实验指导 – C++数据库开发环境配置

一、实验目的

1. 熟悉在 Ubuntu 系列的 Linux 发行版中配置 C++数据库开发环境的搭建，主要是为 C++ OTLv4 的使用作准备。

二、实验环境

1. 操作系统：Ubuntu 9.10 或更新版本；
2. Oracle Database Server：10g R2 XE。
3. MySQL Database Server：5.1+。

三、设置 Oracle 共享库路径

基于 OTL 的 C++应用程序需链接如 libclntsh.so 等共享库，OTL 内部需使用相应的头文件，如 oci.h。

1. 创建 liboracle.conf 文件

```
$ cd /etc/ld.so.conf.d
$ sudo vi liboracle.conf
# 将以下内容加入 liboracle.conf 文件中：
/usr/lib/oracle/xe/app/oracle/product/10.2.0/server/lib
# 保存退出
```

注：/usr/lib/oracle/xe/app/oracle/product/10.2.0/server 目录是 $ORACLE_HOME，也即 Orcale 的安装目录，如果 Oracle 不是安装在该目录，则需更改到正确的目录。

2. 执行 ldconfig 命令，使配置生效

```
$ sudo ldconfig
```

3. 创建链接文件

```
$ cd /usr/lib
$ sudo ln -s $ORACLE_HOME/lib/libclntsh.so.10.1 libclntsh.so
```

4. 测试代码

```
// otl_test.cpp
#include <iostream>

#define OTL_ORA10G_R2 // Compile OTL 4.0/OCI10gR2
#include "otlv4.h" // include the OTL 4.0 header file
```

```
int main() {
  using namespace std;
  otl_connect db; // connect object
  otl_connect::otl_initialize(); // initialize OCI environment

  try {
    // 以下 用户名/口令 按实际情况而定
    db.rlogon("dbdev/q1w2e3"); // connect to Oracle
    cout << "Connected to oracle xe 10g." <<endl;
  } catch (otl_exception& p) { // intercept OTL exceptions
    cerr << p.msg << endl; // print out error message
  }
  db.logoff(); // disconnect from Oracle

  return 0;
}
```

5. 编译、运行

将 otlv4.h 头文件复制到项目源文件所在的路径。

```
$ g++ -o otl_test otl_test.cpp -I$ORACLE_HOME/rdbms/public -lclntsh

$ ./otl_test
Connected to oracle xe 10g.
```

四、设置 MySQL 共享库路径

1. 安装 unixodbc

```
$ sudo aptitude install unixodbc-dev
```

2. 安装 libmyodbc

```
$ sudo aptitude install libmyodbc
```

安装完毕后，该共享库文件位于/usr/lib/odbc/目录。

3. 复制库文件、创建链接文件

```
$ sudo cp /usr/lib/odbc/libmyodbc.so /usr/lib/libmyodbc3_r-3.51.19.so
$ cd /usr/lib
$ sudo ln -s libmyodbc3_r-3.51.19.so libmyodbc.so
```

注意：上述库文件的版本需根据实际情况而定，不一定是 libmyodbc3_r-3.51.19.so 这个文件名，关键看连接到 libmyodbc.so 库的应用程序运行时报错信息，根据该出错信息可以获悉所需的文件名，如应用程序报以下错误：

```
./otl_test2: error while loading shared libraries: libmyodbc3_r-3.51.19.so: cannot open shared object file: No such file or directory
```

由上述错误信息得知，所需文件是 libmyodbc3_r-3.51.19.so，所以**步骤 2** 复制文件时将新文件命名为 libmyodbc3_r-3.51.19.so 即可。

另外，该问题在 Redhat 或其它 Linux 发行版中不一定存在，届时需按具体情况分析。

4. 示例代码

```
// otl_test2.cpp

#include <iostream>

#define OTL_ODBC // Compile OTL 4.0/ODBC
// The following #define is required with MyODBC 3.51.11 and higher
#define OTL_ODBC_SELECT_STM_EXECUTE_BEFORE_DESCRIBE
#define OTL_ODBC_UNIX // uncomment this line if UnixODBC is used
#include "otlv4.h" // include the OTL 4.0 header file

int main() {
  using namespace std;
  otl_connect db; // connect object
  otl_connect::otl_initialize(); // initialize OCI environment

  try {
    // 以下 用户名/口令 按实际情况而定
    db.rlogon("UID=root;PWD=abcdef;DSN=attendance_sys");
    // 或
    // db.rlogon("root/abcdef@attendance_sys"); // connect to ODBC
    cout << "Connected to MySQL." <<endl;
  } catch (otl_exception& p) { // intercept OTL exceptions
    cerr << p.msg << endl; // print out error message
  }
  db.logoff(); // disconnect from MySQL
}
```

5. 编译、运行

将 otlv4.h 头文件复制到项目源文件所在的路径。

```
$ g++ -o otl_test2 otl_test2.cpp -lmyodbc
$
$ ./otl_test2
Connected to MySQL.
```