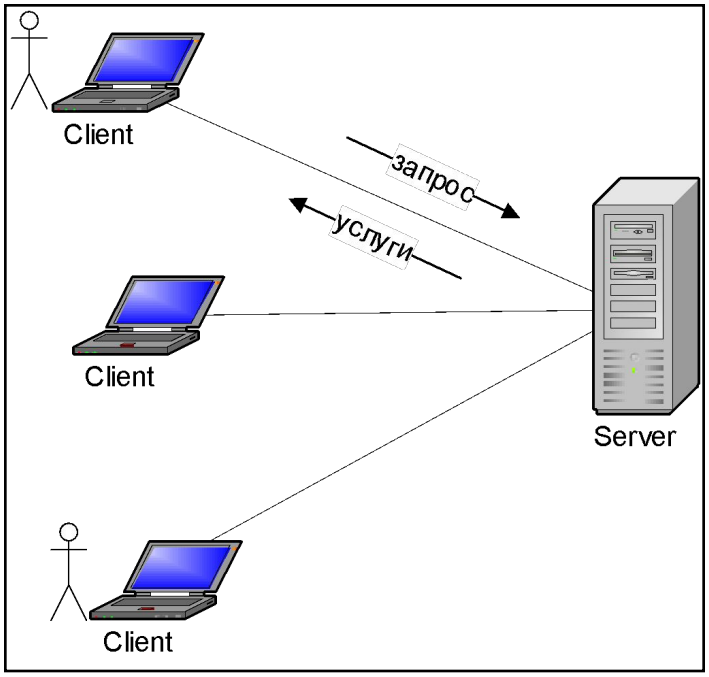
# СУБД в архитектуре «клиент-сервер»

File Server
Remote Data Access
Database Server
Application Server

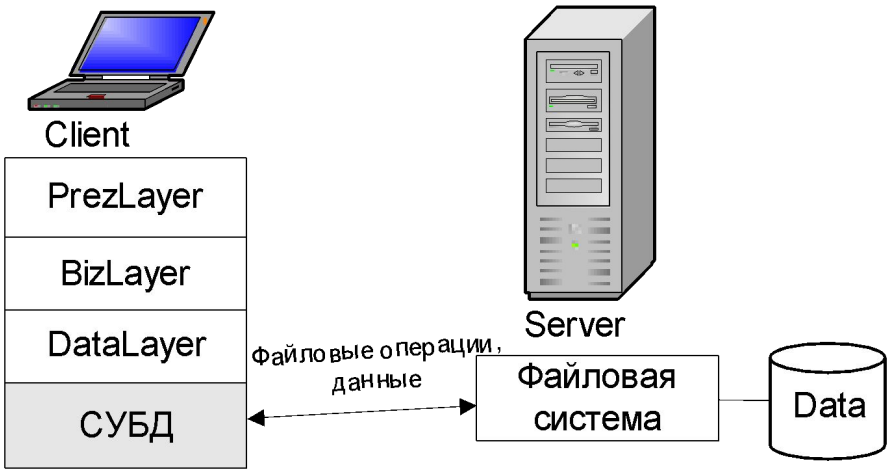
# СУБД в архитектуре «клиент-сервер»




**Группы функций программной системы:**

- Функции отображения и ввода данных
  **Уровень представления, Presentation Layer, PrezLayer**
- Прикладные функции – логика предметной области
  **Уровень бизнес- логики, Business Layer, BizLayer**
- Функции доступа к базам данных
  **Уровень данных, Database Layer, DataLayer**

# Модель «Файловый сервер» (File Server)



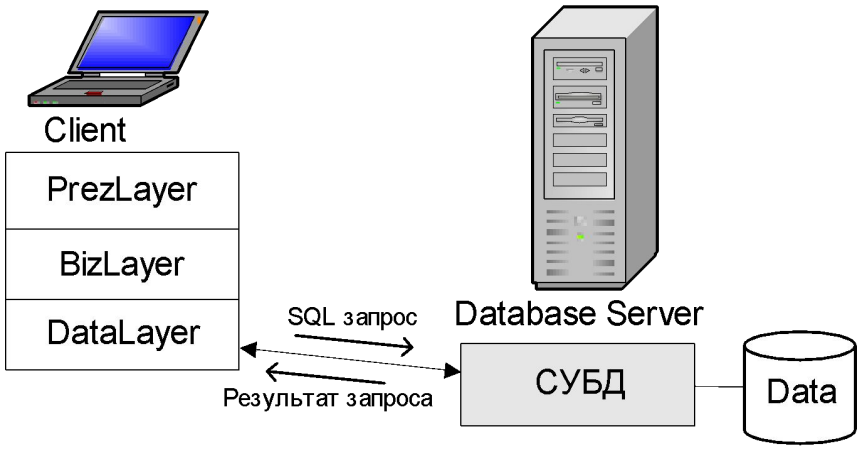

**Достоинства:**

- Использование единых ресурсов сервера (больших НМД)

**Недостатки:**

- Низкая защищенность (уровень файловой системы)

# Модель удаленного доступа к данным (Remote Data Access)




**Достоинства:**

- Повышенная защищенность данных
- Возможность управления транзакциями

**Недостатки:**

- Высокий сетевой трафик
- Сложность координации многопользовательских приложений

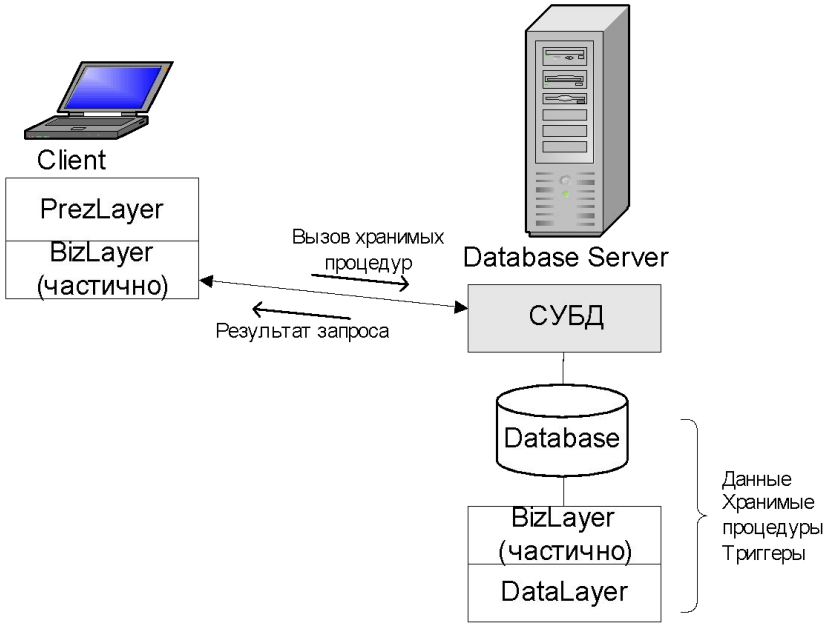
# Модель сервера баз данных (Database Server)

**Основные черты:**

- БД соответствует предметной области (концептуальная модель)
- БД активна – реагирует на изменения данных
- Имеет средства ограничения доступа к данным
- Имеет средства восстановления данных

Основа – хранимые процедуры и триггеры

# Модель сервера баз данных (Database Server)




**Достоинства:**

- Уменьшение сетевой нагрузки
- Уменьшение дублирования кода

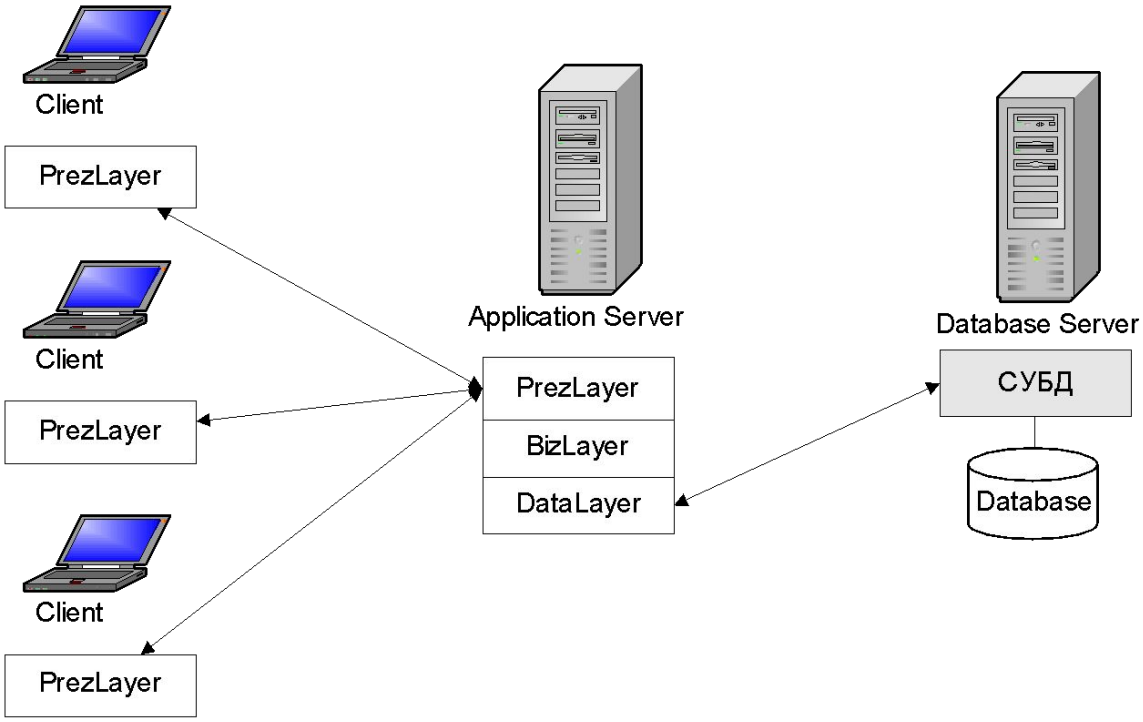
**Недостатки:**

- Высокая загрузка сервера (BL, DL, СУБД)
- Сложность разработки, если бизнес-логика частично реализуется на клиенте

# Модель сервера приложений (Application Server)

Трехуровневая архитектура «клиент-сервер»

- Клиент – уровень представления
- Бизнес логика – на отдельном сервере
- СУБД – на отдельном сервере



# Модель сервера приложений (Application Server)

**Достоинства:**

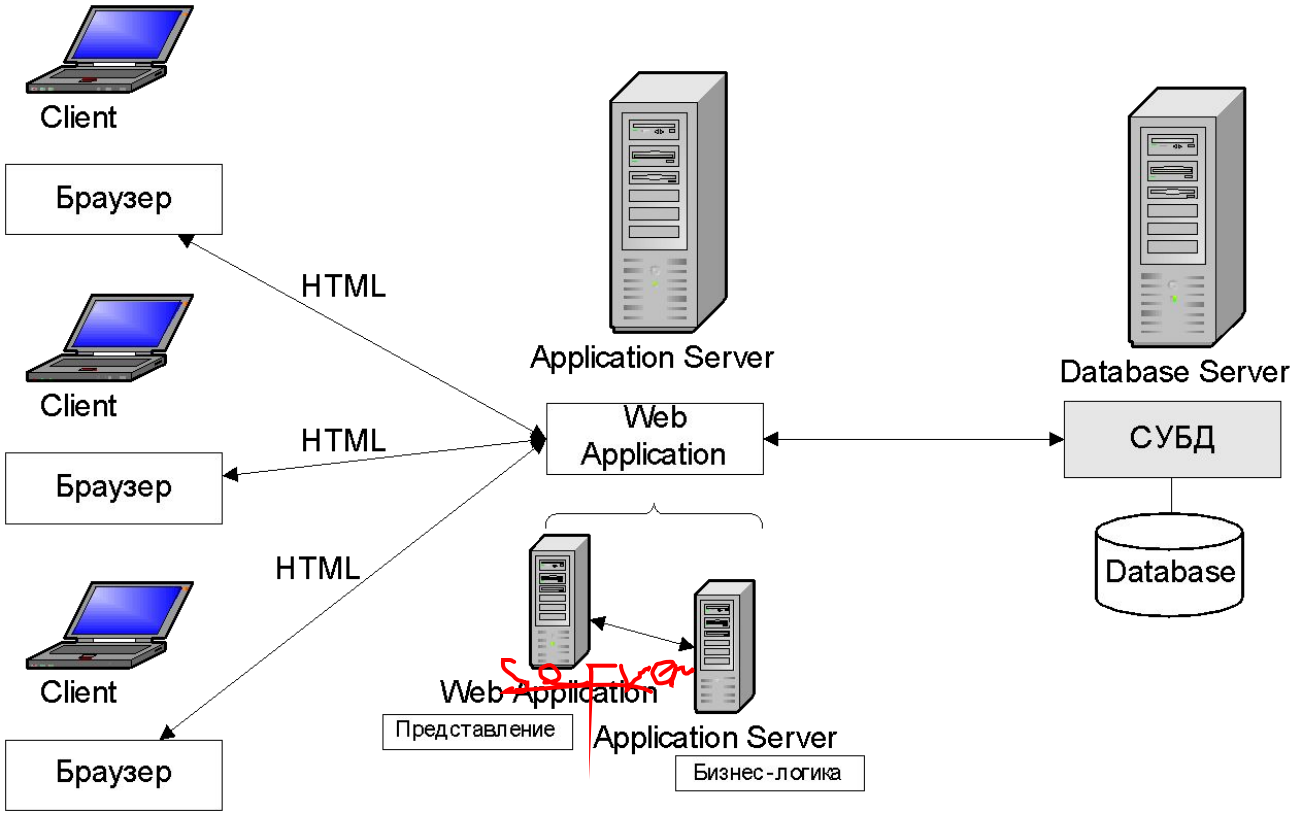
- «Тонкий клиент» - менее дорогая аппаратура
- Централизация бизнес-логики – централизация сопровождения (поддержка, замена оборудования, администрирование)
- Модульность – простая модификация и замена ПО
- Возможность равномерного распределения нагрузки

**Недостатки:**

- Сложность реализации

# Модель сервера приложений (Application Server)

## Web-приложение



# Модель сервера приложений (Application Server)

**Достоинства:**

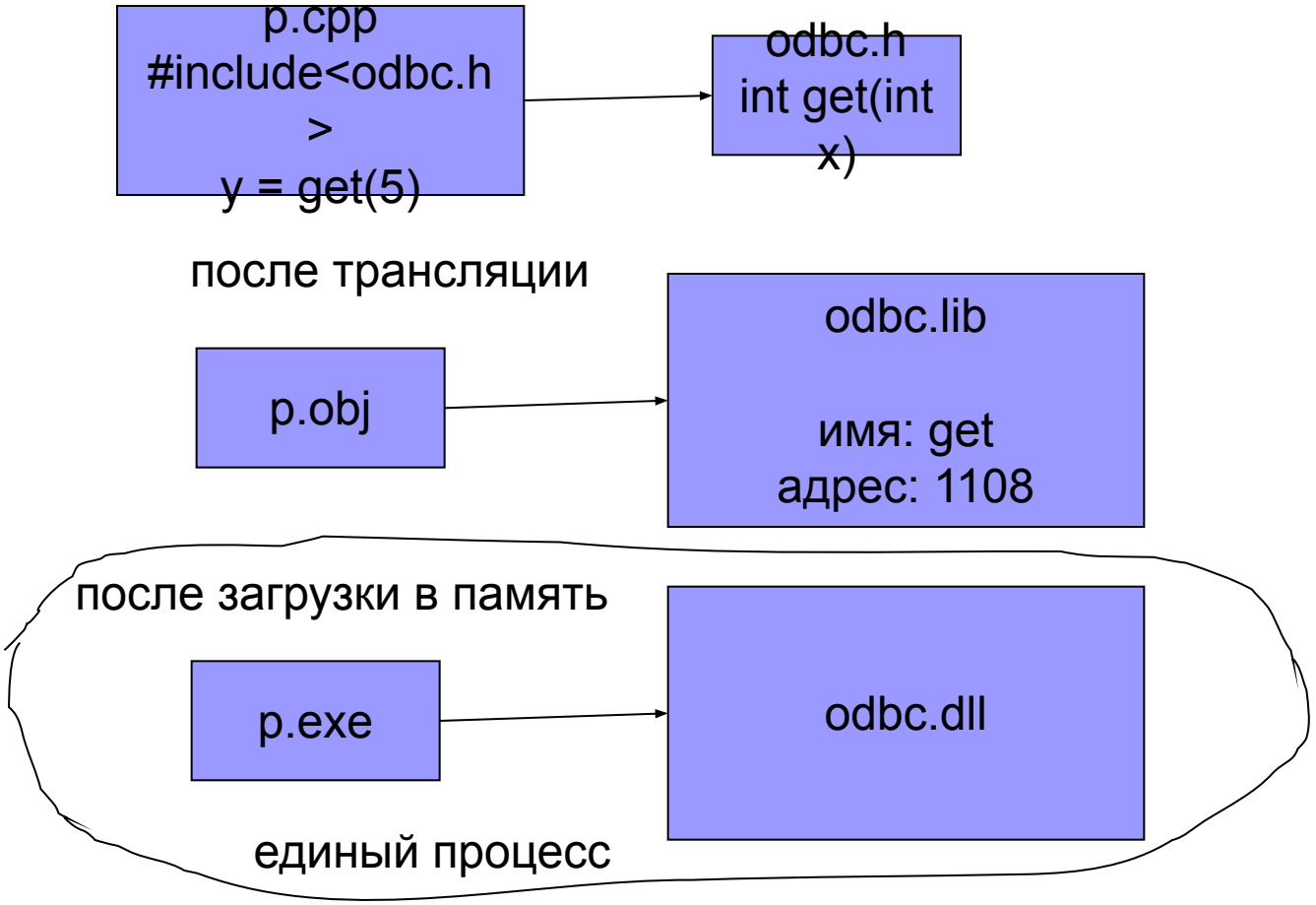
- Простота реализации HTML
- Независимость браузеров от платформы
- Высокий уровень стандартизации уровня представления (HTML)
- Масштабируемость – доступ с произвольного клиента, нет необходимости разворачивать на клиенте сложное ПО

**Недостатки:**

- Низкая надежность из-за ненадежности глобальной сети
- Слабая защищенность данных, передаваемых между клиентом и сервером приложения
- Отсутствие контроля масштабируемости (неожиданно высокая нагрузка на сервер)
- Ограниченная функциональность HTML

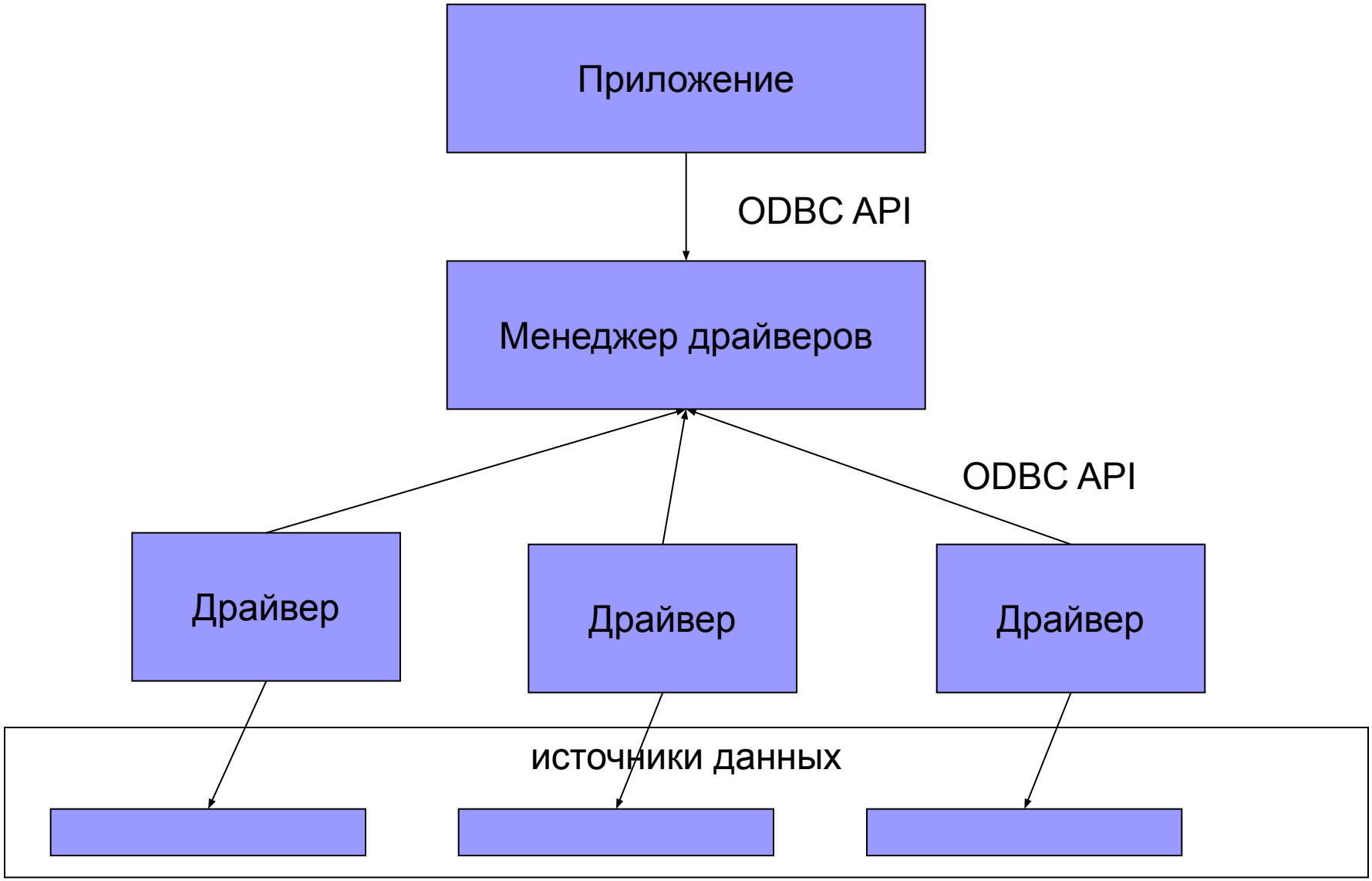
# Знакомство с DLL

**DLL** – динамически связываемая библиотека



# ODBC

ODBC – Open Database Connectivity открытая система связи с базами данных



# ODBC

**Задачи, выполняемые приложением:**

- выбор источника данных
- предоставление SQL операторов на выполнение
- получение результата
- обработка ошибок
- запрос на выполнение фиксации или отката транзакции
- отсоединение от источника данных

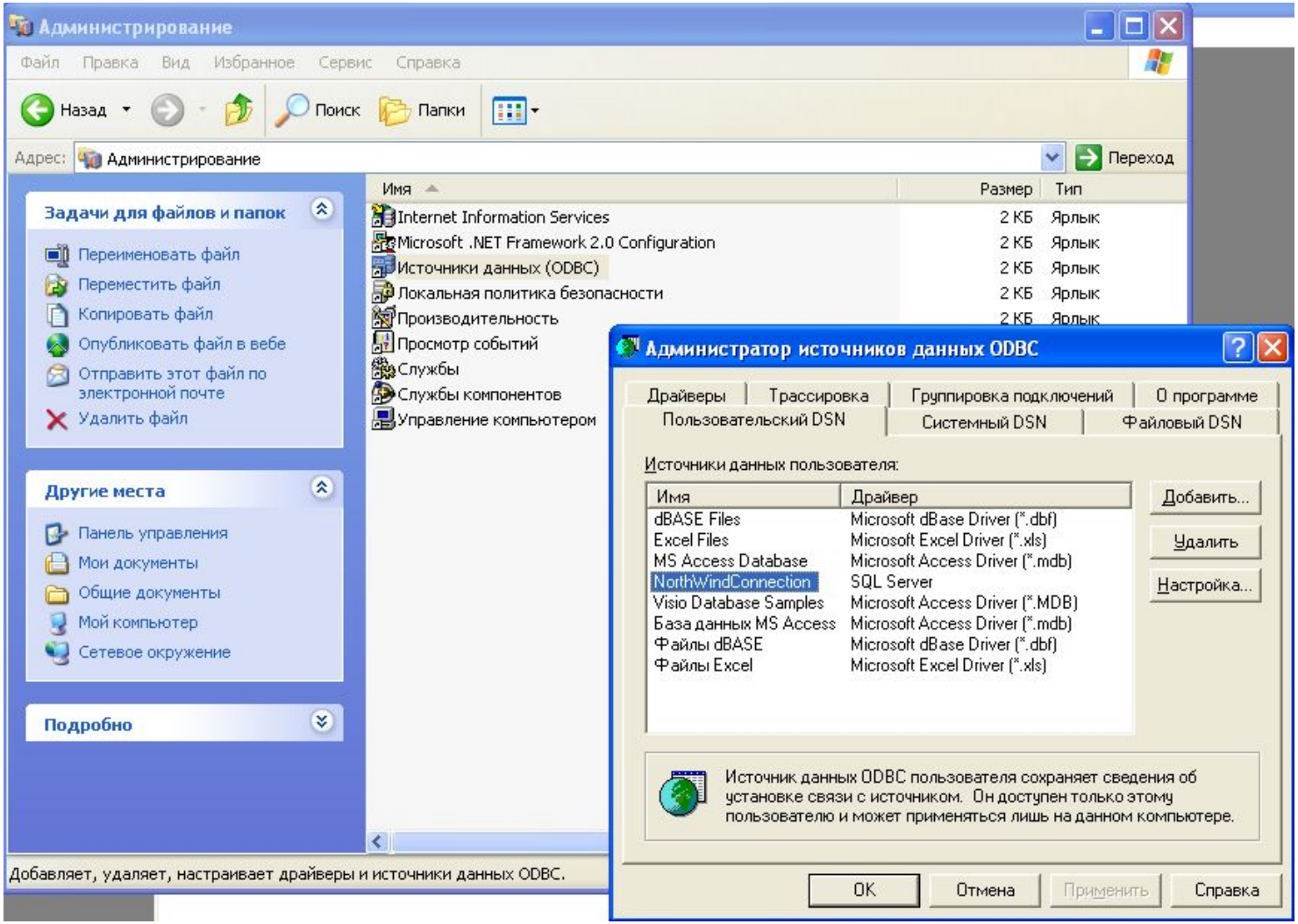
**Задачи, выполняемые Менеджером драйверов:**

- предоставляет стандартный интерфейс
- вызывает функции драйвера по указателю
- поддерживает список источников данных
- подгружает и выгружает не нужные драйверы
- проверяет часть ошибок

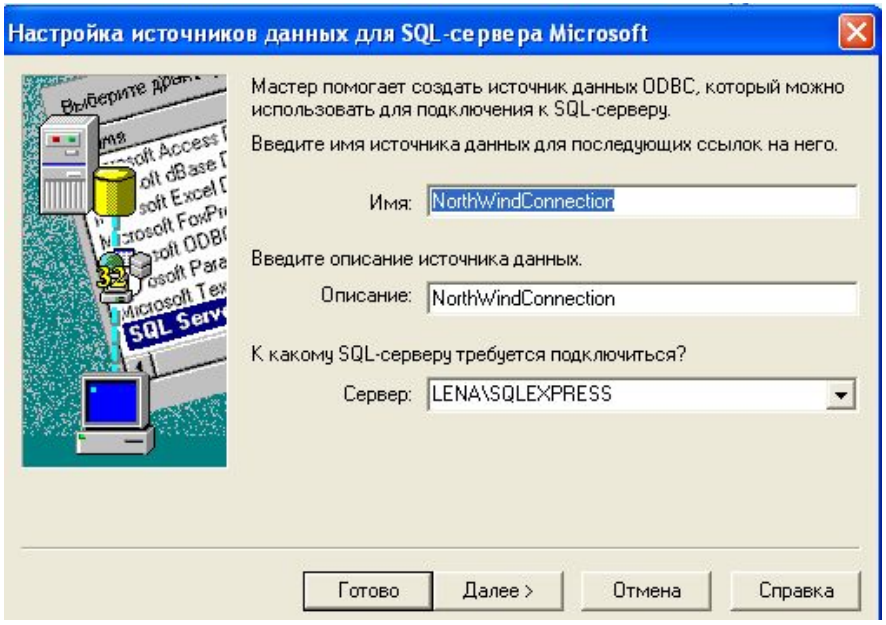
**Задачи, выполняемые Драйвером драйверов:**

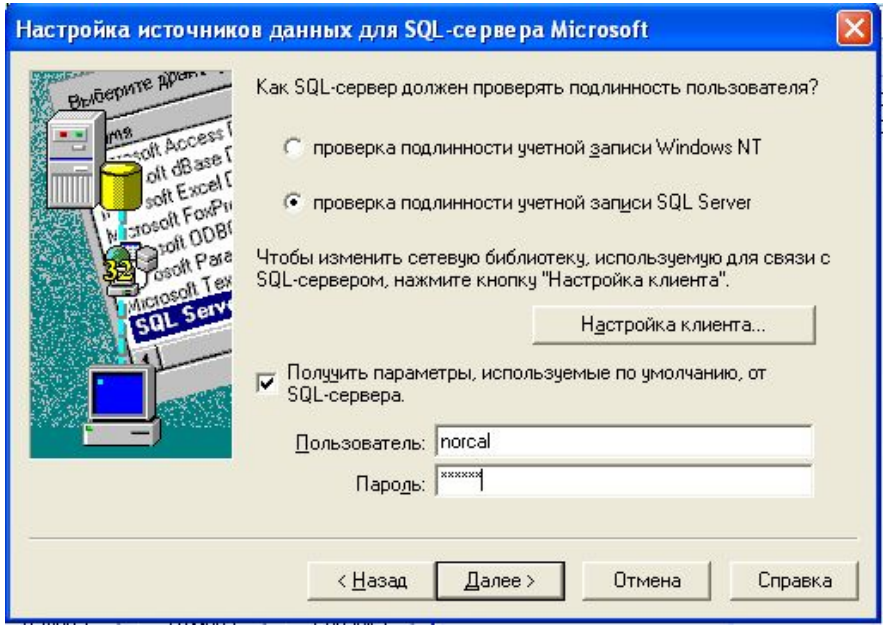
- подсоединяется к источнику данных
- выявляет ошибки
- преобразует SQL операторы в запросы, специфические для источника данных
- передает данные и выполняет преобразование типов

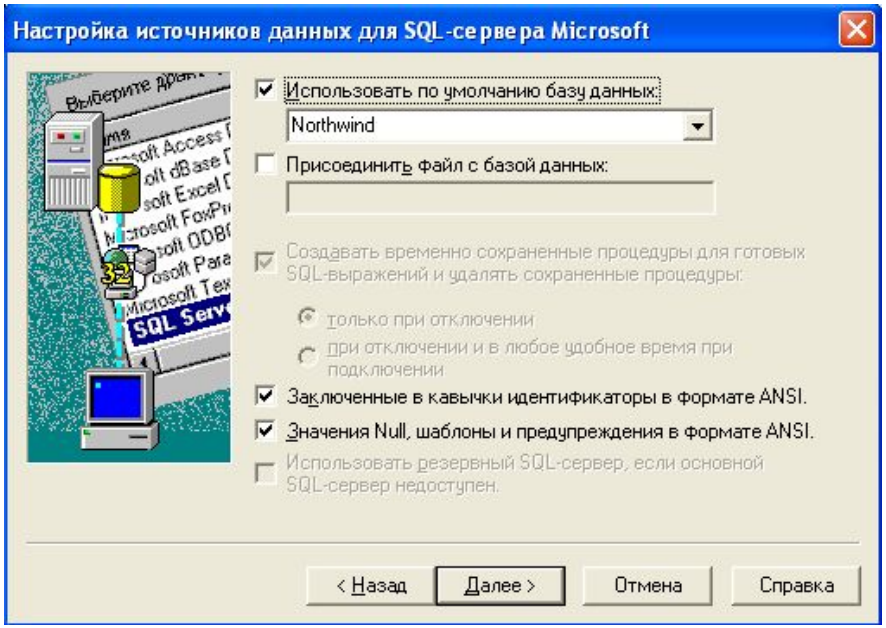
# ODBC

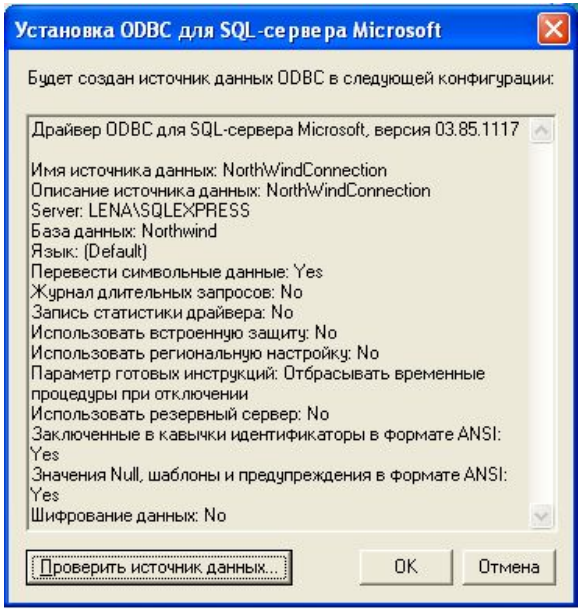

Администрирование
Файл Правка Вид Избранное Сервис Справка
Назад Поиск Папки
Адрес: Администрирование
Переход
Задачи для файлов и папок
Переименовать файл
Переместить файл
Копировать файл
Опубликовать файл в вебе
Отправить этот файл по электронной почте
Удалить файл
Другие места
Панель управления
Мои документы
Общие документы
Мой компьютер
Сетевое окружение
Подробно
Имя
Размер
Тип
Internet Information Services 2 КБ Ярлык
Microsoft .NET Framework 2.0 Configuration 2 КБ Ярлык
Источники данных (ODBC) 2 КБ Ярлык
Локальная политика безопасности 2 КБ Ярлык
Производительность 2 КБ Ярлык
Просмотр событий
Службы
Службы компонентов
Управление компьютером
Добавляет, удаляет, настраивает драйверы и источники данных ODBC.
Администратор источников данных ODBC
Драйверы
Трассировка
Группировка подключений
О программе
Пользовательский DSN
Системный DSN
Файловый DSN
Источники данных пользователя:
Имя
Драйвер
dBASE Files Microsoft dBase Driver (*.dbf)
Excel Files Microsoft Excel Driver (*.xls)
MS Access Database Microsoft Access Driver (*.mdb)
NorthWindConnection SQL Server
Visio Database Samples Microsoft Access Driver (*.MDB)
База данных MS Access Microsoft Access Driver (*.mdb)
Файлы dBASE Microsoft dBase Driver (*.dbf)
Файлы Excel Microsoft Excel Driver (*.xls)
Добавить...
Удалить
Настройка...
Источник данных ODBC пользователя сохраняет сведения об установке связи с источником. Он доступен только этому пользователю и может применяться лишь на данном компьютере.
OK
Отмена
Применить
Справка

# ODBC


Настройка источников данных для SQL-сервера Microsoft
Мастер помогает создать источник данных ODBC, который можно использовать для подключения к SQL-серверу.
Введите имя источника данных для последующих ссылок на него.
Имя: NorthWindConnection
Введите описание источника данных.
Описание: NorthWindConnection
К какому SQL-серверу требуется подключиться?
Сервер: LENA\SQLEXPRESS
Готово
Далее >
Отмена
Справка

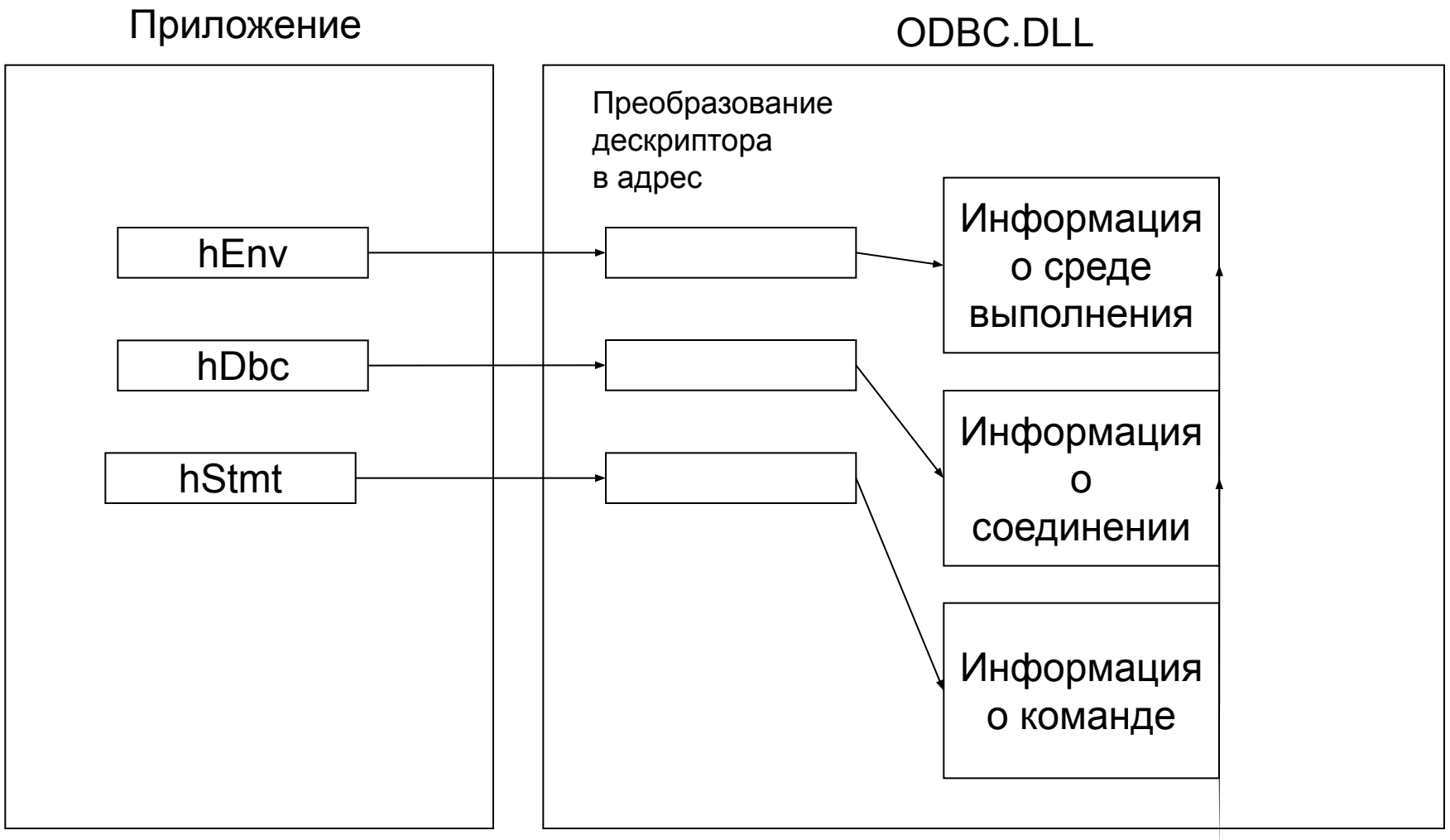

Настройка источников данных для SQL-сервера Microsoft
Как SQL-сервер должен проверять подлинность пользователя?
проверка подлинности учетной записи Windows NT
проверка подлинности учетной записи SQL Server
Чтобы изменить сетевую библиотеку, используемую для связи с SQL-сервером, нажмите кнопку "Настройка клиента".
Настройка клиента...
Получить параметры, используемые по умолчанию, от SQL-сервера.
Пользователь: norcal
Пароль: ******
< Назад
Далее >
Отмена
Справка


Настройка источников данных для SQL-сервера Microsoft
Использовать по умолчанию базу данных:
Northwind
Присоединить файл с базой данных:
Создавать временно сохраненные процедуры для готовых SQL-выражений и удалять сохраненные процедуры:
только при отключении
при отключении и в любое удобное время при подключении
Заключенные в кавычки идентификаторы в формате ANSI.
Значения Null, шаблоны и предупреждения в формате ANSI.
Использовать резервный SQL-сервер, если основной SQL-сервер недоступен.
< Назад
Далее >
Отмена
Справка


Установка ODBC для SQL-сервера Microsoft
Будет создан источник данных ODBC в следующей конфигурации:
Драйвер ODBC для SQL-сервера Microsoft, версия 03.85.1117
Имя источника данных: NorthWindConnection
Описание источника данных: NorthWindConnection
Server: LENA\SQLEXPRESS
База данных: Northwind
Язык: (Default)
Перевести символьные данные: Yes
Журнал длительных запросов: No
Запись статистики драйвера: No
Использовать встроенную защиту: No
Использовать региональную настройку: No
Параметр готовых инструкций: Отбрасывать временные процедуры при отключении
Использовать резервный сервер: No
Заключенные в кавычки идентификаторы в формате ANSI: Yes
Значения Null, шаблоны и предупреждения в формате ANSI: Yes
Шифрование данных: No
Проверить источник данных...
OK
Отмена

# ODBC

Дескрипторы и объекты



# ODBC


1. Соединение с СУБД
SQLAllocEnv(&hEnv)
SQLAllocConnect(hEnv, &hDbc)
SQLDriverConnect(hDbc, …)
SQLAllocStmt(hDbc, &hstmt)
2. Выполнение запроса
SQLExecDirect(hstmt, command, …) SQL запрос к СУБД
3. Получить результат запроса
SQLBindCol(hstmt, 1, …) привязка колонок к переменным
SQLFetch(hstmt) перемещение курсора
4. Отсоединение от СУБД
SQLDisconnect(hDbc)
SQLFreeConnect(hDbc)
SQLFreeEnv(hEnv)

# ODBC

```
#include "stdafx.h"
#include<iostream>

#undef  UNICODE
#include <windows.h>
#include <odbcinst.h>
#include <sql.h>
#include <sqlext.h>
#include <ctype.h>
#include <string>
#include <iostream>

using namespace std;


SQLCHAR connectionStringDSN[256] = "DSN=NorthWindConnection;UID=norcal;
PWD=norcal;";
SQLCHAR connectionStringDriver[256] =
  "DRIVER={SQL Server};SERVER=LENA\\SQLEXPRESS;"
  "UID=norcal;PWD=norcal;DATABASE=Northwind;";

void errorMessage(const char* s)
{
        cout<< "*** Error ***  " << s << endl;
}

void Message(const char* s)
{
        cout << s << endl;
}
```

# ODBC

```
void DoSelect(SQLCHAR connectionString[256])
{
	HENV    hEnv;
	HDBC    hDbc;
	HSTMT   hStmt = SQL_NULL_HSTMT;
	RETCODE rc;
	SQLSMALLINT cbOutConStr = 0;

	// Получить соединение с СУБД
	SQLCHAR outConnectionString[256];
	SQLAllocEnv(&hEnv);
	SQLAllocConnect(hEnv, &hDbc);
	rc = SQLDriverConnect(hDbc, NULL,
		connectionString, SQL_NTS,
		outConnectionString, sizeof(outConnectionString),
		&cbOutConStr, SQL_DRIVER_COMPLETE);
		//SQL_DRIVER_NOPROMPT SQL_DRIVER_COMPLETE

	if(!(rc == SQL_SUCCESS) && !(rc == SQL_SUCCESS_WITH_INFO))
	{
		errorMessage("Error SQLDriverConnect");
		return;
	}
```

# ODBC

```
SQLHSTMT hstmt;
rc = SQLAllocStmt(hDbc, &hstmt);
SQLCHAR command[256] = "SELECT EmployeeId, Address FROM Employees";
rc = SQLExecDirect(hstmt, command, SQL_NTS);

SQLINTEGER EmployeeId;
SQLCHAR Address[60];
SQLINTEGER cbEmployeeId, cbAddress;

if (rc == SQL_SUCCESS || rc == SQL_SUCCESS_WITH_INFO) {
        SQLBindCol(hstmt, 1, SQL_C_ULONG, &EmployeeId, 0, &cbEmployeeId);
        SQLBindCol(hstmt, 2, SQL_C_CHAR, Address, 60, &cbAddress);
        while (TRUE) {
                rc = SQLFetch(hstmt);
                if (rc == SQL_ERROR)
                {
                        errorMessage("Error in the fetch");
                }
                if (rc == SQL_SUCCESS || rc == SQL_SUCCESS_WITH_INFO)
                {
                        cout << EmployeeId << " " << Address << endl;
                } else {
                        break;
                }
        }
        SQLDisconnect(hDbc);
        SQLFreeConnect(hDbc);
        SQLFreeEnv(hEnv);
}
else
        errorMessage("Exec SQL error");
}
```

# ODBC

```
void ListDSN()
{
	const short SQL_MAX_DSN_LENGTH_ = SQL_MAX_DSN_LENGTH;
	UCHAR szDSN[SQL_MAX_DSN_LENGTH+1];
	UCHAR szDescription[256];
	short wDSNLen;
	SQLSMALLINT wDesLen;
	int retCode;
	SQLHENV hEnv = NULL;
	string DSNName;
	string resultString;
	string Descr;

	SQLAllocEnv(&hEnv);
	retCode = SQLDataSources(hEnv, SQL_FETCH_FIRST, szDSN, SQL_MAX_DSN_LENGTH_+1,
		&wDSNLen, szDescription, 256, &wDesLen);
	while(retCode == SQL_SUCCESS || retCode == SQL_SUCCESS_WITH_INFO)
	{
		DSNName = (string)((char *) szDSN);
		Descr = (string)((char *) szDescription);
		resultString += DSNName;
		resultString += "\n";
		retCode = SQLDataSources(hEnv, SQL_FETCH_NEXT, szDSN,SQL_MAX_DSN_LENGTH_+1,
			&wDSNLen, szDescription, 256, &wDesLen);
	}

	SQLFreeEnv(hEnv);
	Message(resultString.c_str());
}
```

# ODBC

```
void ShowMenu()
{
	int choice = 0;
	bool done = false;
	do
	{
		cout << "\n*** MENU ***" << endl;
		cout << "0 - exit" << endl;
		cout << "1 - list all DNS" << endl;
		cout << "2 - select (using DNS)" << endl;
		cout << "3 - select (using driver directly)" << endl;
		cout << "Choice : ";
		cin >> choice;
		switch(choice)
		{
		default : done = true; break;
		case 1 : ListDSN(); break;
		case 2 : DoSelect(connectionStringDSN); break;
		case 3 : DoSelect(connectionStringDriver); break;
		}
	}while(!done);
}


int main(int count, char* cparams[])
{
	ShowMenu();
	return 0;
}
```