#include <iostream>

#include <windows.h>

#include <sql.h>

#include <sqlext.h>

int main() {

setlocale(LC\_ALL, "Russian");

SQLHENV env; // Определение дескриптора окружения

SQLHDBC dbc; // Определение дескриптора соединения

SQLHSTMT stmt; // Определение дескриптора SQL-выражения

SQLRETURN ret; // Переменная для хранения кодов возврата

// Инициализация окружения

ret = SQLAllocHandle(SQL\_HANDLE\_ENV, SQL\_NULL\_HANDLE, &env);

ret = SQLSetEnvAttr(env, SQL\_ATTR\_ODBC\_VERSION, (SQLPOINTER)SQL\_OV\_ODBC3, SQL\_IS\_INTEGER);

// Инициализация соединения

ret = SQLAllocHandle(SQL\_HANDLE\_DBC, env, &dbc);

ret = SQLDriverConnect(dbc, NULL,

(SQLWCHAR\*)L"DRIVER={SQL Server};"

L"Server=DESKTOP-HDR7MIV;" //ТУТ УКАЗАТЬ СЕРВЕР

L"Database=veb7new;", //ТУТ УКАЗАТЬ НАЗВАНИЕ БАЗЫ ДАННЫХ

SQL\_NTS, NULL, 0, NULL, SQL\_DRIVER\_COMPLETE);

// Проверка успешности соединения

if (ret == SQL\_SUCCESS || ret == SQL\_SUCCESS\_WITH\_INFO) {

std::cout << "Соединение с базой данных успешно установлено!" << std::endl << std::endl;

// Инициализация SQL-выражения

ret = SQLAllocHandle(SQL\_HANDLE\_STMT, dbc, &stmt);

// Выполнение SQL-запроса (ВЫ МОЖЕТЕ ИЗМЕНИТЬ ЗАПРОС)

ret = SQLExecDirect(stmt, (SQLWCHAR\*)L"SELECT \* FROM pers", SQL\_NTS);

// Обработка результата SQL-запроса

if (ret == SQL\_SUCCESS || ret == SQL\_SUCCESS\_WITH\_INFO) {

SQLWCHAR name[50]; // Здесь порядок не важен, главное типы нужные указать

// Извлечение данных из результата

while (SQLFetch(stmt) == SQL\_SUCCESS) {

//SQLGetData(откуда данные,

//какой столбец,

//тип данных,

//где хранить результат,

//размер буфера для результата,

//куда записать длину данных (можно всегда NULL ставить));

SQLGetData(stmt, 1, SQL\_C\_WCHAR, name, sizeof(name), NULL);

//от ваших целей зависит сам вывод

std::wcout << name << std::endl;

//DATE в SQL устроен интереснее чем строки или числа

//printf("Дата релиза: %d-%d-%d\n\n", ts.year, ts.month, ts.day);

}

}

else {

std::cout << "Ошибка выполнения SQL-запроса!" << std::endl;

}

// Освобождение ресурсов

SQLFreeHandle(SQL\_HANDLE\_STMT, stmt);

SQLDisconnect(dbc);

SQLFreeHandle(SQL\_HANDLE\_DBC, dbc);

SQLFreeHandle(SQL\_HANDLE\_ENV, env);

}

else {

std::cout << "Ошибка подключения к базе данных!" << std::endl;

}

return 0;

}