# 6. Kết nối database:

## C++

Code phải nói là nhìn như đống shit.

Các kết nối database được define ở file vsSrvrManagerUtil.cpp:

\_ vsConnectToDatabase

\_ vsConnectToDatabaseGlobal

\_ vsConnectToDatabaseWithDSN

\_ vsCleanUpDbConnection

\_ vsCleanUpGlobalConn

\_ vsSrvrCleanUpGlobalConn

\_ …

Muốn sử dụng ta khai báo zô main hoặc zô hàm như saU;

Connection \*con;

con = vsConnectToDatabase();

Khi đó muốn sử dụng thì chỉ càn gán con trỏ **param** cho hàm muốn sử dụng:

listExecUsers(Connection \*con, …)

Tuy nhiên trong project **exec** thì sử dụng **getConnection** của **cCgiGi**:

Đầu tiên khai thẳng trong khai báo mở đầu của **exec.cpp**:

CCgiGlobalInfo gCgiGI("dmn");

Sau đó muốn sử dụng connection thì chỉ việc get, không cần cho vào param của hàm gì hết:

userList = listExecUsers(gCgiGI.getConnection(),…);

Vitalsuite sử dụng 2 cách để gửi query, 1 cách hơi giống java, 1 cách thì siêu ngu luôn:

### Cách 1: sử dụng **dbTable:**

RWCStringVector

listExecUsers(Connection \*con,int \*len, int domainId, u\_long publishMask, u\_long includeChildDomain, RWCString &errMsg)

{

RWCStringVector list;

RWCString qstring, whereClause;

SQLRETURN sqlS;

dbTable t(con);

int count = 0;

qstring = RWCString("select distinct USERID from VNREPORTTABLE");

if(publishMask)

{

whereClause = RWCString("RPTPUBLISHED = ");

whereClause += cvstr(1);

}

if(domainId != DMN\_ALL)

{

if(includeChildDomain){

//Get domainID and its child domain

RWCString subDomainList = " DOMAINID in ( ";

char tmpbuf[64];

memset(tmpbuf, 0, 64);

sprintf(tmpbuf, " %d", domainId);

subDomainList += " select SubDomainID from domaindescendents ";

subDomainList += " where domainId =";

subDomainList += tmpbuf;

subDomainList += " )";

if(whereClause.length())

whereClause += RWCString(" and ");

whereClause += subDomainList;

}

else{

if(whereClause.length())

whereClause += RWCString(" and ");

whereClause += RWCString(" DOMAINID = ");

whereClause += cvstr(domainId);

whereClause += RWCString(" ");

}

}

if(whereClause.length())

{

qstring += RWCString(" where ");

qstring += whereClause;

}

qstring += RWCString(" order by USERID ");

sqlS = t.ReadFirst(qstring);

if(sqlS != SQL\_SUCCESS && sqlS != SQL\_NO\_DATA)

{

errMsg = t.ErrorMsg();

}

else

{

while(sqlS == SQL\_SUCCESS)

{

list.insert(t[1]);

count++;

sqlS = t.ReadNext();

}

}

\*len = count;

return list;

}

### Cách 2: Sử dụng các kế thừa của **dbTable**:

Ví dụ **groupTable** là một kế thừa của **dbTable** như sau:

class DBDLL\_API groupTable : public dbTable {

public:

groupTable ( Connection \*cp ) : dbTable (cp,"VNGroupTable")

{

column ( "groupID", record.index );

column ( "groupName", record.name , sizeof(record.name));

column ( "groupType", record.groupType );

column ( "domainID", record.domainID);

column ( "lastUpdateTime", record.lastUpdateTime, sizeof(record.lastUpdateTime) );

}

GroupTable\_Struct record;

};

Khi đó sử dụng kết nối như sau:

groupTable grpTable(gCgiGI.getConnection());

char whereString[128];

sprintf(whereString,"groupType = %d and DomainId = %d",

RESOURCE\_GROUP, gDomainId);

grpTable.where(whereString);

grpTable.orderby("groupName");

SQLRETURN r = grpTable.ReadFirst();

while(r == SQL\_SUCCESS) {

fprintf(cgiOut,",{ value:\"%d\", name:'%s', selected: false}",

grpTable.record.index, grpTable.record.name);

r = grpTable.ReadNext();

}