**Раздел 4. Разработка приложений для работы с базой данных**

**Тема 4.1** Обзор технологий доступа к данным

**Тема 4.2** Описание интерфейса среды. Компоненты визуализации данных

**Тема 4.1 Обзор технологий доступа к данным**

Вопросы:

* 1. Обзор технологий доступа к данным.
  2. Сравнительный анализ технологий.

**Цель**: сформировать знания основных технологий доступа к данным, провести сравнительный анализ технологий.

**Обзор технологий доступа к данным**

На сегодняшний день существует множество технологий доступа к данным, таких как BDE, OLE, ODBC, DАО, АDО, и до сих пор разрабатываются новые, более надежные, удобные в работе и более быстродействующие технологии.

Механизмы доступа к **базам данных** снижают сложность обмена информацией с базами, однако интерпретация результатов их работы также достаточно трудоемка. Поэтому реализованы наборы компонентов, предназначенные для взаимодействия с механизмами обмена.

Таким образом, можно выделить несколько субъектов, участвующих в движении информации между **базой данных** и приложением (например, пользовательским интерфейсом):

1) интерфейсная часть приложения или его программная часть, манипулирующая информацией, хранимой в базе данных.;

2) компоненты, обеспечивающие связь приложения с механизмом доступа к базе данных;

3) механизм доступа к базе данных;

4) база данных.

На рисунке 1 представлена схема движения информации между приложением и базой данных.

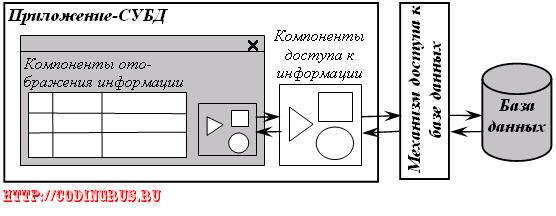


Рисунок 1 – Движение информации от базы данных к приложению

Из рисунка видно, что при разработчике приложения-СУБД программист работает с наборами компонентов, предназначенных для обмена информацией с базами данных и ее отображения. В зависимости от выбранного механизма доступа к базе данных некоторые наборы компонентов могут не использоваться, однако все они, вне зависимости от особенностей используемой базы данных и механизма доступа к ней, имеют схожие свойства и методы.

**Сравнительный анализ технологий**

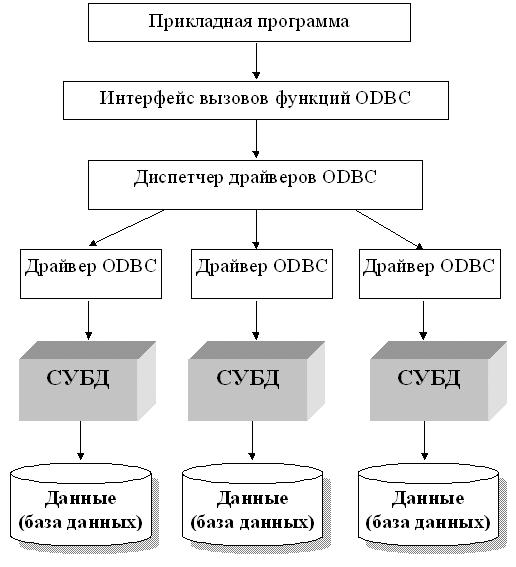
**ODBC (**Open Database Connectivity – открытый доступ к базам данных**) –** разработанный компанией Microsoft универсальный интерфейс программирования приложений для доступа к базам данных.

Основной целью разработки протокола ODBC считается стандартизация механизмов взаимодействия с различными СУБД. Основная проблема, связанная с разработкой приложений, взаимодействующих с базами данных на основе специальных SQL API, состояла в том, что каждая СУБД имела собственный программный интерфейс доступа, каждый из них имел свои особенности и функционировал не совсем так, как другие. В связи с этим разработка приложения существенно зависела от используемой СУБД. Компания Microsoft сделала важный шаг для решения этой проблемы. Основная идея заключалась в разработке универсального интерфейса на уровне семейства операционных систем Windows, который мог бы быть поддержан в разных СУБД.

Рассмотрим кратко структуру программного обеспечения ODBC:

* **интерфейс вызовов функций ODBC**: это так называемый верхний уровень ODBC, содержащий API, который и используется непосредственно приложениями. Данный API реализован в виде библиотеки динамической компоновки Dll и входит в состав операционной системы Windows;
* **драйверы ODBC**: это так называемый нижний уровень ODBC, содержащий набор драйверов для СУБД, поддерживающих протокол ODBC. В рамках технологии для каждой СУБД может быть разработан соответствующий ODBC-драйвер, который будет являться промежуточным звеном между прикладной программой и СУБД, транслируя вызовы функций СУБД в вызовы внутренних специализированных функций СУБД. Таким образом решается проблема стандартизации. Для многих современных СУБД существуют специализированные драйверы ODBC, отдельно устанавливаемые в операционную систему;
* **диспетчер драйверов ODBC**: данный программный механизм представляет средний уровень ODBC, управляя процессом загрузки необходимых драйверов.

Схема выполнения программы с использованием протокола ODBC для доступа к данным приводится на рисунке 2.



**Рисунок 2 –** Схема выполнения программы с использованием протокола ODBC для доступа к данным

Операционная система Windows имеет в своем составе несколько механизмов доступа к базам данных: *ODBC*, *OLE DB* и *ADO*.

**Технология ODBC** (*Open Database Connectivity* – открытый механизм доступа к базам данных) - это компонент операционной системы *Windows*, предназначенный для унификации доступа к информации, хранящейся в **базах данных** различных видов. *ODBC* состоит из набора драйверов, осуществляющих операции обмена с определенными **базами данных**, и менеджера драйверов, осуществляющего передачу запросов от приложения к драйверу и передачу информации от драйвера к приложению (рисунок 3).

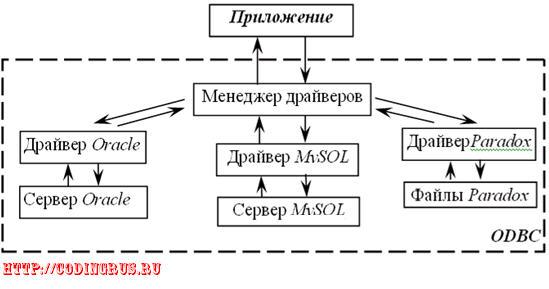


Рисунок 3 – Движение информации между приложением и БД при использовании ODBC

Для получения и изменения данных используется язык запросов **SQL**, вне зависимости от того, поддерживается ли он базой данных, к которой обращается приложение. Если база данных не поддерживает язык **SQL**, то доступ к ней не отличается от доступа к **БД**, поддерживающим **SQL**. В этом и заключается унификация доступа к базам данных системой **ODBC** – приложение указывает название драйвера, который должен использоваться для подключения к базе данных, и передает запрос, в котором описан состав требуемой информации. Далее механизм **ODBC** выполняет все необходимые операции по получению информации, скрывая от приложения специфику работы с конкретной базой данных. Доступ приложений к **ODBC** осуществляется через **API**-функции, реализованные в динамических библиотеках.

**Технология OLE DB** (*Object Linking and Embedding DataBase* – механизм доступа к базам данных с помощью технологии *OLE*) представляет собой набор **СОМ**-интерфейсов, обеспечивающих универсальный доступ приложений к базам данных и другим хранилищам информации (в том числе и нереляционным - файловым системам, системам электронной почты и др.), в которых результатом за-просов является информация, представленная в табличной форме.

Характерной особенностью механизма *OLE DB* является то, что данные, возвращаемые *OLE DB* приложению, представляют собой не просто массив информации, а *СОМ*-объект, обладающий, помимо самой информации, методами управления этой информацией (например, фильтрация и сортировка). Логическая схема доступа к данным с помощью *OLE DB* представлена на рисунке 4.

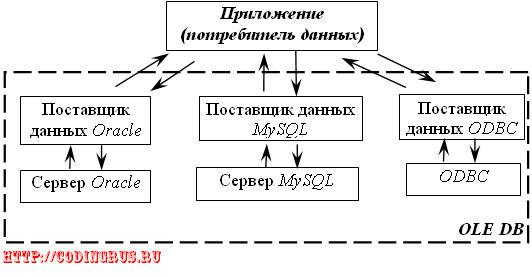


Рисунок 4 – Движение информации между приложением и БД при использовании OLE DB

Приложение, называемое потребителем данных, обращается к поставщику данных, который перенаправляет запрос от приложения к базе данных, либо обрабатывает запрос самостоятельно. Таким образом, поставщик данных аналогичен драйверам *ODBC* и *BDE*. Механизм *OLE DB* имеет доступ к *ODBC*, поддерживая, таким образом, большое количество драйверов, реализованных для *ODBC*.

Недостатком *OLE DB* (и соответственно технологии *ADO*, построенной на *OLE DB*) является более низкое быстродействие по сравнению с механизмами *ODBC* и *BDE*. Это связано с использованием **СОМ**-объектов, применение которых нагружает операционную систему. Однако работа механизмов связи с базой данных при использовании *OLE DB* обычно занимает меньшее время по сравнению с работой самой базы данных по выдаче запроса.

**Технология ADO** (от англ. *ActiveX Database Objects* – механизм доступа к базам данных через объекты ActiveX) – технология доступа к данным, разработанная *Microsoft*, является надстройкой над механизмом доступа *OLE DB* и предназначена для унификации работы с поставщиками данных *OLE DB*. Обеспечивает удобный и надежный доступ к данным, хотя и несколько более медленный, чем в технологиях *BDE* и *dbExpress*. Хорошо подходит для работы с системами управления базами данных (СУБД) от фирмы *Microsoft* (*MS Access, MS SQL Server*), поскольку не требует дополнительных компонентов (библиотек, драйверов), так как они уже есть на компьютере пользователя. Схема доступа к информации с помощью *ADO* представлена на рисунке 5.

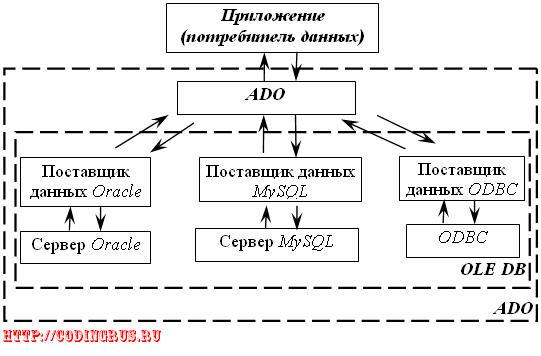


Рисунок 5 – Движение информации между приложением и БД при использовании ADO

Механизм **ADO** предоставляет несколько основных **СОМ**-объектов, используемых для получения и управления информацией (имеются дополнительные **СОМ**-объекты, расширяющие функциональность *ADO*):

• Connection для управления соединением с базой данных и передачи запросов поставщику данных;

• Command для управления информацией о запросе к базе данных или команде;

• Recordset, содержащий таблицу, которая является результатом запроса к базе данных;

• Field, содержащий описание поля в таблице, возвращенной поставщиком данных. Список всех полей таблицы содержится в подобъекте Fields объекта RecordSet;

• Error, содержащий расширенную информацию об ошибке, о которой сообщил поставщик данных. Если ошибок несколько, доступ к ним можно получить с помощью объекта Errors.

**BDE** (англ. *Borland Database Engine* – механизм доступа к базам данных фирмы *Borland*) – базовая технология доступа к БД от фирмы *Borland*, является аналогом *ODBC* и имеет схожую с ней архитектуру. Этот механизм позволяет получать доступ к реляционным БД с помощью специальных *BDE* драйверов или через *ODBC* драйверы. Исторически является первой технологией доступа к БД в средах *Borland*. Существенным недостатком использования этой технологии является достаточно трудоемкий процесс развертывания программы работы с базой данных: помимо самой программы на компьютере пользователя необходимо установить *BDE* и выполнить его настройку, а также отсутствие поддержки со стороны фирмы-разработчика.

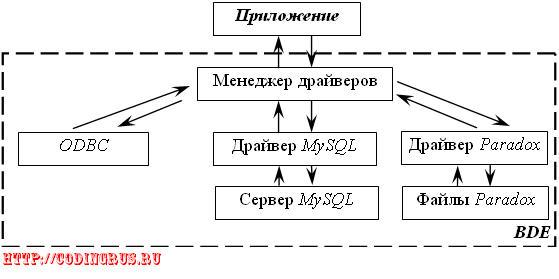


Рисунок 6 – Движение информации между приложением и БД при использовании BDE

**dbExpress** – более новая и совершенная технология доступа к удаленным БД от фирмы *Borland*. Схема работы этого механизма аналогична схеме работы *BDE*, но при использовании *dbExpress* не происходит обращения к *ODBC*. При использовании этого механизма нет необходимости развертывания *dbExpress* вместе с приложением: достаточно установить только **dll**-библиотеку либо прилинковать драйвер к исполняемому файлу. Технология *dbExpress* по сравнению с *BDE* обеспечивает более быстрый и удобный доступ к данным.

**InterBase** – специализированная технология доступа к серверу БД *Borland InterBase*. При работе с этой СУБД обеспечивает наивысшую производительность по сравнению с другими технологиями.

Принципы работы с БД с использованием любой из вышеперечисленных технологий практически одинаковы.

**Протокол JDBC**

JDBC (Java Database Connectivity) представляет собой API для выполнения SQL-запросов к базам данных из программ, написанных на языке Java.

С развитием глобальных сетей, в частности Интернета, и всех сопутствующих технологий стали появляться новые языки, специально предназначенные для работы в новых условиях. Одним из таких языков является язык программирования Java. В настоящее время Интернет-приложения занимают существенное место на рынке, работая в рамках 2-, 3- и многозвенной архитектуры. При этом значение языка Java как средства создания приложений, работающих с базами данных, существенно возрастает. Именно это и явилось одной из основных причин разработки нового программного интерфейса – JDBC. Первоначально интерфейс JDBC был разработан компанией Sun Microsystems, в настоящий момент этот API поддерживается всеми ведущими коммерческими СУБД.