Beschreibung der wichtigsten CLI-Funktionen in ODBC

SQLAllocHandle legt ein Umgebungs-, Verbindungs-, Anweisungs- oder Deskriptorhandle an.

SQLRETURN **SQLAllocHandle**(SQLSMALLINT HandleType, SQLHANDLE InputHandle, _Out_ SQLHANDLE *OutputHandle);

HandleType: Typ des anzulegenden Handles

InputHandle: Das Handle in dessen Kontext das neue Handle angelegt werden soll. Wenn

HandleType = SQL_HANDLE_ENV ist, dann musss InputHandle SQL_NULL_HANDLE sein. Wenn HandleType = SQL_HANDLE_DBC ist, dannmuss es ein Environment Handle sein und wenn es ein SQL_HANDLE_STMT oder SQL_HANDLE_DESC ist, dann

muss es ein Connection Handle sein

OutputHandle: Zeiger auf das neu anzulegende Handle.

SQLSetEnvAttr definiert sogenannte Umgebungsattribute.

SQLRETURN SQLSetEnvAttr(SQLHENV EnvironmentHandle, SQLINTEGER

Attribute, SQLPOINTER ValuePtr, SQLINTEGER StringLength);

EnvironmentHandle: Das zugehörige Environment Handle

Attribute: Das zu setzende Attribut.

ValuePtr: Zeiger auf den Wert des Attributes.

StringLength: Falls ValuePtr auf einen String zeigt, ist hier die Zeichenanzahl anzugeben.

SQLSetConnectAttr definiert sogenannte Verbindungsattribute.

SQLRETURN SQLSetConnectAttr(SQLHDBC ConnectionHandle, SQLINTEGER

Attribute, SQLPOINTER ValuePtr, SQLINTEGER StringLength);

ConnectionHandle: Das zugehörige Connection Handle

Attribute: Das zu setzende Attribut.

ValuePtr: Zeiger auf den Wert des Attributes.

StringLength: Falls ValuePtr auf einen String zeigt, ist hier die Zeichenanzahl anzugeben.

SQLDriverConnect baut eine Verbindung zur Datenbank auf

SQLRETURN SQLDriverConnect(SQLHDBC hdbc, SQLHWND hwnd,

SQLCHAR *szConnStrIn, SQLSMALLINT cchConnStrIn, SQLCHAR szConnStrOut, SQLSMALLINT cchConnStrOutMax,

SQLSMALLINT *pcchConnStrOut,
SQLUSMALLINT fDriverCompletion);

hdbc: Das Connection Handle

hwnd: Window Handle für die Verwendung von Dialogfenstern – sonst NULL

szConnStrIn: String mit den Verbindungsdaten

cchConnStrIn: Länge des Strings mit den Verbindungsdaten

szConnStrOut: Vom DBS vervollständigter String mit Verbindungsdaten. Puffergröße sollte 1024

Zeichen sein.

cchConnStrOutMax: Größe des Puffers für szConnStrOut

pcchConnStrOut: Puffer in den die Länge des szConnStrOut geschrieben werden kann

fDriverCompletion: Flag zur Steuerung, ob weitere vervollständigende treiberangaben nötig sind

SQLConnect baut eine Verbindung zu einer Datenquelle auf

SQLRETURN SQLConnect(SQLHDBC hdbc,

SQLCHAR* ServerName, SQLSMALLINT NameLength1, SQLCHAR* UserName, SQLSMALLINT NameLength2,

SQLCHAR* Authentication, SQLSMALLINT NameLength3);

hdbc: Das Connection Handle
ServerName: Name der Datenquelle
NameLength1: Länge des Servernamens
UserName: Name des Benutzers
NameLength2: Länge des Nutzernamens

Authentication: Passwort

NameLength3: Länge des Passwortes

SQLExecDirect veranlasst die einmalige sofortige Ausführung einer SQL-Anweisung

SQLRETURN SQLExecDirect(SQLHSTMT StatementHandle, SQLCHAR*

StatementText, SQLINTEGER TextLength);

StatementHandle: Das zuvor angelegte Handle für die Anweisung

StatementText: Das SQL Statement als String

TextLenght: Die Zeichenzahl des SQL Statements

SQLBindCol verbindet den Speicherplatz von Anwendungsdaten mit den Spalten der Ergebnismenge

SQLRETURN **SQLBindCol(**SQLHSTMT StatementHandle, SQLUSMALLINT

ColumnNumber, SQLSMALLINT TargetType, SQLPOINTER

TargetValuePtr, SQLLEN BufferLength, SQLLEN

*StrLen_or_IndPtr);

StatementHandle: Das zuvor angelegte Handle für das SQL Statement

ColumnNumber: Nummer der zu verbindenden Spalte (1..n)

TargetType: Datentyp der zu bindenden Spalte

TargetValuePtr: Zeiger auf den Datenpuffer

BufferLenght: Größe des Puffers

StrLen or IndPtr: Zeigervariable für die Puffergröße

SQLFetch holt den nächsten Ergebnisdatensatz und liefert die Daten der verbundenen Spalten

SQLRETURN SQLFetch(SQLHSTMT StatementHandle);

StatementHandle: Das zuvor angelegte Handle für das SQL Statement

SQLDisconnect schließt die angegebene Verbindung zur Datenbank

SQLRETURN SQLDisconnect(SQLHDBC ConnectionHandle);

ConnectionHandle: Das Connection Handle

SQLFreeHandle gibt Ressourcen free, die durch das entsprechende Handle belegt sind

SQLRETURN SQLFreeHandle(SQLSMALLINT HandleType, SQLHANDLE

Handle);

HandleType: Der Typ des freizugebenden Handles

Handle: Das freizugebende Handle

Ein minimales C-Code-Beispiel für ODBC

```
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <sqlext.h>
#include <sql.h>
#define NAME LEN 50
#define ORT LEN 21
int main()
  SQLHENV henv;
  SQLHDBC hdbc;
  SQLHSTMT hstmt = 0;
  SQLRETURN retcode:
  SQLCHAR szName[NAME_LEN], szOrt[ORT_LEN], szKunr[NAME_LEN];
  SQLLEN cbKunr = 0, cbName = 0, cbOrt = 0;
  retcode = SQLAllocHandle(SQL HANDLE ENV, SQL NULL HANDLE, &henv);
                                                           // Allocate environment handle
  retcode = SQLSetEnvAttr(henv, SQL ATTR ODBC VERSION, (SQLPOINTER*)SQL OV ODBC3, 0);
                                             // Set the ODBC version environment attribute
  retcode = SQLAllocHandle(SQL HANDLE DBC, henv, &hdbc);
                                                          // Allocate connection handle
  // >>>>> Connect to data base <<<<<<<
  SQLCHAR retconstring[1024];
  char* Connection = "DRIVER={SQL Server};SERVER=141.56.2.45, 1433;
            DATABASE=DB Verkauf;UID=XXXXXXX;PWD=XXXXXXXX;";
  retcode = SQLDriverConnect(hdbc, NULL, (SQLCHAR*)Connection, SQL NTS, retconstring, 1024,
            NULL, SQL DRIVER NOPROMPT);
  // >>>>> Connect to data sorce <<<<<<<
  // retcode = SQLConnect(hdbc, (SQLCHAR*) "MSSQLSERVER", SQL NTS, (SQLCHAR*) "student",
            SQL NTS, (SQLCHAR*) "pw student", SQL NTS);
  retcode = SQLAllocHandle(SQL HANDLE STMT, hdbc, &hstmt);
                                                            // Allocate statement handle
  // >>>>> perform the SQL statement <<<<<<<
  char* selectStmt = "SELECT Kunr, Name, Ort FROM Kunde";
  retcode = SQLExecDirect(hstmt, (SQLCHAR *)selectStmt, SQL NTS);
  printf("Performed: %s\n", selectStmt);
  retcode = SQLBindCol(hstmt, 1, SQL C CHAR, szKunr, 100, &cbKunr);
                                                                       // Bind columns 1
  retcode = SQLBindCol(hstmt, 2, SQL_C_CHAR, szName, NAME_LEN, &cbName);
                                                                      // Bind columns 2
  retcode = SQLBindCol(hstmt, 3, SQL C CHAR, szOrt, ORT LEN, &cbOrt);
                                                                       // Bind columns 3
```

```
for (int i=0;; i++)
                 // Fetch and print each row of data. On an error, display a message and exit.
    retcode = SQLFetch(hstmt);
    if (retcode == SQL_ERROR | | retcode == SQL_SUCCESS_WITH_INFO)
      printf("SQL ERROR\n");
    if (retcode == SQL_SUCCESS || retcode == SQL_SUCCESS_WITH_INFO)
      printf("%d: %s %s %s\n", i + 1, szKunr, szName, szOrt);
    else
      break;
  }
  SQLFreeHandle(SQL_HANDLE_STMT, hstmt);
  SQLDisconnect(hdbc);
  SQLFreeHandle(SQL HANDLE DBC, hdbc);
  SQLFreeHandle(SQL HANDLE ENV, henv);
  return 0;
}
```