. To enable setting up the data-source you should download following two files to the laptop:

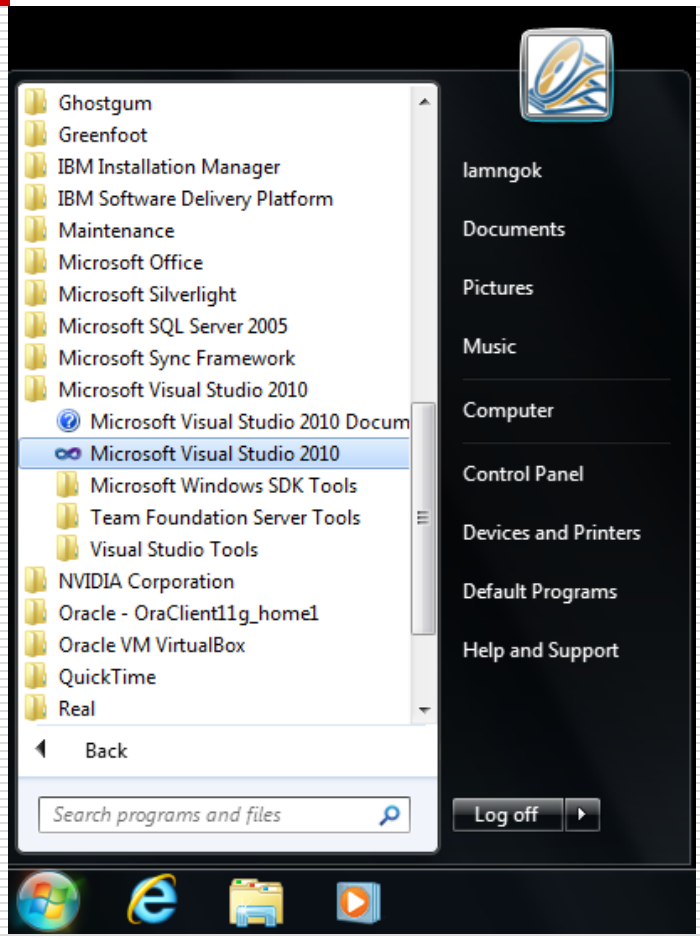
  http://course.cse.ust.hk/comp3311/labs/sqlnet.ora

  http://course.cse.ust.hk/comp3311/labs/tnsnames.ora

- (Optional) Store the two files to the default Oracle_path that you selected for installing the ODBC driver (for me that was c:\app\lamngok\product\11.2.0\client_1\Network\Admin).

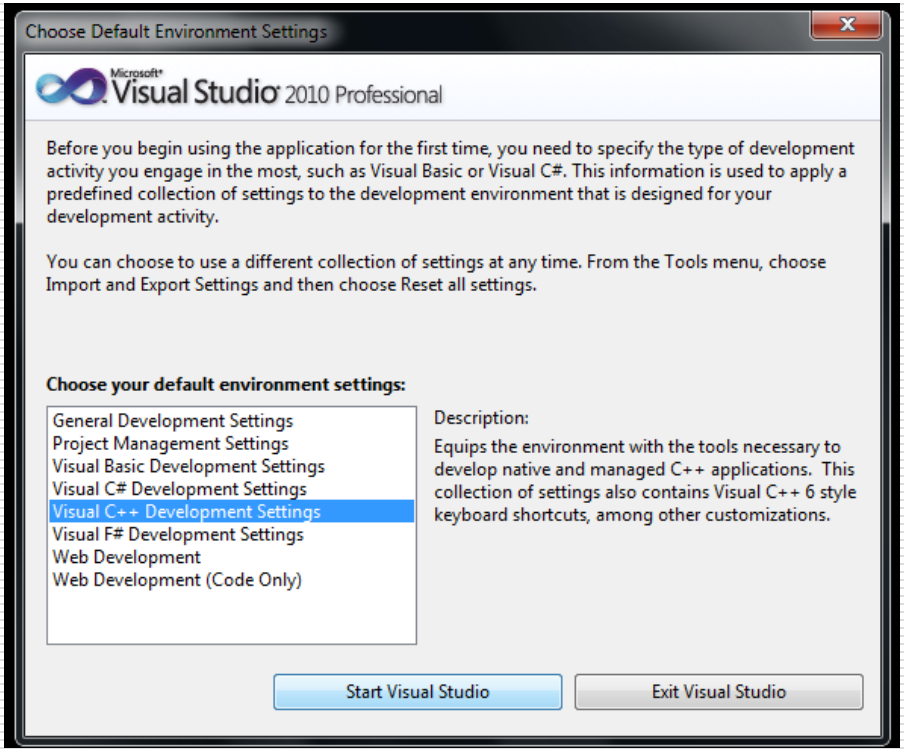
Appendix 2: Working with Visual Studio 2010 1

- To start Visual Studio, one need to locate the visual studio 2010 package.
- Click the "start" button, and expand the folder "Microsoft Visual Studio 2010".
- Click on the "Microsoft Visual Studio 2010" icon as shown at the right to start it.



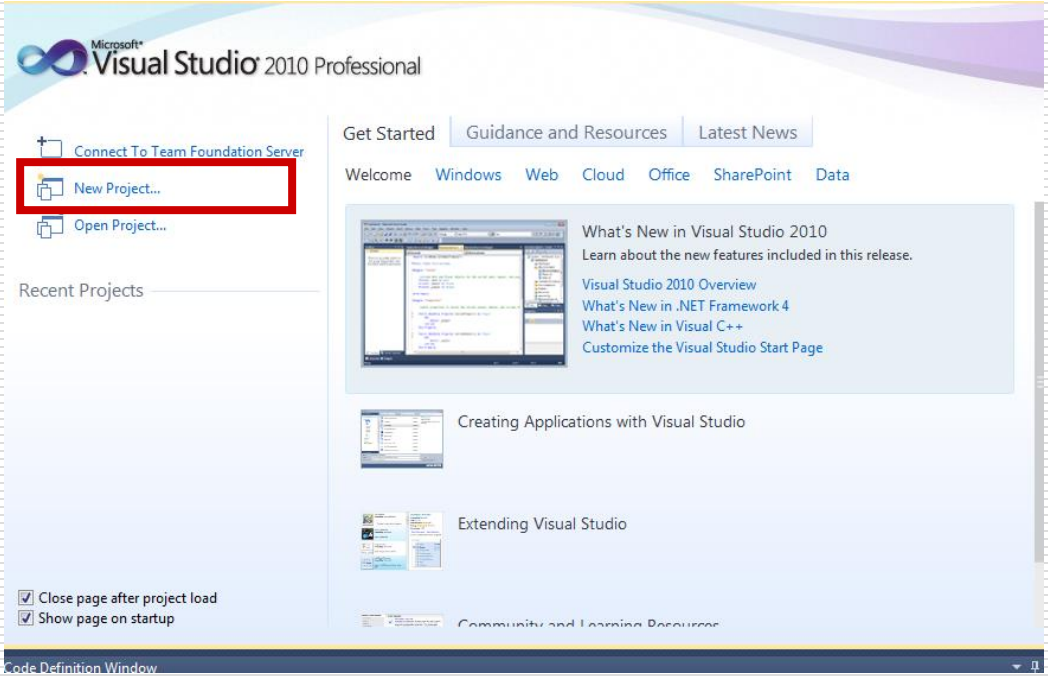
Appendix 2: Working with Visual Studio 2010 2

- Select “Visual C++ Development Settings”



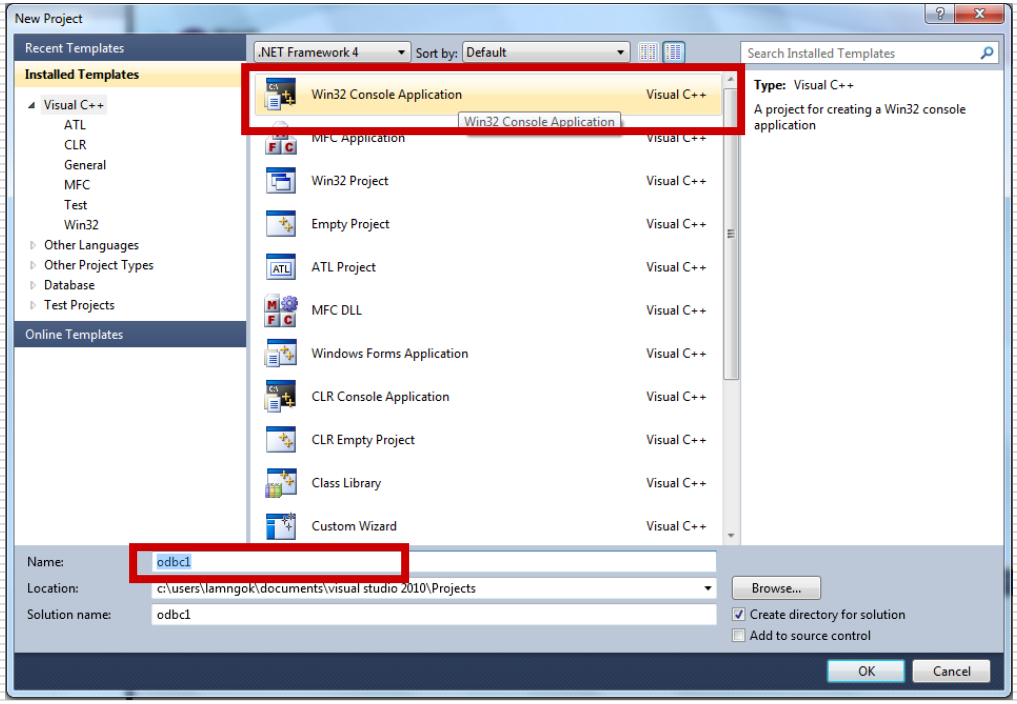
Appendix 2: Working with Visual Studio 2010 3

- Select "New Project…" to start a new C/C++ project



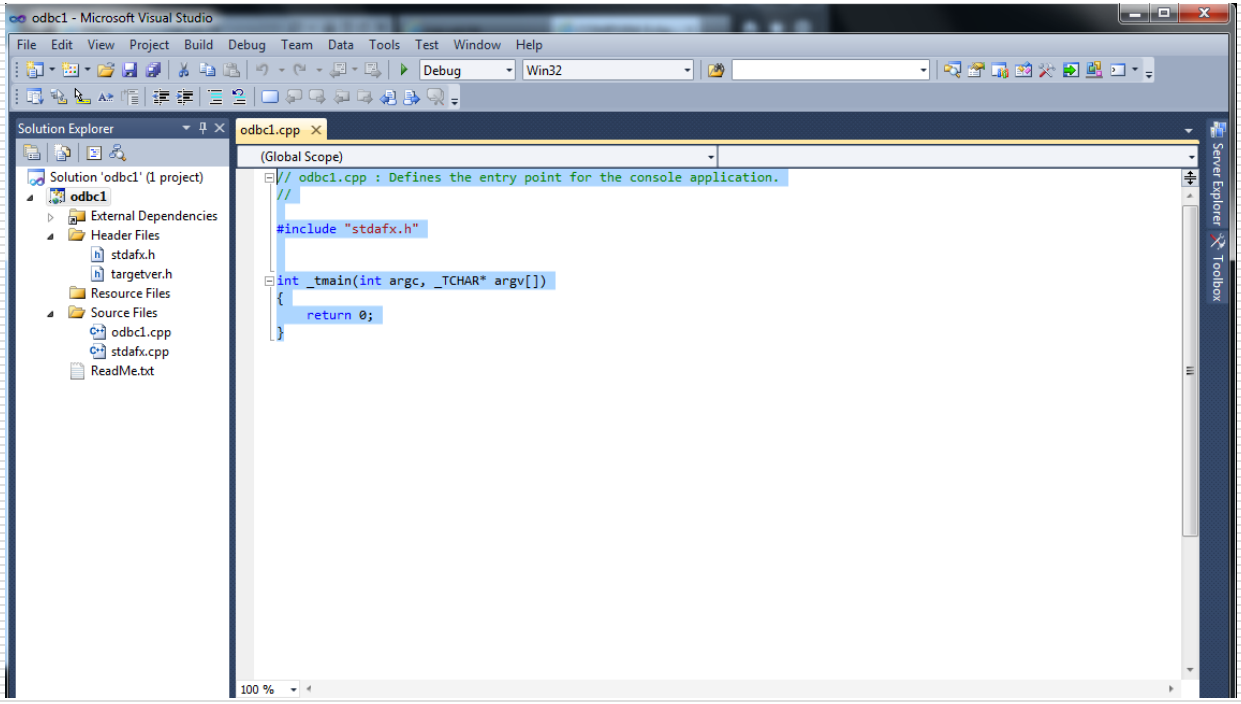
Appendix 2: Working with Visual Studio 2010 4

- Select “Win32 Console Application”, and give the C++ project a name.



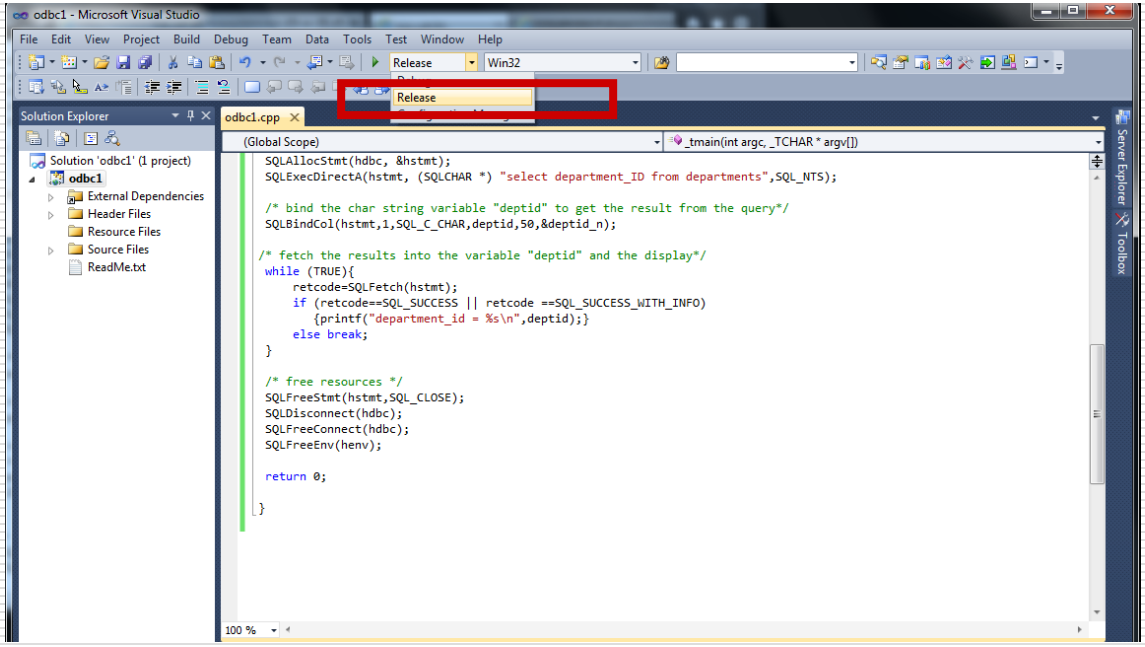
Appendix 2: Working with Visual Studio 2010 5

- Then click “next” and “finish”, and you will see the following:



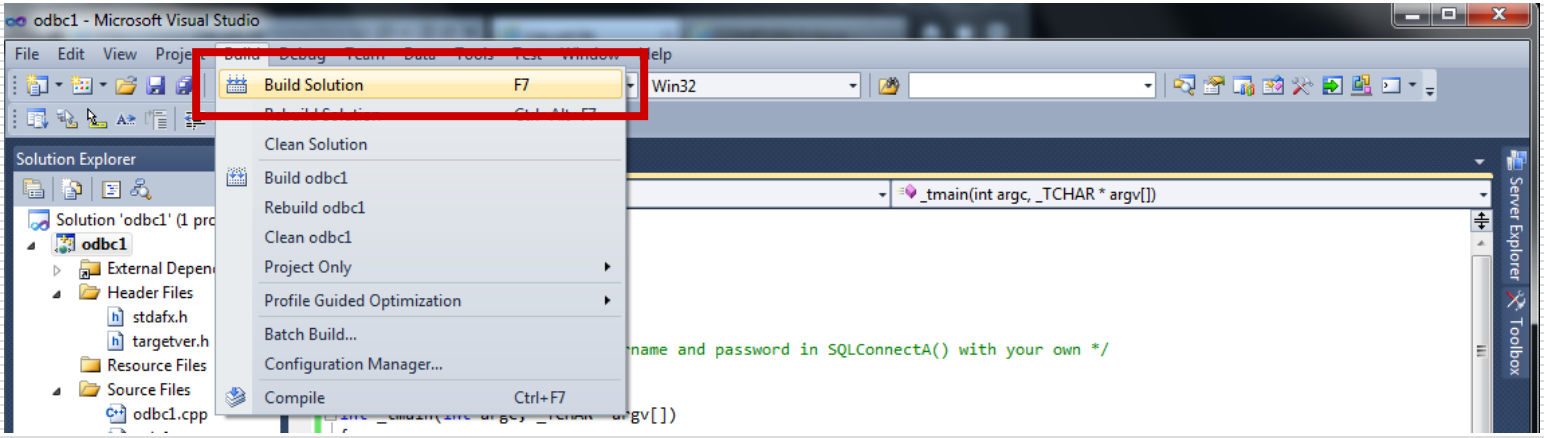
Appendix 2: Working with Visual Studio 2010 6

- Replace the C-codes with odbc1.cpp you have downloaded, and select "Release" for the compilation option.



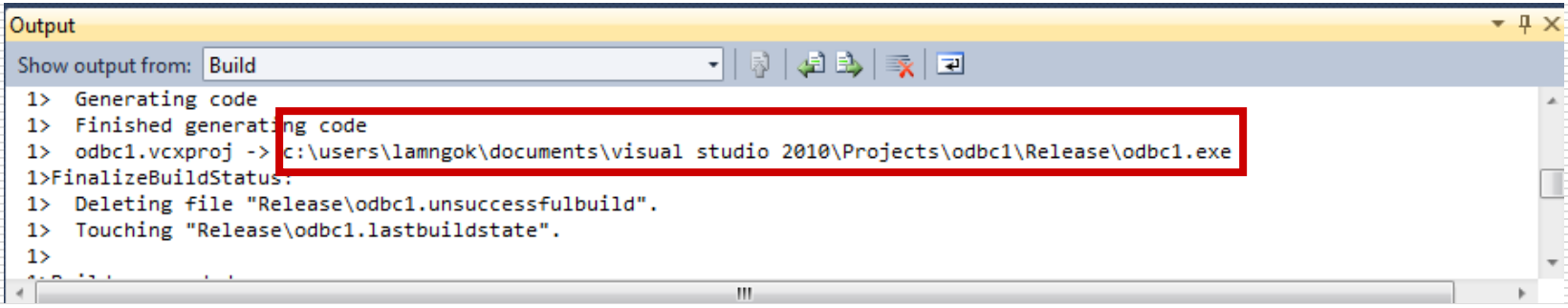
Appendix 2: Working with Visual Studio 2010 7

- Save the file and compile the program by selecting "Build Solution".



Appendix 2: Working with Visual Studio 2010 8

- Open a Windows command window (type "cmd" at "run…")
- Change to the directory where the executable is located:



Appendix 2: Working with Visual Studio 2010 9

- Then run the executable from the path.
- Note that the connection to the Oracle server is successful, and results are returned.